1 General

This manual is intended for personnel technically qualified to install, operate and maintain the products which are described herein. It contains all the necessary information for correct use of the products. However, for advanced use of our products please contact your nearest sales office for additional information.

The contents of this manual are not contractual and cannot under any circumstance extend or restrict contract warranty clauses.

2 Qualification of personnel

Only **qualified personnel** are authorized to install, operate or maintain the products. Any work performed by a unqualified person or non-observance of the safety instructions in this document or attached to the equipment may risk the safety of personnel and/or cause irreparable damage to equipment. The following personnel may be regarded as being **"Qualified"** :

- those involved with application design. Design office personnel familiar with control system safety concepts (for example, design engineers, etc),
- those involved with equipment installation. Individuals who are familiar with the installation, connection and startup of control system equipment (for example installers or wiring technicians working during the installation phase, technicians setting up the equipment, etc),
- those involved with operation. Personnel trained to operate and manage control system equipment (for example, operators, etc),
- those performing preventive or corrective maintenance. Personnel who are trained and experienced in the adjustment and repair of control system equipment (for example, installation engineers, after sales service engineers, etc).

3 Warnings

Warnings serve to prevent specific risks encountered by personnel and/or equipment. They are indicated in the documentation and on the products by different warning symbols, according to the severity of the risk :

Danger or **Caution or Attention**

Indicates that not following instructions or ignoring the warning may cause serious personal injury, death and/or serious damage to equipment.

Warning or **Important** or 

Indicates that not following a specific instruction may lead to minor injury and/or damage to equipment.

Note or **Comment**

Highlights important information relating to the product, its operation or its accompanying documentation.

4 Conformity of use

The products described in this manual **conform to the European Directives** (*) to which they are subject (CE marking). However, they can only be used correctly in the context of the applications for which they are intended (described in the various documents) and when connected to approved third party products.

As a general rule, if all handling, transport and storage specifications are observed, and all instructions for installation, operation and maintenance are followed, the products will be used correctly, with no danger to personnel or equipment.

(*) DEMC and DLV Directives, concerning Electromagnetic Compatibility and Low Voltage.

5 Installing and setting up equipment

It is important to observe the following rules when installing and starting up equipment. In addition, if the installation includes digital links, it is essential to follow the basic wiring rules given in the user's guide**, reference TSX DG GND,** and manual **TSX DR NET**, part C.

- safety instructions must be followed meticulously. These instructions are in the documentation or on the equipment being installed and set up.
- the type of equipment defines the way in which it should be installed :
  - a flush-mountable device (for example, a process control terminal or a cell controller) must be flush-mounted,
  - a device which is to be built in (for example, PLC) must be placed in a cabinet or enclosure,
  - the casing of a laptop or portable device (for example, a programming terminal or a notebook) must remain closed,
- if the device is permanently connected, its electrical installation must include a device to isolate it from the power supply and a circuit-breaker to protect it against overcurrents and isolation faults. If this is not the case, the power socket must be grounded and be easily accessed. **In all cases, the device must be connected to the protective mechanical ground PG using green/yellow wires (NFC 15 100)**.
- low voltage circuits (even though they are low voltage) must be connected to the protective ground so that dangerous voltages can be detected.
- before a device is powered up, its nominal voltage must be checked to ensure that it has been adjusted to conform with the supply voltage.
- if the device is supplied with 24 or 48 VDC, the low voltage circuits must be protected. Only use power supplies which conform to the standards currently in force.
- check that the supply voltages remain within the tolerance ranges defined in the technical characteristics of the devices.
- all measures must be taken to ensure that any power return (immediate, warm or cold) does not lead to a dangerous state which may risk personnel or the installation.
- emergency stop devices must remain effective in all the device's operating modes, even those which are abnormal (for example, when a wire becomes disconnected). Resetting these devices must not cause uncontrolled or improper restarts.
- cables which carry signals must be located where they do not cause interference with the control system functions by capacitive, inductive or electromagnetic interference.
- control system equipment and their control devices must be installed in such a way as to ensure that they are protected against unintentional operation.
- appropriate safety measures must be taken for the inputs and outputs, to prevent improper states in the control system device, if no signal is received.

6 Equipment operation

The operational safety and availability of a device is its ability to avoid the appearance of faults and to minimize their effects if they occur.

A system is said to be fail-safe if the appearance of faults **never** causes a dangerous situation.

A fault inside the control system is known as :

- passive, if it results in an open output circuit (no command is sent to the actuators).
- active, if it results in a closed output circuit (an command is sent to the actuators).

From the safety point of view, a given fault is dangerous or not depending on the type of command given during normal operation. A passive fault is dangerous if the normal command is the operation of an alarm. An active fault is dangerous if it maintains or activates an undesirable command.

It is important to note the basic difference between the behavior of an electromechanical relay and an electronic component (for example a transistor) :

- there is a high probability, approximately 90%, that the failure of a relay will cause an open circuit (control circuit powered off).
- there is a 50% probability that the failure of a transistor will cause either an open circuit or a closed circuit.

This is why it is important to correctly estimate the types and consequences of faults when automating a system using electronic products such as PLCs, including when relay output modules are used on PLCs.

The system designer must **use devices external to the PLC** to protect against active faults inside the PLC, which are not indicated and are judged to be dangerous to the application. This may require solutions from various different technologies such as mechanical, electromechanical, pneumatic or hydraulic devices (for example, directly wiring a limit switch and emergency stop switches to the coil of a movement control contactor).

To protect against dangerous faults which may occur on output circuits or preactuators, it is sometimes beneficial to resort to general principles and use the large processing capacity of PLCs, for example by using "inputs to check the correct execution of commands requested by the program".

7 Electrical and thermal characteristics

Details of the electrical and thermal characteristics of devices are given in the associated technical documents (installation manuals, service instructions).

8 Environmental conditions

Devices, such as TSX Nano PLCs, meet "TC" treatment (1) requirements. For installations in industrial production workshops or in environments which correspond to "TH" treatment (2), these devices should be installed in enclosures with minimum IP54 protection, stipulated by standards IEC 664 and NF C 20 040.

TSX Nano PLCs, which themselves have IP20 protection, can therefore be installed without enclosures in restricted-access locations which do not exceed pollution level 2 (control room with no machines or dust-producing activities).

(1) "TC" treatment : all climate treatment.
(2) "TH" treatment : treatment for hot and humid environments.

9 Preventive or corrective maintenance

Availability

The availability of a system is its ability, in terms of its combined reliability, maintainability and maintenance logistics, to be in a state to perform a required function, at a given moment and within a defined time period.

Availability is therefore specific to each application, since it is a combination of :

- the architecture of the automatic system,
- the reliability and maintainability : intrinsic characteristics of the equipment (PLCs, sensors, machine, etc),
- maintenance logistics : characteristic intrinsic to the user of the control system (software structure, fault indication, process, on-site replacement parts, training of personnel).

Troubleshooting procedure

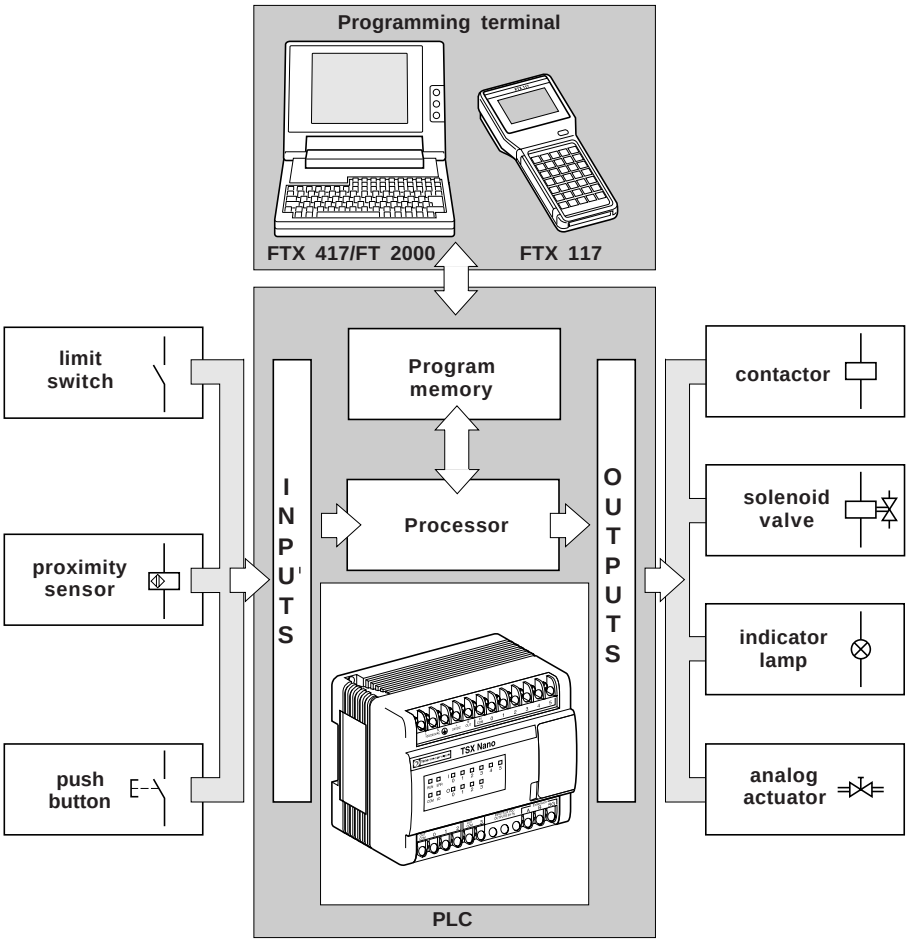
- control system equipment should only be repaired by qualified personnel (after sales service engineer, or technician approved by AEG Schneider Automation). Only certified replacement parts or components should be used.
- before performing any operation on equipment (for example opening an enclosure), always cut the power supply off (disconnect the power plug or open the power isolation switch).
- before performing any "mechanical" operation on equipment on site, cut the power supply off and mechanically lock any moving parts.
- before performing an operation on the PLC, modifying a connection, etc, check in the manual whether this should be done with the power off or if it is possible while the device is powered up. Meticulously follow the instructions given in the manual.
- on positive logic outputs or negative logic inputs, take all necessary precautions to prevent a disconnected wire from coming into contact with the mechanical ground (risk of undesirable command).

Section **Page**

1.1 The PLC in the control system structure

A PLC can be divided into four main sections :
- Inputs
- Outputs
- The memory, where the user program instructions and data are stored
- The processor, which reads the input data and controls the outputs according to the user program instructions.

The programming terminal is the tool used to :
- Create and transfer the user program to program memory
- Debug the user program and control system start-up
- Perform installation diagnostics.

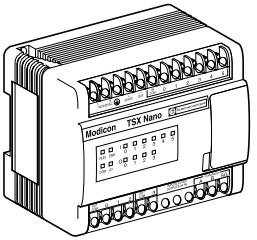


1.2 Discrete I/O TSX Nano PLCs

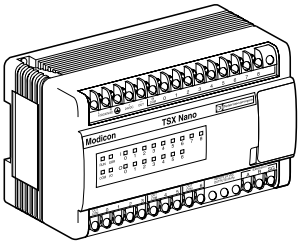
1.2-1 Presentation

Discrete I/O TSX Nano PLCs are available in four configurations depending on whether extendability is required :

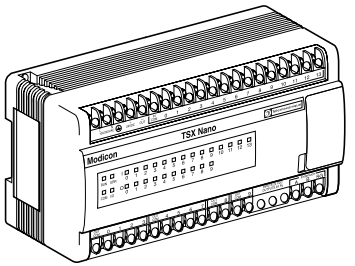
- "base PLC" non-extendable PLCs (14 and 20 I/O).
- "base PLC" non-extendable PLCs with 1 integrated analog input (10, 16 and 24 I/O).
- "I/O extensions" (16 and 24 I/O).
- "base PLC or I/O extension" extendable PLCs (10, 16 and 24 I/O).



10 I/O (6 inputs + 4 outputs)

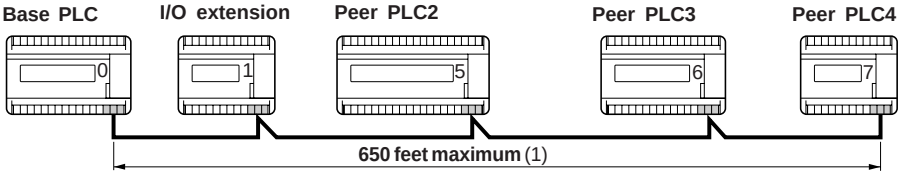


14 I/O (8 inputs + 6 outputs)
16 I/O (9 inputs + 7 outputs)



16 I/O (9 inputs + 7 outputs) PLC with 115 VAC inputs
20 I/O (12 inputs + 8 outputs)
24 I/O (14 inputs + 10 outputs)

A "base PLC" extendable PLC **can be extended by an I/O extension or by an extendable PLC configured as an I/O extension**.
In addition, a maximum of three peer PLCs (extendable PLCs configured as peer PLCs) can be connected to the "base PLC" extendable PLC, communicating via exchange words.

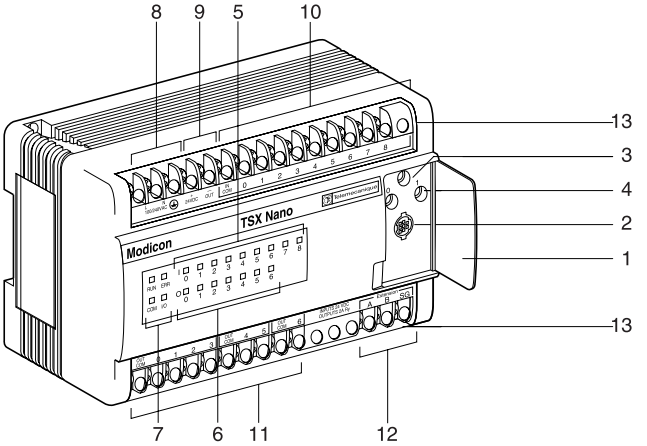


Discrete I/O TSX Nano PLCs are programmed in PL7 language (Ladder or Instruction List languages).

These PLCs are programmed via :
- either an FTX 117 terminal (Instruction List)
- or an FTX 417 or PC compatible terminal (Ladder or Instruction List).

(1) See Part A, Section 3.5 for the type of cable to use

Description of discrete I/O TSX Nano



1	Hinged cover for access to items **2**, **3**, and **4.**		All PLCs
2	Port for connecting an ASCII or UNI-TELWAY device. ASCII, RS485 UNI-TELWAY Master or slave protocols.		B, B/E
3	Selector switch for PLC function code : 0 = base PLC, 1 = I/O extension. 5 = Peer PLC no. 2, 6 = Peer PLC no. 3, 7 = Peer PLC no. 4.		B/E
4	Potentionmeter : • 1 on PLCs with 10, 14 or 20 I/O, • 2 on PLCs with 16 or 24 I/O,		B, B/E
5	Input status display (1),		All PLCs
6	Output status display (1),		
7	PLC status display : RUN, ERR, COM, I/O,		
8	Power supply connections,		
9	Sensor supply on models with 100/240 VAC : 24VDC/150 mA supply. This supply is not available on the model with 115VAC inputs.		
10	Input connections.		
11	Output connections.		
12	Extension connector	I/O Extension	E, B/E
		Peer PLC	B/E
		Modbus Slave (V3)	B/E
	Analog input channel		see 1.11
13	Removable cover for protecting the terminals.		All PLCs

(1) A maximum of 16 internal bits can be displayed (%S69=1).
B : Non-extendable PLCs, E : I/O extension, B/E : Extendable PLCs.

1.2-2 Summary of catalog references (Discrete I/O)

No. I/O	PLC type	Power supply 110/240V ∼	Power supply 24V ⎓	Type of inputs isolated 24V ⎓	Type of inputs 115V ∼	Type of outputs transistor 24V ⎓ sink	Type of outputs transistor 24V ⎓ source	Type of outputs relay 24V ⎓ 24/240V ∼	References
	B/E	•		•				•	TSX 07 (x)0 1028
	B/E	•		•		•			TSX 07 (x)0 1008
6/4	B/E		•	•		•			TSX 07 (x)0 1002
	B/E		•	•				•	TSX 07 (x)0 1022
	B/E		•	•			•		TSX 07 (x)0 1012
	B/E	•		•				•	TSX 07 32 1028
8/6	B	•		•				•	TSX 07 3L 1428
	B/E	•		•				•	TSX 07 (x)1 1628
	B/E	•			•			•	TSX 07 (x)1 1648
9/7	B/E	•		•		•			TSX 07 (x)1 1608
	B/E		•	•		•			TSX 07 (x)1 1602
	B/E		•	•				•	TSX 07 (x)1 1622
	B/E		•	•			•		TSX 07 (x)1 1612
	E	•		•				•	TSX 07 EX 1628
	E		•	•			•		TSX 07 EX 1612
	B/E	•		•				•	TSX 07 33 1628
12/8	B	•		•				•	TSX 07 3L 2028
	B/E	•		•				•	TSX 07 (x)1 2428
	B/E	•		•		•			TSX 07 (x)1 2408
	B/E		•	•		•			TSX 07 (x)1 2402
14/10	B/E		•	•				•	TSX 07 (x)1 2422
	B/E		•	•			•		TSX 07 (x)1 2412
	E	•		•				•	TSX 07 EX 2428
	E		•	•			•		TSX 07 EX 2412
	B/E	•		•				•	TSX 07 33 2428

B : Non-extendable PLC, E : I/O extension, B/E : Extendable PLC.
(x) 2 : V2 models; 3 : V3 models

Note :
TSX 07 33• •••• models have additional software functions to those offered by the TSX 23• •••• models, such as Uni-Telway Master/Slave on the terminal port, Modbus slave PLC on the extension port, connection of analog I/O modules, etc.

1.2-3 Main functions of TSX Nano PLCs

By default, all I/O are configured as discrete I/O. However, certain I/O can be assigned to specific tasks during configuration (RUN/STOP input, latching inputs, 10 kHz fast counting I/O or 1kHz up/down counting, PLC status (SECURITY) output, pulse train output, pulse width modulation output).
TSX Nano PLCs are programmed in reversible PL7 language (Instruction List or Ladder) which enables the following functions to be used : schedule blocks (RTC), timers, up/down counters, LIFO/FIFO registers, shift registers, drum controllers, step counters.

Scanning	Normal (cyclical) or periodic (2 to 150 ms)		
Scan time	Less than 1 ms for 1000 elementary instructions (3) Less than 0.6 ms for 100 elementary instructions		
Execution time	0.2 µs to 2 µs for a single Boolean elementary instruction		
Memory capacity	Data	256 internal words, 64 constant words, 128 internal bits (64 of these retentive)	
	Program	- 1000 instructions (battery-backed RAM and EEPROM)	
Backup	PLC RAM : by battery. Backup duration : 30 days		
Language	Reversible PL7 : Instruction List or Ladder		
Terminal port	RS 485 link, UNI-TE protocol, 9600 bits/s / 19200 bits/s. Max. distance : FTX 117 : 10 m; UNI-TE : 50 m		
I/O extension	1 per PLC. Max. distance between base PLC and extension : 200 meters (650 feet) (1)		
Peer PLCs	3, connected to the "base PLC" extendable PLC, and communicating via exchange words. Maximum distance between base PLC and last peer PLC : 200 meters (650 feet) (1)		
Modbus link	Non-isolated RS485 type, length limited to 200 m. ASCII or RTU mode		
Function blocks	Schedule blocks (RTC)	16	(2)
	Timers	32	time base : 1ms (for the first two), 10ms, 100ms, 1s, 1min with a preset range from 0 to 9999
	Up/down counters	16	preset range from 0 to 9999
	LIFO/FIFO registers	4	16-word blocks
	Shift registers	8	16 bits

(1) See Part A, Section 3.5 for the type of cable to use
(2) Available on all "base PLC" PLCs with 16 and 24 I/O : TSX 07 •• 16/24
To improve performance, it is advisable to deconfigure the unused extensions (see Section A-1.6).

Function blocks (continued)	Drum controllers	4	8 steps, 16 control bits
	Step counters	4	256 steps
Potentiometers	1 (on TSX Nano, 10/14/20 I/O), 2 (on TSX Nano 16/24 I/O)		
Analog channel	Function specific to certain modules, see Section 1.11		
Displaying internal bits (Memory Display)	On a "base PLC" PLC or a peer PLC (extendable PLC configured as a peer PLC), it is possible to display the status of 8 internal bits (TSX Nano with 10 and 16 I/O) or 16 internal bits (TSX Nano with 24 I/O) on the front panel, see Section 1.9.		
Programmable input filter	On a base PLC or a peer PLC, it is possible to change the input filter time during configuration : No filtering or filtering at 3 ms or 12 ms (see Section A-1.7). I/O points are configured in groups.		
Special I/O	It is possible to assign specific functions to certain I/O during configuration.		
	Inputs	RUN/STOP : 1 of the first 6 inputs of the base or the peer PLC (%I0.0 to %I0.5).	
		Latching : the first 6 inputs of the base or the peer PLC (%I0.0 to %I0.5).	
		Analog input module connected on %10.0 according to frequency meter	
		Fast counter : 10 kHz Frequency meter : 10 kHz Fast up/down counter : 1 kHz	
	Outputs	PLC status (SECURITY) : 1 of the first 4 outputs of the base or the peer PLC (%Q0.0 to %Q0.3)	
		PULSE : pulse train (4.9 kHz maximum)	
		PWM : pulse width modulation (4.9 kHz maximum)	
		Analog input module connected on %Q0.0 according to PWM (pulse width modulation)	
		Threshold outputs : 2 (%Q0.1 and %Q0.2), associated with fast counting, are updated without waiting for an end-of-scan update	

Note :
PULSE and PWM outputs can be used on PLCs with relay outputs as long as the on-off period is greater than the relay's closing response time (approximately 50 Hz). If they are used, the maximum number of permitted operations is likely to be reached very quickly.
We therefore recommend using these outputs mainly on PLCs with transistor outputs.

Summary of the main functions according to PLC type

Functions		10/14/16 I/O PLC (x)0 10 •• (x)1 16 •• 3L 1428 3(x) ••28	16 I/O PLC (x)1 1648	20/24 I/O PLC 3L 2028 (x)1 24 •• 33 2428	I/O extension 16/24 E/S EX •• ••
Inputs	RUN/STOP	●	●	●	
	Latching	●		●	
	Counting 10 kHz Analog/ Frequency meter	●		●	
	Up/down counting 1kHz	●		●	
Outputs	PLC status	●	●	●	
	PULSE	●	●	●	
	Analog/ PWM	●	●	●	
	Threshold (fast counters)	●		●	
Programmable input filters		(x)1 16 •• 3L 1428		●	
Schedule block (RTC)		(x)1 16 ••	●	(x)1 24 ••	
Potentiometers		●	●	●	
Display of internal bits on I/O indicator lamps		●	●	●	
Communication					
prog. port	UNI-TELWAY	●	●	●	
	ASCII	●	●	●	
extension port	I/O Ext. or Peer PLC	(x)0 10 ••	●	(x)1 24 ••	
	Modbus slave	(x)1 16 ••	●	(x)1 24 ••	
Function blocks	Timers	-	●	●	
	Up/down counters	●	●	●	
	LIFO/FIFO register blocks	●	●	●	
	Shift registers	●	●	●	
	Drum controllers	●	●	●	
	Step counters	●	●	●	
Instruc-tions (1)	Grafcet	●	●	●	
	Master relay	●	●	●	
Varia-bles (1)	Bit strings	●	●	●	
	Word table	●	●	●	
	Indexing	●		●	

(1) All other instructions can be accessed by the various types of PLC.

1.3 Program execution

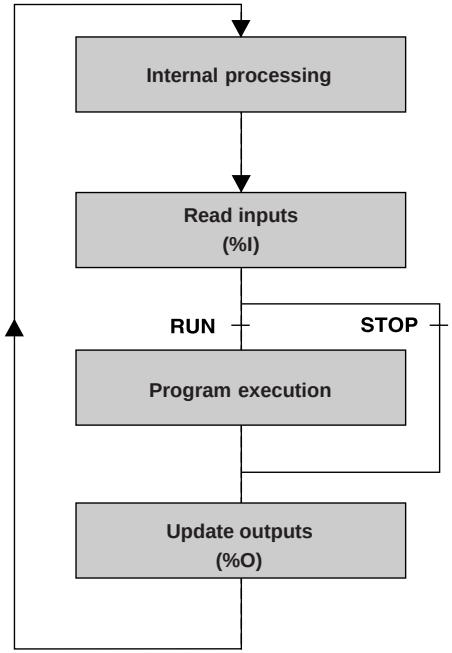
1.3-1 Normal (cyclical) operation

By default, the PLC scan executes cyclically, as follows :

Internal processing :

The system :

- Monitors the PLC :
  - checks the execution capability of the program memory
  - manages the time ; updates the current schedule block (RTC)
  - updates the LEDs : RUN, I/O, ERR, COM
  - detects the RUN/STOP changeover
  - monitors other system parameters
- Processes requests from the programming port and the extension port.



Read inputs :

- Input memory is updated with the status of the physical inputs (%I).

Execute program :

- The program written by the user is then executed.

Update outputs :

- The states of the physical outputs (%Q) are updated from the output memory.

The operating cycle

- PLC running
  The processor manages the system, reads the inputs, executes the program and updates the outputs.
- PLC stopped
  In this case, the processor only manages the system, reads the inputs and updates the output image table.

Warning

Watchdog timer

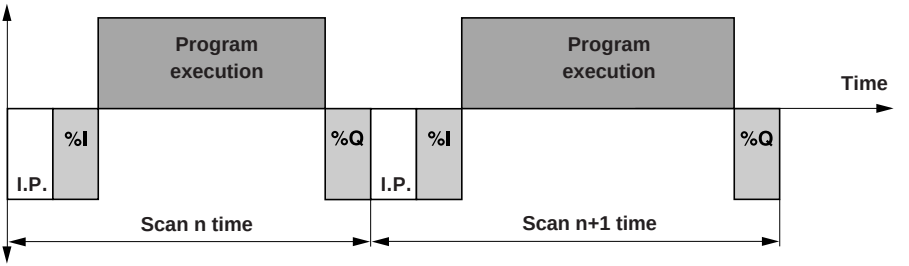
The scan time of the user program is monitored by the PLC watchdog timer and must not exceed 150 ms. Otherwise a fault appears causing the PLC to stop immediately (the RUN and ERR LEDs flash).

The possibilities :

1 Scan time - watchdog time (150 ms)
Normal operation, the next scan is started

2 Scan time > watchdog time
The PLC stops, the RUN and ERR LEDs flash and system bit %S11=1.

Diagram of cyclical scan time



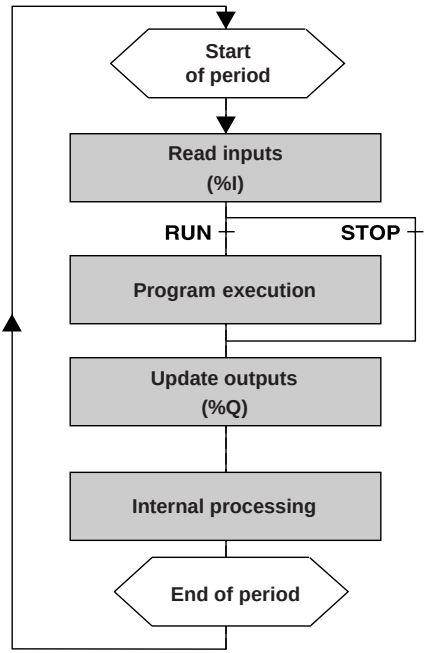
Key :
I.P.= internal processing **%I** = read inputs **%Q** = update outputs

1.3-2 Periodic operation

In this case, reading of inputs, program processing and output updating is executed periodically according to the time defined by the user at configuration (2 to 150 ms).

At the start of the PLC scan, a software timer is set to the value defined during configuration. The PLC scan must finish before the time elapses. The next scan is started when the timer finishes.

When the scan time exceeds the preset time limit, system bit %S19 is set to 1. This bit can be tested and reset by the user or by the user program.



Warning

Watchdog timer

The scan time of the user program is monitored by the PLC watchdog timer and must not exceed 150 ms. Otherwise a fault appears causing the PLC to stop immediately (the RUN and ERR LEDs flash).

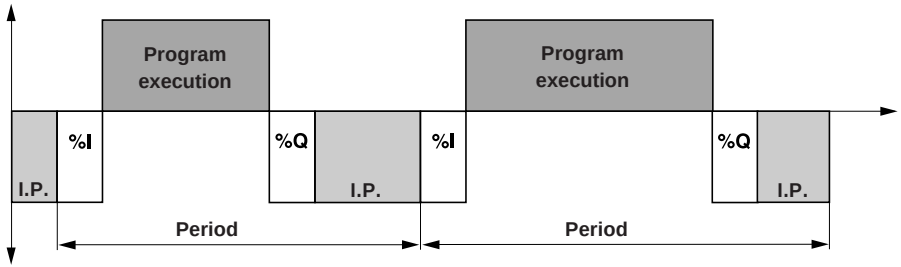
The possibilities :

1 **Scan time - preset time limit**
Normal operation, the next scan is launched as soon as the programmed period has elapsed.

2 **Preset time limit < scan time - watchdog time**
The system bit %S19 is set to 1 by the system and the program user is responsible for resetting it to 0. The PLC remains in RUN.

3 **Scan time > watchdog time**
The PLC stops, the RUN and ERR LEDs flash and system bit %S11=1.

Diagram of periodic execution time

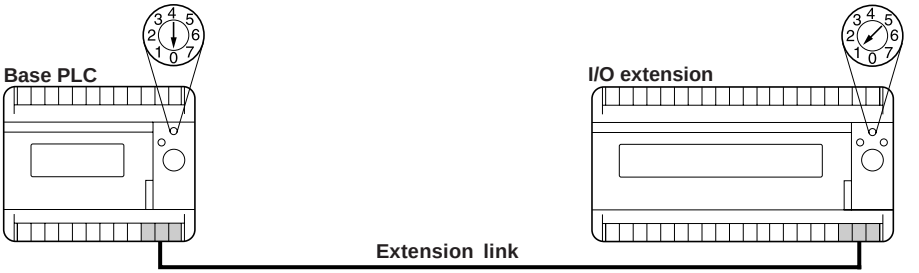


Key :

I.P.= internal processing **%I** = read inputs **%Q** = update outputs

1.4 The I/O extension

"Base PLC" extendable PLCs can be extended using an I/O extension. This extension is created using a "base PLC" extendable PLC (10, 16 or 24 I/O) configured as an I/O extension (10 or 24 I/O).



The configuration of a "base PLC" **extendable PLC** is defined by the position of the selector switch, as shown in the diagram above :

- Selector switch at position 0 = base PLC.
- Selector switch at position 1 = I/O extension.

An I/O extension module does not require any particular configuration since it is dedicated for this use only.

The extension link between the base PLC and the I/O extension should be a shielded twisted pair cable.
The maximum distance between the base PLC and the I/O extension is 200 meters (650 feet).

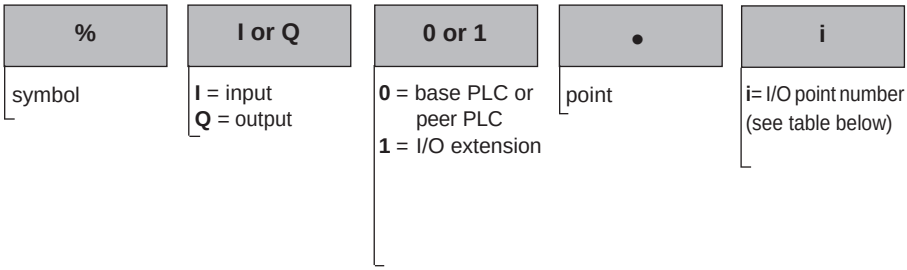
Cable product reference : TSX CA0 003 (length 30 cm - 12 inches).
For distances greater than this, the user should use a cable whose characteristics are defined in Part A, Section 3.5.

Recommended input and output distribution between the base PLC and the I/O extension :

To accommodate any slight variation in update times, critical I/O points should be assigned to the base PLC. This will ensure optimum operation of the control system.

1.5 I/O addressing

An I/O address is defined by the following characters :



Examples

%I0.3 : Input, I/O point n°3 on the base PLC
%Q1.6 : Output, I/O point n°6 on PLC used as I/O extension

Numbers of I/O points for the various PLC types

PLC model	Number of I/O	Inputs	Outputs
TSX 07•• 10••	10 (6I + 4O)	i = 0 to 5	i = 0 to 3
TSX 07 3L 1428	14 (8I + 6O)	i = 0 to 7	i = 0 to 5
TSX 07•• 16••	16 (9I + 7O)	i = 0 to 8	i = 0 to 6
TSX 07 3L 2028	20 (12I + 8O)	i = 0 to 11	i = 0 to 7
TSX 07•• 24••	24 (14I + 10O)	i = 0 to 13	i = 0 to 9

1.6 Special I/O

Introduction

By default, all I/O are configured as discrete I/O. However, certain I/O of a "base PLC" PLC or an extendable PLC configured as a peer PLC can be assigned to specific functions during configuration. An I/O point already used for one function cannot be used for another.

Special inputs

Input functions		Base PLC or Peer PLC inputs %I0.0	%I0.1	%I0.2	%I0.3	%I0.4	%I0.5	%I0.6 to %I0.13
RUN/STOP		•	•	•	•	•	•	_
Latching		•	•	•	•	•	•	_
Inputs linked with fast up/ down counting	Upcounting	•	_	_	_	_	_	_
	Counter set	_	•	_	_	_	_	_
	Fast counter (FC) enable	_	_	•	_	_	_	_
	Downcounting	_	_	_	•	_	_	_
	FC read	_	_	_	_	•	_	_
Analog input		•	_	_	_	_	_	_

Special outputs

Output functions (1)		Base PLC or Peer PLC outputs %Q0.0	%Q0.1	%Q0.2	%Q0.3	%Q0.4	% to %Q0.10
PLC status (SECURITY)		•	•	•	•	_	
PULSE : pulse train		•	_	_	_	_	
PWM : pulse width modulation		•	_	_	_	_	
Analog output		•	_	_	_	_	
Outputs linked with fast up/ down counting	FC OUT 0	_	•	_	_	_	
	FC OUT 1	_	_	•	_	_	

(1) Outputs %Q0.0, %Q0.1, %Q0.2 and %Q0.3, when configured as special outputs, must not be used as outputs in the application program (for example, with Boolean instructions (ST, STN, S, R) with the command bits for drum controller blocks %DRi) or as outputs assigned to the Schedule blocks RTC.

Definition

Use of these I/O is defined in detail in Part A, Section 4. The description below simply defines each I/O function.

- **RUN/STOP input** (see Section 4.1)
  Via an external switch, this input :
  - starts program execution (RUN),
  - stops program execution (STOP).
- **Latching input** (see Section 4.3)
  This input is used to store a pulse which lasts for less than a scan, so that it is taken into account in the following scan.
- **Inputs linked with fast counting** (see Section 4.4)

  Upcounting input
  This input enables the counting of pulses to be taken into account, at a maximum frequency of :
  - 5 or 10 kHz if configured as a fast counter or frequency meter
  - 1 kHz if configured as a fast up/down counter.

  One of the applications is the management of an analog input module (see Section 1.11)

  Set counting input
  This input is used :
  - either to reset the counter to 0 when using it as a fast counter.
  - or to preset it to a value defined at configuration when using it as an up/down counter.

  Fast counter enable input
  This input ensures that signals received at the up/down counting inputs are taken into account.

  Downcounting input
  This input enables downcounting pulses to be taken into account at a maximum frequency of 1kHz.

  Fast counter value read input
  This input enables the current value to be read during the scan, rather than when the scan is completed.
- **PLC status (SECURITY) output** (see Section 4.2)
  Normally set at 1, it changes to 0 on a PLC fault and can therefore be used in external safety circuits.
- **PULSE output** (see Section 4.5)
  Used to generate a signal lasting for a variable period but with a constant duty cycle, or on to off ratio, equal to 50% of the period.
- **PWM output** (see Section 4.6)
  Used to generate a signal lasting for a constant period with the option of varying the duty cycle, or on to off ratio.
  One of the applications is the management of an analog input module (see Section 1.11)
- **Fast Counter threshold outputs** (see Section 4.4)
  Linked to fast counting, these outputs enable data to be read during the scan, without waiting for an update at the end of the scan.

1.7 Additional information about the I/O

1.7-1 Programmable input filters

Input filtering on a "base PLC" PLC or an extendable PLC configured as a peer PLC may be configured from the terminal by altering the filter time. The values which can be configured are as follows :

- 12 ms : filter contact bounce and electrical noise
- 3 ms : filter contact bounce and electrical noise
- No filtering : reading short signals for fast applications but with increased sensitivity to filter contact bounce or electrical noise. In this case the use of volt-free contacts is not advised.

Important

Each type of configurable filter value can be considered as three zones separated by 2 values : the immunity value and the recognition value. Any signal with a duration less than or equal to the immunity value will be rejected. Any signal with a duration greater than or equal to the recognition value will be accepted. Any signal with a duration between these 2 values may accepted or rejected. If no filtering has been configured on an input, the immunity and recognition values are set for inputs %I0.8 to %I0.13 but for inputs %I0.0 to %I0.7 they will depend on whether or not the counting function or the frequency meter function has been configured on input %I0.0.

Filter configured	Immunity	Recognition
12 ms	10 ms	13 ms
3 ms	2 ms	4 ms
No filtering and %FC not configured	0.125 ms for %I0.8 to %I0.13 0.025 ms for %I0.0 to %I0.7	0.375 ms for %I0.8 to %I0.13 0.100 ms for %I0.0 to %I0.7
No filtering and %FC configured as 5 khz counter or 5 khz frequency meter	0.125 ms for %I0.8 to %I0.13 0.025 ms for %I0.0 to %I0.7	0.375 ms for %I0.8 to %I0.13 0.100 ms for %I0.0 to %I0.7
No filtering and %FC configure as 10 khz counter or 10 khz frequency meter	0.125 ms for %I0.8 to %I0.13 0.007 ms for %I0.0 to %I0.7	0.375 ms for %I0.8 to %I0.1. 0.037 ms for %I0.0 to %I0.7

Important

Where no filtering has been configured, and the recognition values are less than the PLC scan time (and thus the scan time of the inputs), to ensure that a signal with a duration greater than the recognition time is processed, use the input which handles this signal to capture the pulses.

• Discrete inputs

By default, all inputs on a "base PLC" PLC or an extendable PLC configured as an I/O extension or peer PLC are configured with a filter time of 12 ms. This time can be modified for each I/O point group on a base or peer PLC.

• Latching inputs

Each of the first 6 inputs (%I 0.0 to %I 0.5) on a "base PLC" PLC or an extendable PLC configured as a peer PLC can be configured individually as latching inputs. This type of operation is used to memorize any pulse with a duration less than the PLC scan time. In this case, the immunity and recognition values also depend on the configuration of the fast counting function %FC.

Inputs %I0.0 to %I0.5 in read mode	**Immunity**	**Recognition**
%FC not configured	0.025 ms	0.100 ms
%FC configured as 5 khz counter or 5 khz frequency meter	0.025 ms	0.100 ms
%FC configured as 10 khz counter or 10 khz frequency meter	0.007 ms	0.037 ms

• Fast counting inputs

If a fast counter, a frequency meter or an up/down counter is declared at the time of configuration, the following inputs are automatically assigned to the counting pulse input.

- %I0.0 for fast counting and frequency meter,
- %I0.0 and %I0.3 for up/down counting.

Used for fast counting or as frequency meter :

Two operating modes can be configured : a 10 khz mode and a 5 khz mode. Input %I0.0 then counts the pulses, which must respect the characteristics of the minimum duration and the minimum separation between pulses to be taken into account.

Mode	**Immunity**	**Min. pulse duration**	**Min. sep. between pulses**
5 Khz	0.025 ms	0.100 ms	0.100 ms
1**0 Khz**	0.004 ms	0.045 ms	0.045 ms

Used for up/down counting:
Inputs %I0.0 and %I0.3 upcount and downcount the pulses, which must respect the characteristics of the minimum duration and the minimum separation between pulses to be taken into account. The maximum frequency is 1 khz.

%I0.0 and %I0.3	Immunity	Min. duration of pulse	Min. sep. between pulses
	0.025 ms	0.100 ms	0.100 ms

Important

• While counter input %I0.0 is at state 1, no downcounting action will be processed on input %I0.3.
• While counter input %I0.3 is at state 1, no upcounting action will be processed on input %I0.0.

1.7-2 Protected transistor outputs on TSX 07 •• •• 12

Protection against overloads and short-circuits

TSX 07 •• •• 12 PLCs have 4, 7 or 10 transistor outputs which are protected against overloads of up to 1A and short-circuits.

0.5A transistor outputs have an electronic device which enables an overload or a short-circuit to be detected on any active output. The appearance of this type of fault causes :

- current limitation (typically 1A) of the affected output,
- tripping of all the outputs in the block (base PLC or I/O extension)
- activation with a steady light of the I/O LED on the base PLC (if base PLC outputs tripped) or LEDs on the base PLC and I/O extension (if I/O extension outputs tripped)
- setting the I/O fault system bit %S10 to 0, and system bits %S118, %S119, %SW118:X0 and %SW119:X0 to change state (see sections B 5.1, B 5.2 and B 6.2).

Reactivating transistor outputs

When a fault has caused tripping of the PLC outputs, they need to be reset. Since tripping of outputs results in downgraded operation of the process being controlled by the PLC, it is advisable to make resetting transistor outputs conditional upon manual intervention. The operator can then, before performing the reset, take any necessary precautions with regard to the control system and personnel, (for example, request changeover to manual mode). The recommended programming is described in section B 6.2.

Note :
If the process being controlled by the PLC is the responsibility of the user, it is possible to program an automatic reset (see section B 6.2).

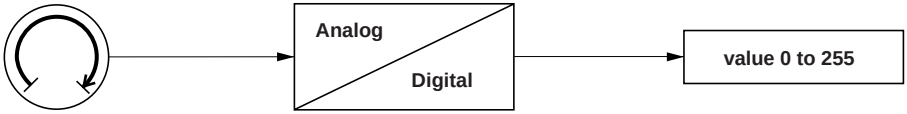
Important

Cold restart of the PLC (set %S0) removes the tripping fault and attempts system reactivation.

1.8 Potentiometers

Principle

An analog/digital converter translates the voltage at the terminals of a potentiometer into a digital value (0 to 255) which is then stored in a word. This value can then, for example, be used as the preset value for a timer that the user can manually adjust without the need for a programming terminal. An example of software start-up is described in Part B, Section 3.2.

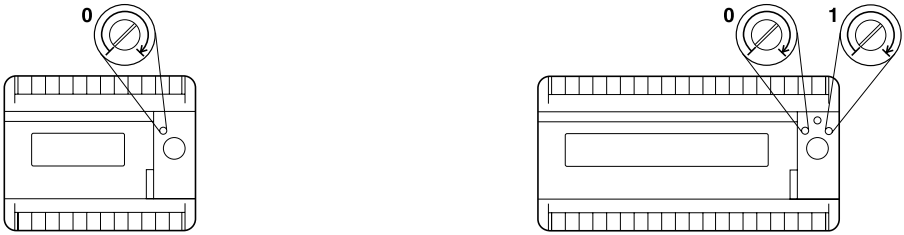


Potentiometers can only be used on "base PLC" PLCs and extendable PLCs configured as peer PLCs.
The number of potentiometers varies depending on the type of PLC :

- 1 on PLCs with 10, 14 and 20 I/O, labelled 0
- 2 on PLCs with 16 and 24 I/O, labelled 0 and 1.

Storage word depending on the potentiometers

Position of the potentiometers



1.9 Status display of the PLC and the I/O

• Displaying the status of the PLC

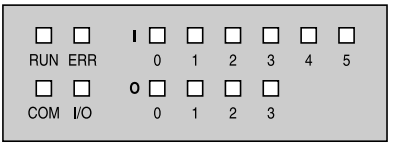
The results of the self-tests performed continually by TSX Nano modules are displayed on the front panel via 4 LEDs : RUN, ERR, I/O and COM.

LED	LED status	Base PLC or peer PLC	I/O extension
RUN	on	PLC in RUN	Same as base PLC
(green)	flashing	PLC in STOP or execution fault	Same as base PLC
	off	Not powered or application not executable	Incorrectly or not connected
ERR	on	Internal faults (watchdog, etc)	Same as base PLC
(red)	flashing	Application not executable	- - - - - - - - - - -
	off	OK	OK
COM	on	Exchanges present on extension link (1)	Same as base PLC
(yellow)	flashing	Exchanges present on Modbus slave (1)	- - - - - - - - - - -
	off	No exchanges on extension link or Modbus	
I/O	on	I/O fault (outputs tripped, sensor supply)	Same as base PLC
(red)	flashing	_	OK
	off	OK	_

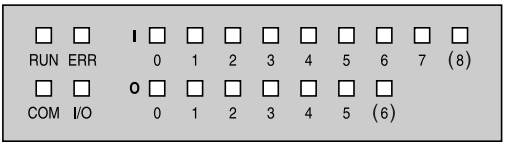
(1) Operation of the I/O or Modbus is exclusive

• Displaying the I/O

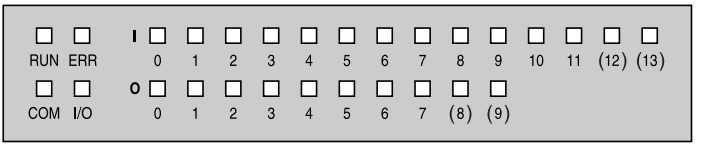
The status of each I/O point is displayed on the front panel of the PLC via a red LED for PLCs version V3.0 or earlier, and a green LED for PLCs version V3.0 or later.



PLC with 10 I/O



PLC with 14/16 I/O



PLC with 20/24 I/O

LED on : I/O on
LED off : I/O off

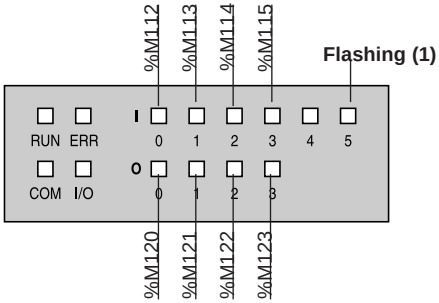
When switched on, all LEDs are lit for approximately 1s.

- **Displaying internal bits**

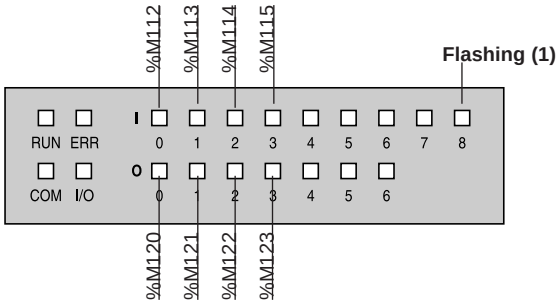
When system bit %S69 is set to 1, TSX Nano PLCs can display the status of 8 or 16 internal bits, rather than displaying the I/O status. The numbering of memory bits is fixed from 112-127.

The corresponding display on the front panel is shown below :

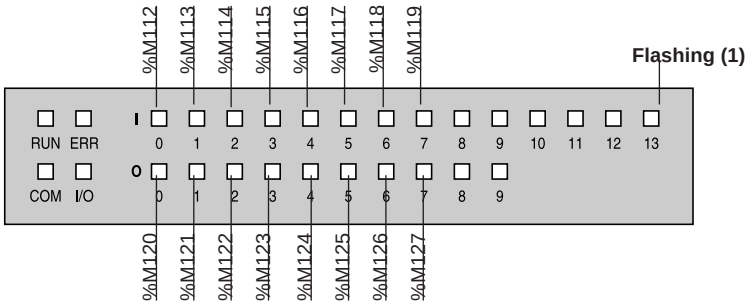
10 I/O PLC



14/16 I/O PLC



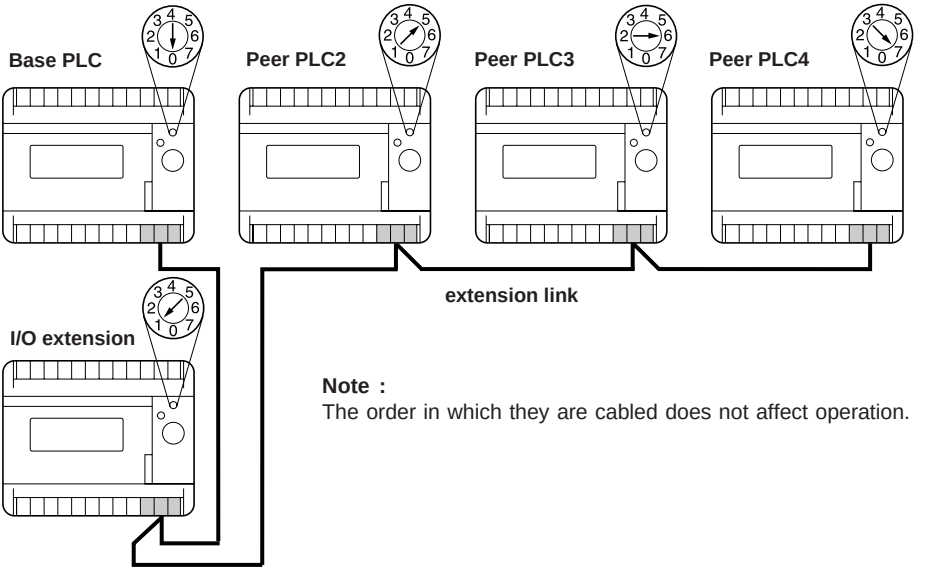
20/24 I/O PLC



(1) series of 5 short flashes, once every second

1.10 Peer PLCs

A maximum of three extendable PLCs configured as peer PLCs which communicate via exchange words (%IW and %QW) can be connected to the "base PLC" extendable PLC. In this case, only the base PLC can have an I/O extension.



The function of each PLC is defined by the position of the selector switch, as shown in the diagram above :

PLC function	Base PLC	I/O Ext.	TSX Micro extension (module TSX STZ10)			Peer PLC2	Peer PLC3	Peer PLC4
Selector position	0	1	2	3	4	5	6	7
Link address	0	1	2	3	4	2	3	4

I/O addressing for peer PLCs 2, 3, and 4 is identical to that for the base PLC (%I0.i and %Q0.i)

The extension link between the base PLC and the extensions (I/O and/or PLCs) should be a shielded twisted pair cable (see Part A, Section 3.5 for the type of cable to use).

The maximum distance between the PLC base and the last peer PLC is 200 meters (650 feet).

Important :

The position of the selector switch is only taken into account when the PLC is powered up.

In order to optimize the communication exchange between the base PLC and the peer PLCs or I/O extension, the number of devices to be scanned and the baud rate on the link must be defined. See Part C, Section 3.2 for information on baud rate configuration.

Overall duration of exchange scan (complete scan)

Number of extensions scanned	**Transmission speed** 9600 bits/s (1)	19200 bits/s
1	17 to 19 ms	6 to 8 ms
2	34 to 35 ms	16 to 18 ms
3	53 to 55 ms	26 to 28 ms
4	72 to 74 ms	35 to 36 ms

Note :
System bit %S72 can be used to prevent any scanning of peer PLCs. It takes priority over selections made during configuration.

Important

Bits X1, X2, X3 and X4 of system word %SW71 enable the status of communication with each peer PLC to be tested on the extension link (at state 1, communication is OK). Using PL7-07 data animation at 19200 bit/s may lead to reduced performance.

The same speed must be configured for all PLCs in the same Nanet network.

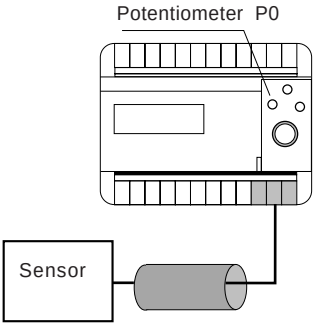
1.11 TSX Nano PLCs with 1 integrated analog input

1.11-1 Presentation

TSX 07 32/33 •• •• PLCs include a non-isolated 0/10V analog input channel. They have the same characteristics and functions as TSX07 30/31 •• •• (see Section 1.2-3). In particular, they allow use of an external analog input or output channel.

The analog input channel replacesthe extension port function in TSX 07 30/31•• ••. TSX 07 32/33 PLCs cannot therefore be connected to other PLCs or to a Modbus slave type link.

The analog input channel is managed via system word %SW112 in the application. A potentiometer P0 is used to correct any errors caused by the analog measurement circuit in some applications.



For details on managing the analog input channel of TSX 07 32/33 •• see Part B Section 3.3.

1.12 Analog modules

1.12-1 Presentation

TSX 07 30/31 •• •• "base PLC" PLCs with discrete I/O (starting from version V3) are able to manage analog I/O modules.

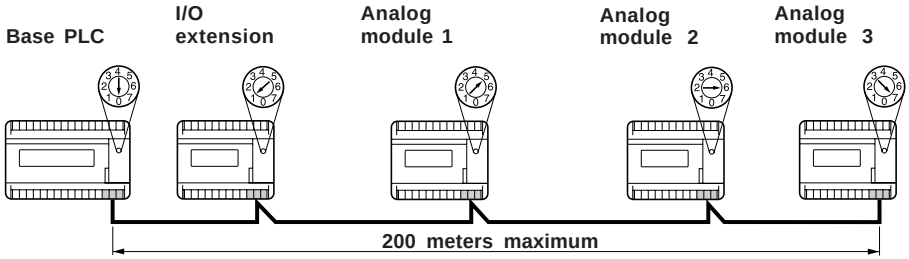
There are three types of analog module in the TSX Nano offer :

- TSX AMN 4000/4001 analog I/O modules managed via the I/O extension link.
- TSX AEN••• modules with one analog input using input %I0.0 of the "base PLC" PLC in frequency meter mode.
- TSX ASN••• modules with one analog output managed via the PWM output (base PLCs must be fitted with transistor outputs).

1.12-2 Presentation of TSX AMN 4000/4001

TSX AMN 4000/4001 analog modules are available in two configurations depending on the power supply required. Each module includes 3 input channels and one output channel.

These modules are managed by the "base PLC" PLC as peer PLCs (maximum of 3). Analog modules and peer PLCs can be mixed on the same link.



Addressing I/O in TSX AMN 400• analog modules is defined by setting the selector switch as shown in the table below.

PLC function / analog module	Base PLC	I/O Ext.	TSX Micro Extension (TSX STZ10 module)			Analog mod. 1	Analog mod. 2	Analog mod. 3
Selector switch position	0	1	2	3	4	5	6	7
Link address	0	1	2	3	4	2	3	4

Important

The position of the selector switch is only taken into account when the PLC is powered up.

Analog modules are configured in the PL7-07 configuration menu.

Exchanges between the "base PLC" PLC and analog modules are managed via exchange words %IW and %QW.

For details on managing the analog I/O modules of TSX 07 AMN 4000/4001 see Part B Section 4.2.

1.12-3 Status display of TSXAMN 4000/4001 modules

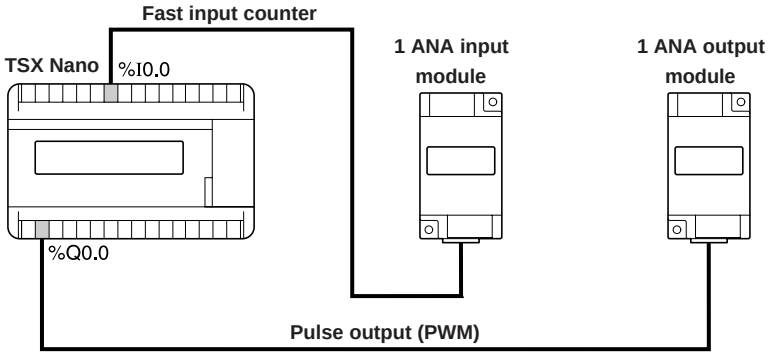
• **Display of the PLC status**

The results of the self-tests performed continually by analog modules are displayed on the front panel via 4 LEDs : RUN, ERR, COM and I/O.

LED	LED status	Meaning
RUN	on	Application in RUN
(green)	off	Not powered or application not executable
ERR	on	Internal faults (autocalibration fault or self-tests not OK)
(red)	flashing	Incorrect module addressing
	off	Functioning OK
COM	on	Communication present
(yellow)	off	No exchanges present
I/O	on	High and low stops on analog inputs exceeded
(red)	off	Functioning OK

1.12-4 Presentation of TSX AEN/ASN •••

Each "base PLC" PLC can support a maximum of one input module and one output module.
The analog input module is managed via the fast counting input on the PLC.
The analog output module is managed via the PWM output on the PLC (base PLCs must be fitted with transistor outputs).



The fast counting input and the PWM output are configured in the PL7-07 configuration menu.

Management of the control of input functions, the value of the analog input read, the control of output functions and the analog output value to be generated are determined by reading and writing system words %SW100 to %SW103 in the application.

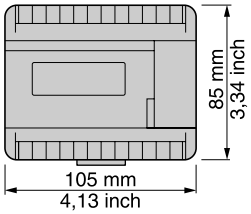
For details on managing the analog I/O modules see Part B Section 4.3 and 4.4.

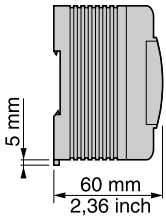
1.12-5 Summary of catalog references

Supply voltage 100/240 V ~	Supply voltage 24 V ⎓	I/O module	Range 0-10V	Range 4-20mA	Range +/-10V	Resolution 8 bits	Resolution 8/12 bits	Resolution 10/12 bits	References
	•	1I	•					•	TSX AEN 101
	•	1I		•				•	TSX AEN 102
	•	1I			•			•	TSX AEN 105
	•	1O	•			•			TSX ASN 101
	•	1O		•		•			TSX ASN 102
	•	1O			•	•			TSX ASN 105
•		3I/1O	•	•	•		•		TSX AMN 4000
	•	3I/1O	•	•	•		•		TSX AMN 4001

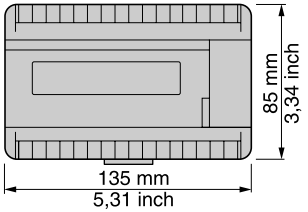
2.1 Dimensions

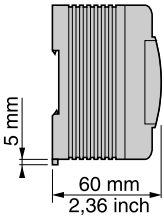
- **TSX 07 •• 10••, TSX 07 32 1028, TSX AMN4000/4001**



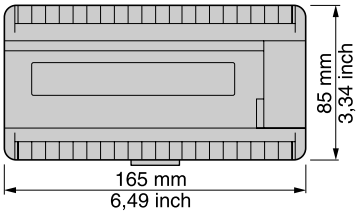


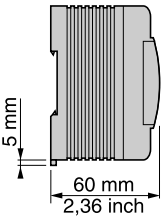
- **TSX 07 •• 16•• except TSX 07 •1 1648, TSX 07 3L 1428**



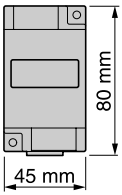


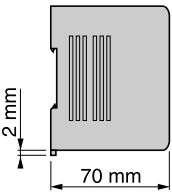
- **TSX 07 •• 24•• with 16 I/O : TSX 07 •1 1648, TSX 07 2028**





- **TSX AEN 10• and TSX ASN 10•**

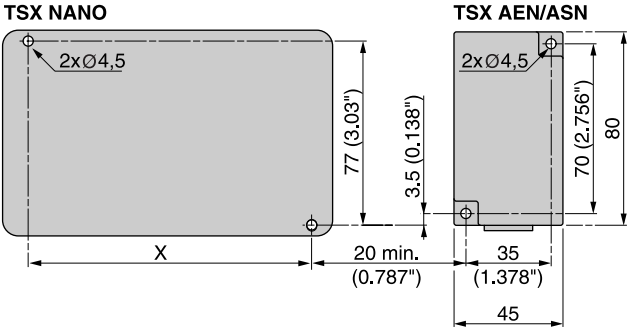




2.2 Mounting

TSX Nano PLCs and analog modules can be mounted on :

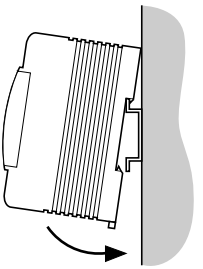
- a mounting plate or panel using 2 Ø M3 screws (not supplied)
- a 35 mm DIN mounting rail.
- **Mounting on a mounting plate or panel using screws**



PLCs / Analog modules	TSX 07 ••10•• TSX 07 32 1028 TSX AMN 400•	TSX 07 •• 16•• TSX 07 3L 1428	TSX 07 •1 1648	TSX 07 ••24•• TSX 07 2028
X	86 mm 3.38 inches	116 mm 4.56 inches	146 mm 5.74 inches	146 mm 5.74 inches

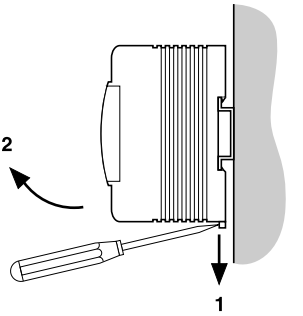
- **Mounting on a 35 mm DIN rail**

Mounting



Position the PLC and clip it on to the DIN rail as shown in the above diagram.

Removal



1 With a screwdriver, release the latch at the back which locks the PLC onto the DIN rail.
2 Keeping the latch open, pivot the PLC as shown in the above diagram.

Warning

When mounting PLCs and analog modules on a DIN rail, use 2 end stops, type AB1-AB8P35 or equivalent.

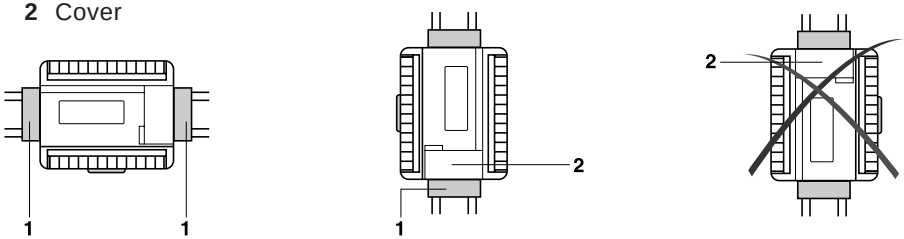
2.3 Installation rules

TSX Nano PLCs and analog modules should be mounted on a vertical plane and at the minimum distances shown on the figure below, in order to maintain a natural circulation of air.

- **Mounting positions**

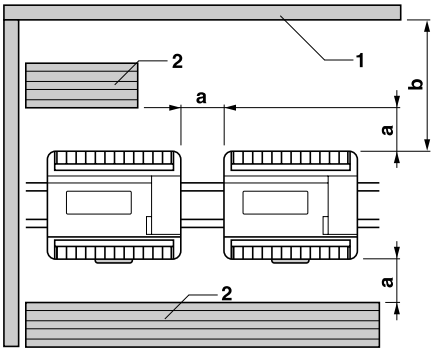
  - Vertical plane

1 End stops AB1-AB8P35
2 Cover



  - Horizontal plane : Do not mount on a horizontal plane.

- **Minimum clearances to be observed**



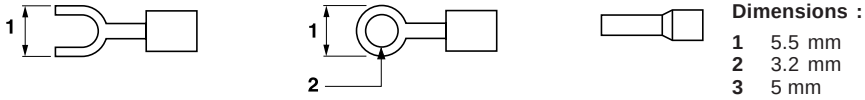
1 Switchgear, enclosure or machine frame.
2 Cable ducting or clips.
a ≥ 20 mm
b ≥ 40 mm

Avoid mounting heat generating devices (transformers, power supplies, contactors, etc) beneath the PLCs.

3.1 I/O wiring rules and precautions

3.1-1 General precautions and rules

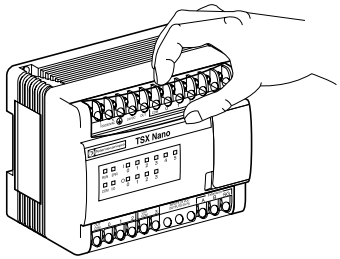
The connection terminals on TSX Nano PLCs are protected by a removable cover which protects users when the PLC is powered up. Each terminal can take up to two 1 mm^2 (16 AWG) wires fitted with open or closed cable ends. **Coupling torque of terminal screws : max. 0.5 Nm.**



Dimensions :
1 5.5 mm
2 3.2 mm
3 5 mm

When using closed cable ends, it is necessary to remove the cover in order to take out the screw needed for fixing the cable ends.

Removing the cover



TSX Nano PLCs have integral protective devices on the I/O, which give them a good resistance to industrial environments. However, certain rules must be observed so as to maintain this immunity.

- **Discrete inputs**
  All multi-core cables carrying sensor data must include the sensor common.
- **Discrete outputs**
  Relay outputs : the following must be connected in parallel with the terminals of the output :
  - An RC or MOV (ZNO) protection circuit for AC.
  - A freewheel diode for DC.
- **Cable routing**
  - External to the device
    All I/O wiring must be routed in cable ducting which is separate from those carrying high voltage cables, and at least 100 mm (4 inches) from parallel power cables.
  - Inside the device
    Power cables (power supply, contactors, etc) must be kept separate from input (sensors) and output wiring.
    If possible, I/O wiring should be routed in separate cable ducting.

3.1-2 Special precautions for connecting discrete low noise immunity inputs

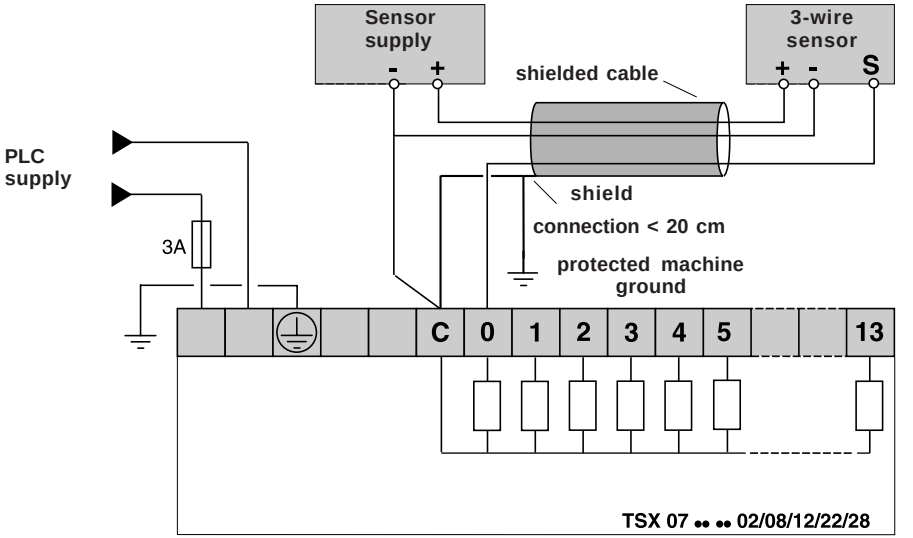
Some inputs can be configured as :

- Fast counters or frequency meter inputs, with a frequency of 5 or 10 kHz (input %I0.0).
- Up/down counter inputs, 1 kHz (inputs : upcounter %I0.0, downcounter %I0.3).
- Latching inputs (inputs %I0.0 to %I0.5).
- Discrete inputs without filter (%I0.0 to %I0.3 and %I0.4 to %I0.7).

The noise immunity of the inputs which have been configured in this way is reduced, and certain connection precautions must be taken.

- **Input %I0.0 used as a fast counter or frequency meter input (5 or 10 kHz), 1 kHz counter input, latching input or discrete input without filter :**
  A shielded cable must be used with :
  - the cable shield connected to terminal C (COM), the input common (to the power supply - with positive logic (sinking) inputs or to the power supply + with negative logic (sourcing) inputs),
  - terminal C (COM) connected to the machine protected ground.

  Example : Wiring diagram for connecting input %I0.0 to a 3-wire sensor wired for positive logic

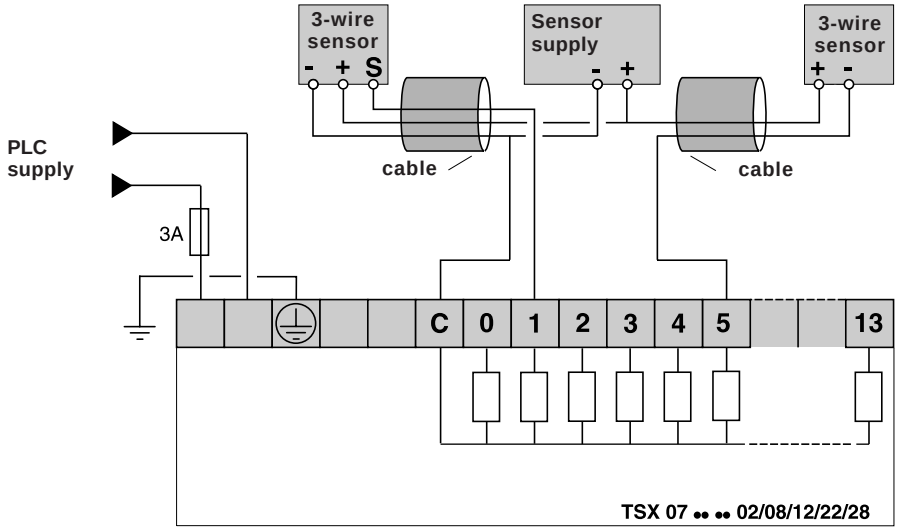


Note :
When using the 1 kHz up/down counting function, input %I0.3, which is used as the downcounting input, will be connected following the same principle as for input %I0.0

- **Other inputs used as latching inputs (%I0.1 to %I0.5) or discrete inputs without filter (%I0.1 to I0.7) :**

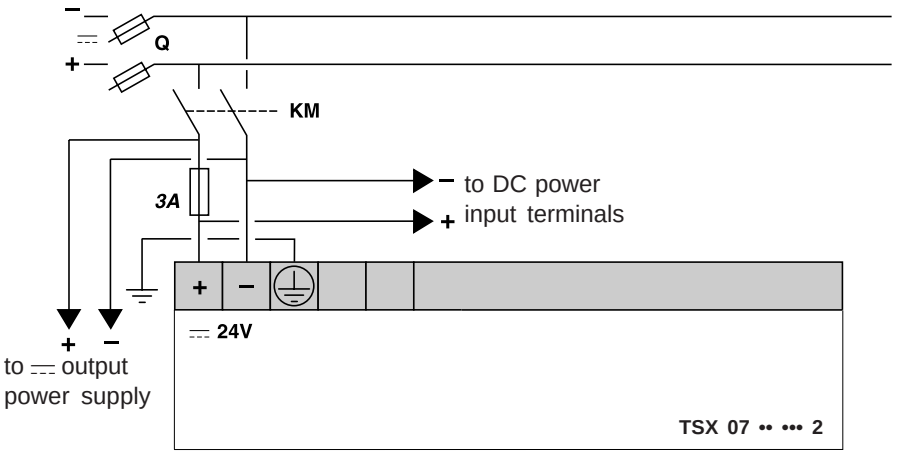
  The general principle is to group together on the same cable all the wires relating to the sensor which is controlling the input. Thus one cable is used per I/O point.

  Example : Wiring diagram with inputs wired for positive logic (sinking)

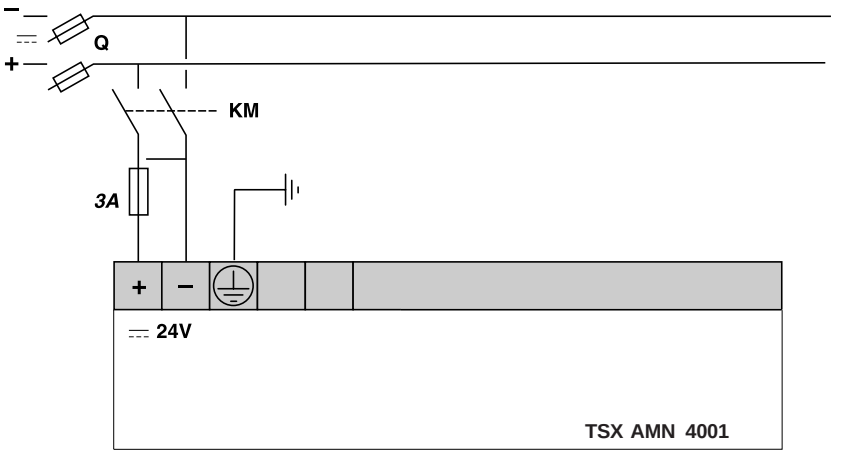


3.2 Connection of power supply

- **24 VDC power supply**



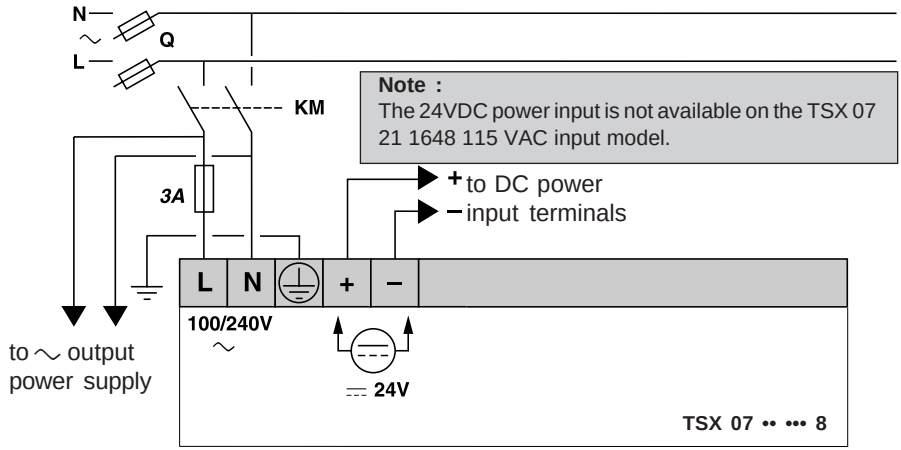
Q Master switch
KM Line contactor or circuit breaker (not essential on small installations)



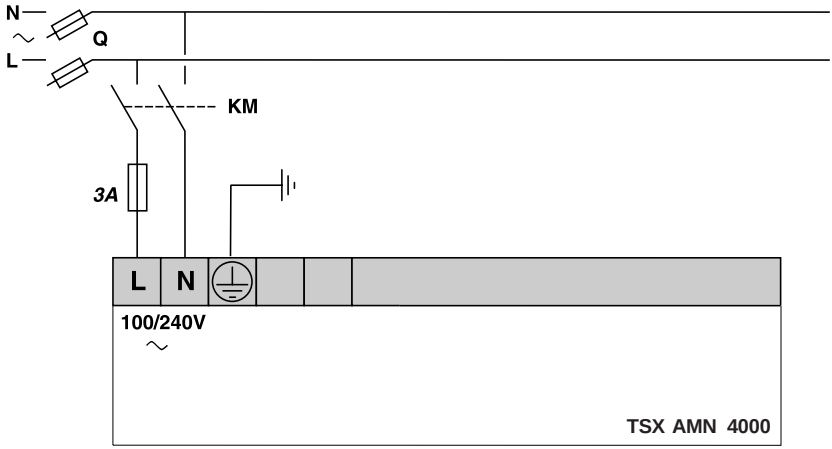
Q Master switch
KM Line contactor or circuit breaker (not essential on small installations)

Key : ∼ AC, ⎓ DC

- **100 to 240 VAC power supply**



Q Master switch
KM Line contactor or circuit breaker (not essential on small installations)



Q Master switch
KM Line contactor or circuit breaker (not essential on small installations)

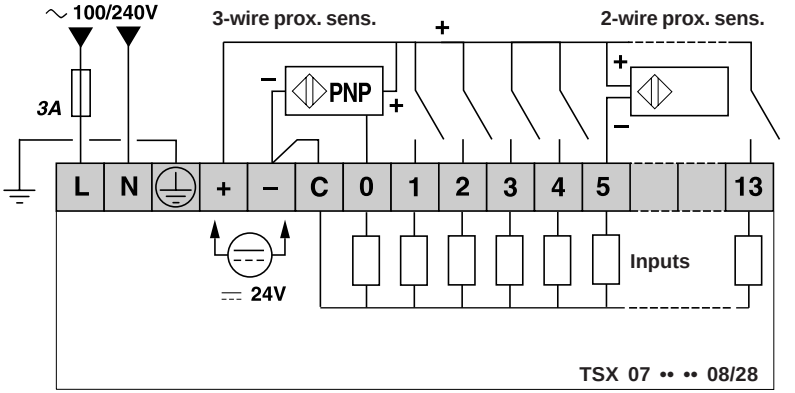
Key : ~ AC, ⎓ DC

3.3 Discrete input connection

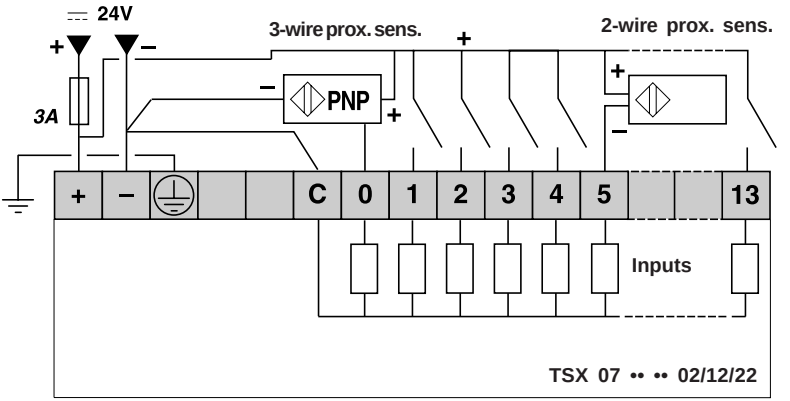
3.3-1 24V input connections

- **Connection of the positive logic (sinking) inputs**

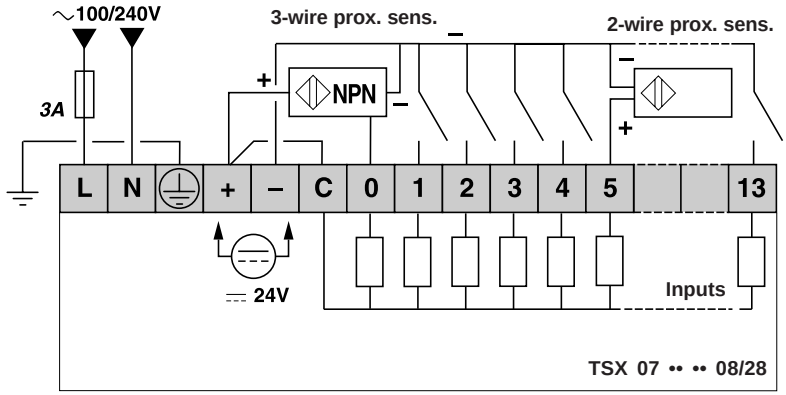
  On PLCs with AC supply :
  - 10 I/O : TSX 07 •0 1008, TSX 07 •• 1028
  - 14 I/O : TSX 07 3L 1428
  - 16 I/O : TSX 07 •1 1608, TSX 07 •• 1628
  - 20 I/O : TSX 07 3L 2028
  - 24 I/O : TSX 07 •1 2408, TSX 07 •• 2428



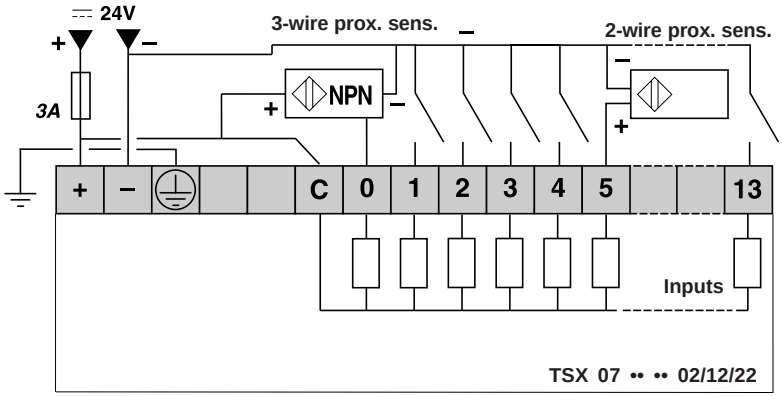
On PLCs with DC supply :
- 10 I/O : TSX 07 •0 1002, TSX 07 •0 1012, TSX 07 •0 1022
- 16 I/O : TSX 07 •1 1602, TSX 07 07 •• 1612, TSX 07 •1 1622
- 24 I/O : TSX 07 •1 2402, TSX 07 07 •• 2412, TSX 07 •1 2422



- **Connection of the negative logic (sourcing) inputs**
  On PLCs with AC supply :
  - 10 I/O : TSX 07 •0 1008, TSX 07 •• 1028
  - 14 I/O : TSX 07 3L 1428
  - 16 I/O : TSX 07 •1 1608, TSX 07 •• 1628
  - 20 I/O : TSX 07 3L 2028
  - 24 I/O : TSX 07 •1 2408, TSX 07 •• 2428

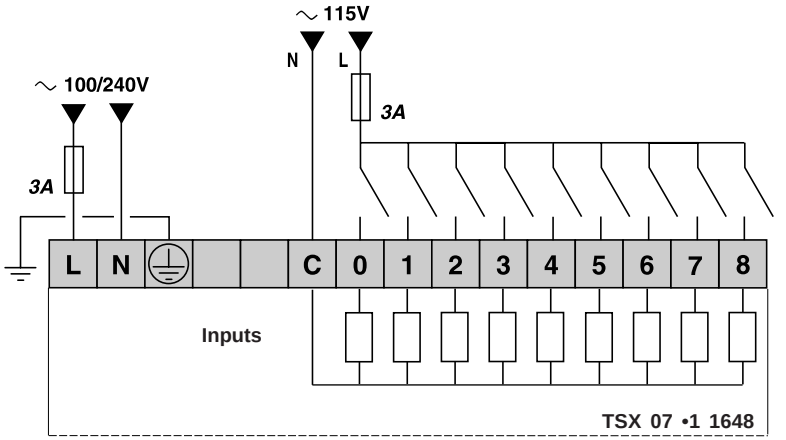


On PLCs with DC supply :
- 10 I/O : TSX 07 •0 1002, TSX 07 •0 1012, TSX 07 •0 1022
- 16 I/O : TSX 07 •1 1602, TSX 07 •• 1612, TSX 07 •1 1622
- 24 I/O : TSX 07 •1 2402, TSX 07 •• 2412, TSX 07 •1 2422



3.3-2 Connection of 115 VAC inputs

- PLC : TSX 07 •1 1648

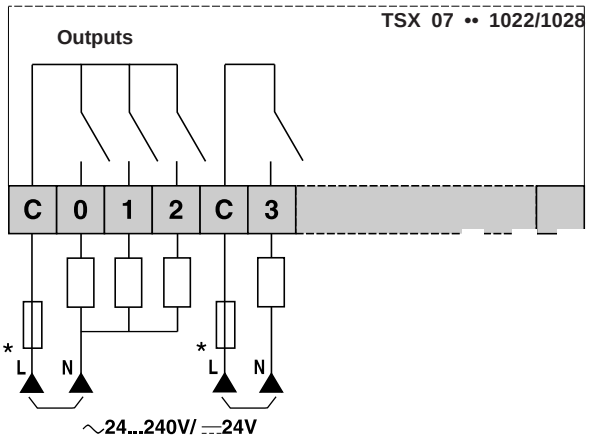


3.4 Connection of discrete outputs

3.4-1 Connection of relay outputs

(except TSX 07 •1 1648)

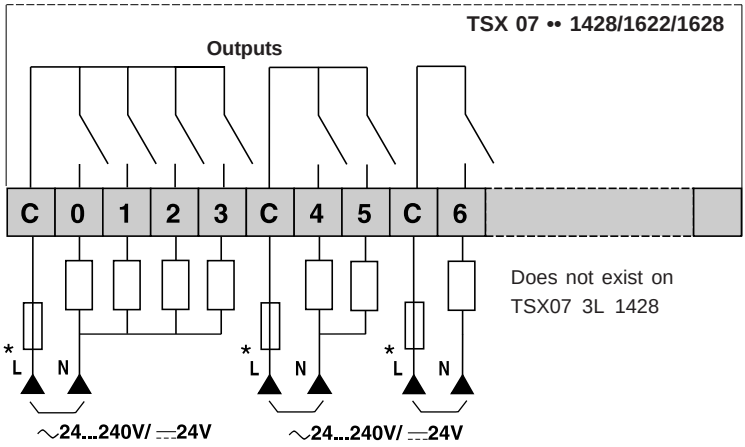
- PLCs with 10 I/O : TSX 07 •0 1022, TSX 07 •• 1028



* : fuse rated for load

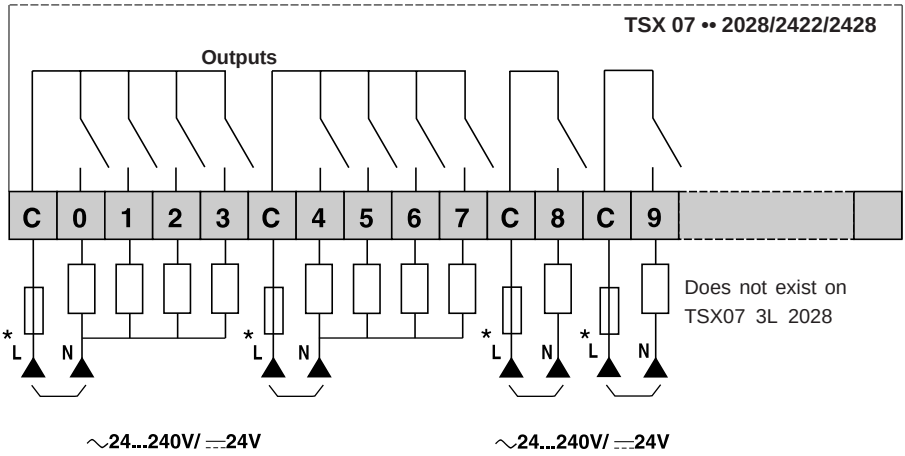
Connection of relay outputs (continued)

- PLCs with 14 I/O : TSX 07 3L 1428
  16 I/O : TSX 07 •1 1622, TSX 07 •• 1628



* : fuse rated for load

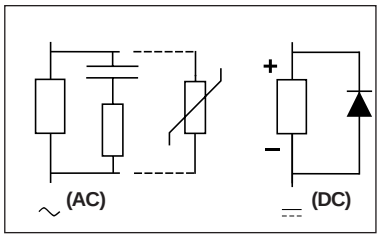
- PLCs with 20 I/O : TSX 07 3L 2028
  24 I/O : TSX 07 •1 2422, TSX 07 •• 2428



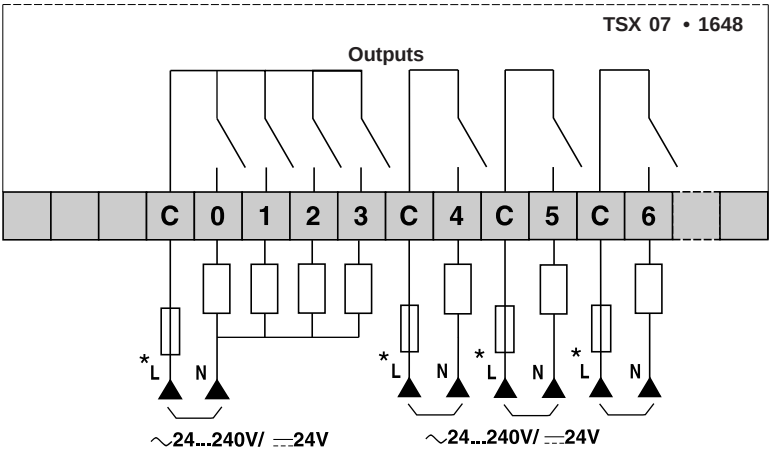
* : fuse rated for load

Essential protection for each output terminal

- An RC circuit or MOV (ZNO) suppressor circuit for AC
- A freewheel diode for DC



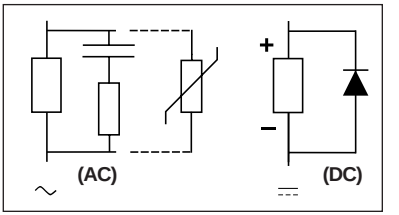
3.4-2 Connection of relay outputs on TSX 07 •1 1648 PLC



* : fuse rated for load

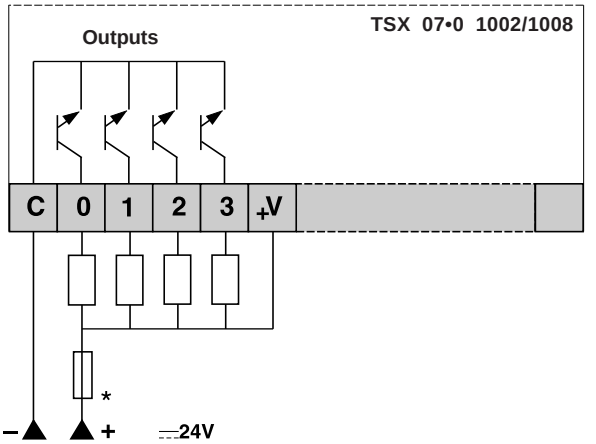
Essential protection for each output terminal

- An RC circuit or MOV (ZNO) suppressor for AC
- A freewheel diode for DC



3.4-3 Connection of negative logic (sinking) transistor outputs

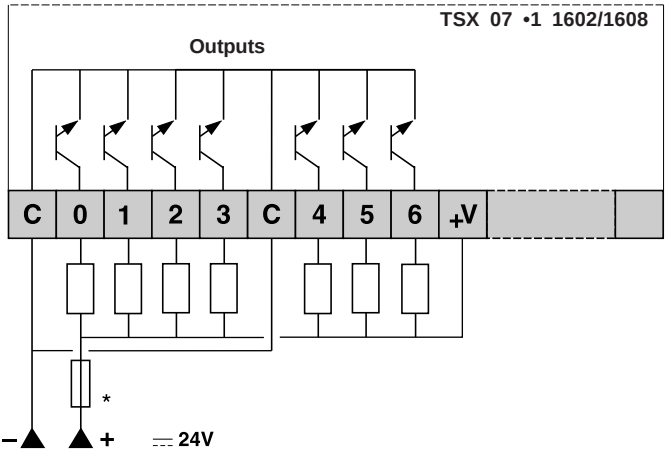
- PLCs with 10 I/O : TSX 07 •0 1002 and TSX 07 •0 1008



* : fuse rated for load

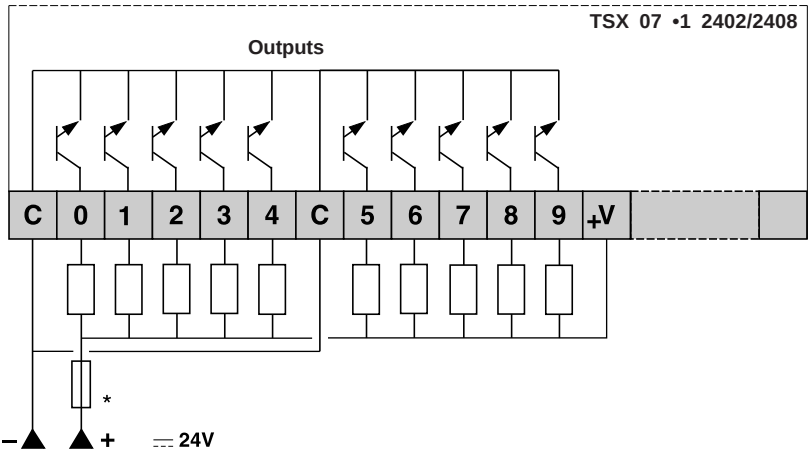
Connection of negative logic (sinking) transistor outputs (continued)

- PLCs with 16 I/O : TSX 07 •1 1602 and TSX 07 •1 1608



* : fuse rated for load

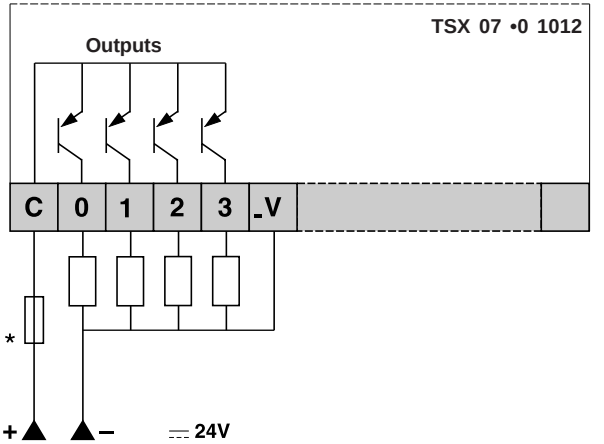
- PLCs with 24 I/O : TSX 07 •1 2402 and TSX 07 •1 2408



* : fuse rated for load

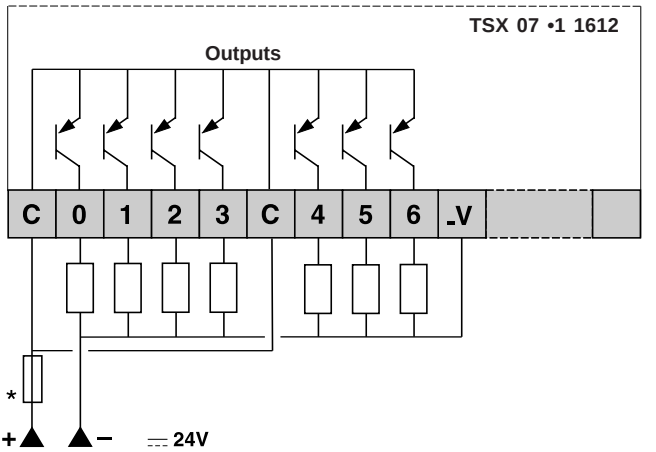
3.4-4 Connection of positive logic (sourcing) transistor outputs

- PLC with 10 I/O : TSX 07 •0 1012



* : fuse rated for load

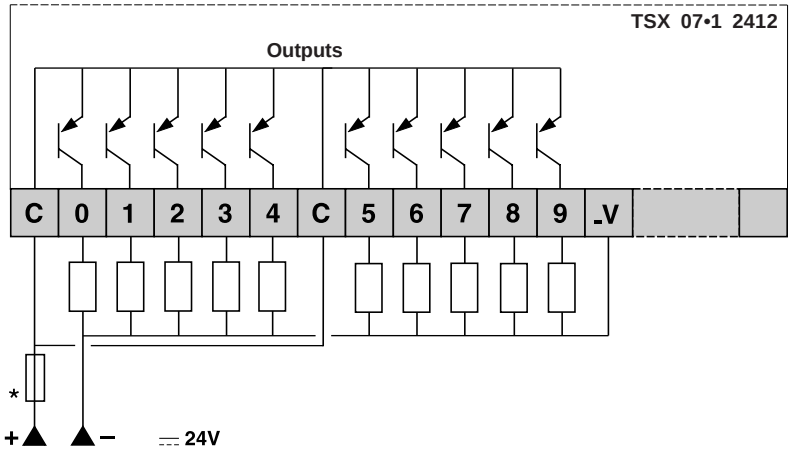
- PLC with 16 I/O : TSX 07 •• 1612



* : fuse rated for load

Connecting positive logic transistor (sourcing) outputs (continued)

- PLC with 24 I/O : TSX07 •• 2412



* : fuse rated for load

3.5 I/O extension connections

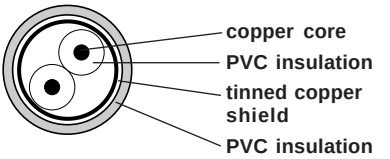
The I/O extension is connected to the "base PLC" PLC via a shielded twisted pair cable :

- Cable length 30 cm (12 inches) : reference TSX CA 0003.
- If a longer cable is required, use :
  - either a UNI-TELWAY shielded twisted double pair cable :
    TSX STC 50 : length 50 meters (165 feet), or TSX STC 200 : length 200 meters (650 feet)
  - or a **shielded twisted pair** cable, the main characteristics of which are defined below :

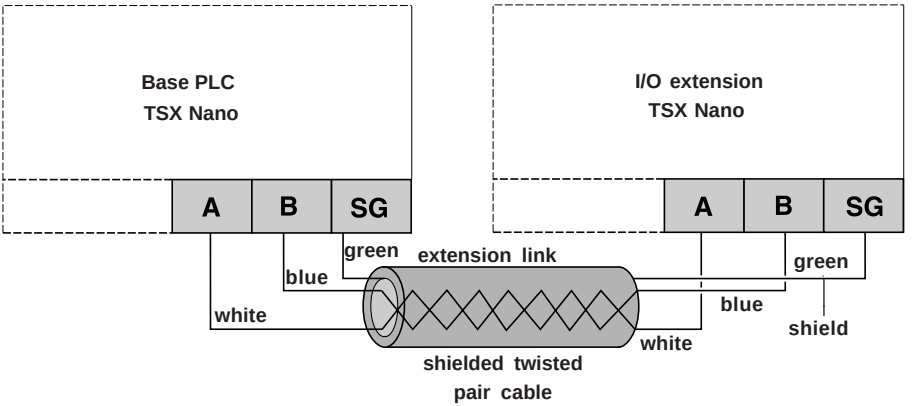
Mechanical characteristics :
Tinned copper core, 18 to 24 gauge, tinned copper shield

Electrical characteristics :
Load resistance per unit length of one wire : - 85 Ω / Km
Load resistance per unit length of shield : - 12 Ω / Km

Structure



The maximum authorized distance between the base PLC and the I/O extension is 200 meters (650 feet).



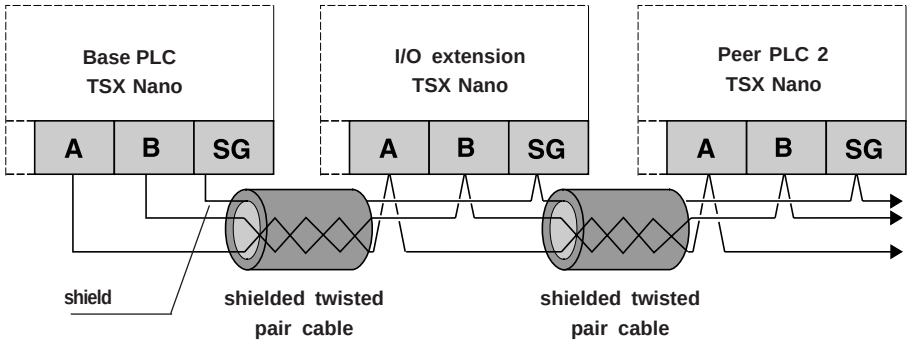
Note :
Colors shown refer to connection using cable TSX CA 0003

3.6 Connection of peer PLCs

Peer PLCs are connected to the "base PLC" PLC in the same way as I/O extensions, that is via a shielded twisted pair cable (for the cable to use see the previous page).

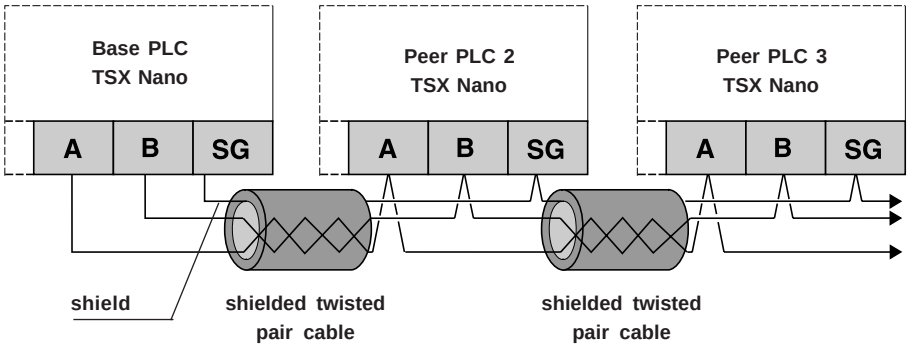
The maximum permitted distance between the base PLC and the last peer PLC is 200 meters (650 feet).

- Connection of an I/O extension to the "base PLC" PLC, and a peer PLC.



Note :
If peer PLCs 3 and 4 are used, the continuity of the extension link is provided by a shielded twisted pair cable with identical connections to those between the I/O extension and peer PLC 2.

- Connection of peer PLCs only to the "base PLC" PLC.



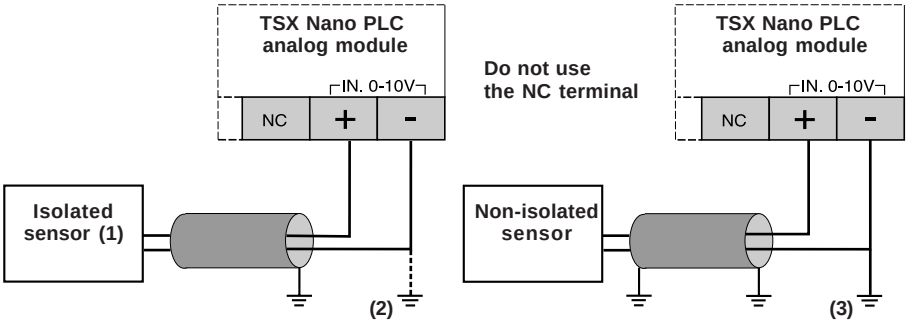
Note :
If peer PLC 4 is being used, the continuity of the extension link is provided by a shielded twisted pair cable with identical connections to those between the I/O extension and peer PLCs 2 and 3.

3.7 Connection of an analog sensor (TSX 07 32/33 ••28)

The analog sensor is connected to the analog input channel of the PLC via a shielded twisted pair cable (cable identical to the one used to connect an I/O extension, see Section 3.5).

The maximum permissible distance between the sensor and the analog channel of the PLC is :

- isolated sensor ≤ 30 m (99 feet) with shielded cable,
- non-isolated sensor ≤ 10 m (33 feet) with shielded cable.



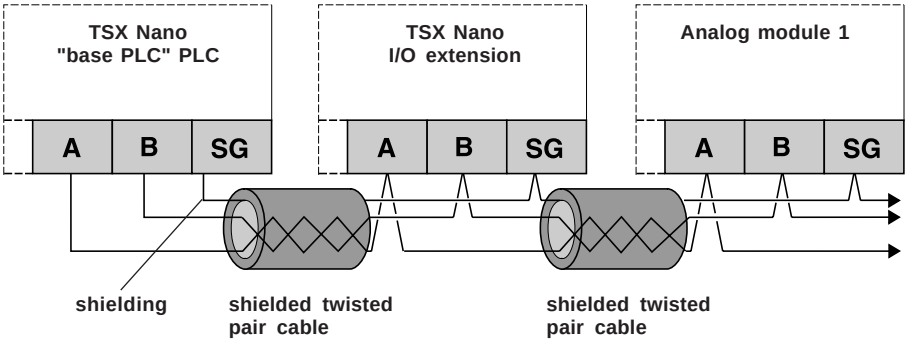
(1) Floating with respect to ground.
(2) Grounding the - of the analog input is recommended.
(3) If a non-isolated sensor is used, the - of the analog input must be grounded.

3.8 Connection of analog modules (TSX AMN 4000/4001)

3.8-1 Connection of analog modules to the "base PLC" PLC

Like discrete I/O extensions, TSX AMN 400• analog modules are connected to the PLC via a shielded twisted pair cable (cable identical to the one used to connect an I/O extension, see Section 3.5).

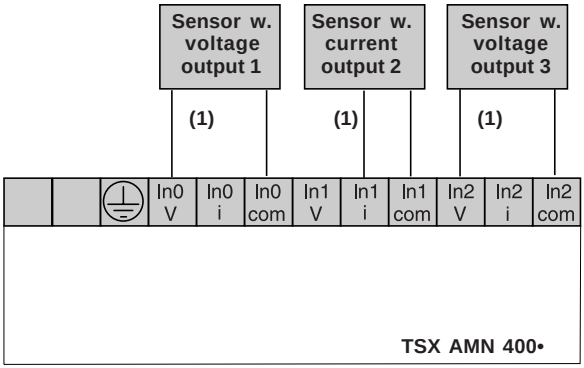
The maximum permissible distance between the "base PLC" PLC and the last analog module is 200 meters (650 feet).



Note :
If analog modules 2 and 3 are used, continuity of the extension link is provided by a shielded twisted pair cable with connections identical to those made between the I/O extension and analog module 1.

3.8-2 Connection of analog inputs

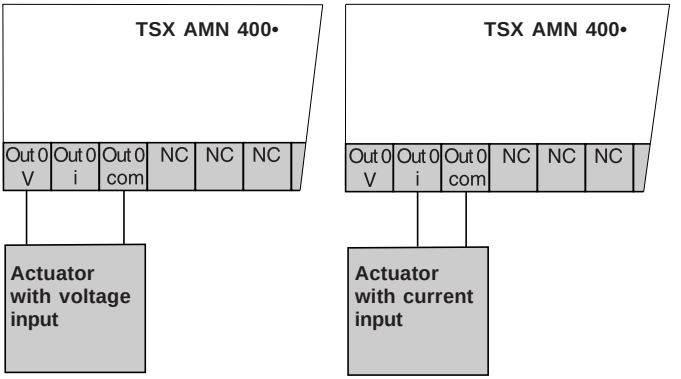
The analog input channels of the TSX AMN 400• module can take sensors with either a voltage output or a current output.
The example below illustrates one type of configuration.



(1) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

3.8-3 Connection of analog outputs

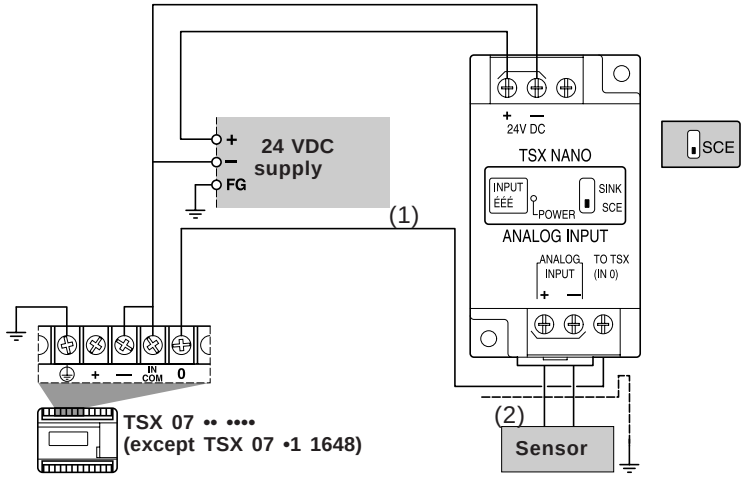
The analog output channel of the TSX AMN 400• module allows connection to an actuator with either a voltage input or a current input.



(1) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

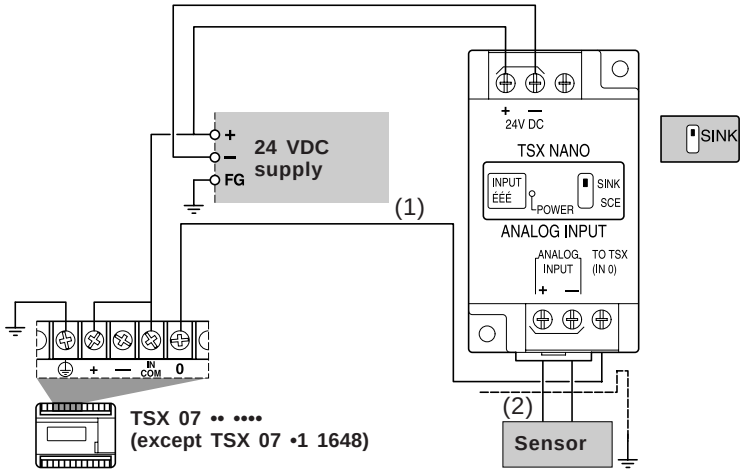
3.9 Connection of analog inputs (TSX AEN •••)

3.9-1 Connection with input 0 of the TSX Nano wired as sink (positive logic)



(1) Maximum 2 meters with Ø 0.5 mm shielded cable
(2) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

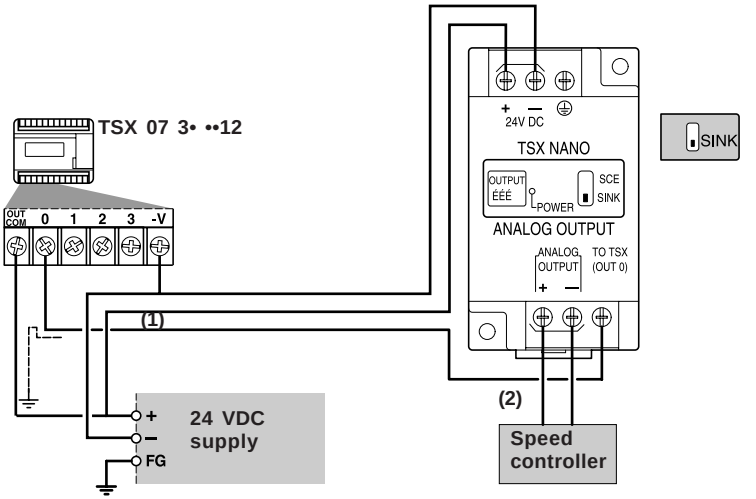
3.9-2 Connection with input 0 of the TSX Nano wired as source (negative logic)



(1) Maximum 2 meters with Ø 0.5 mm shielded cable
(2) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

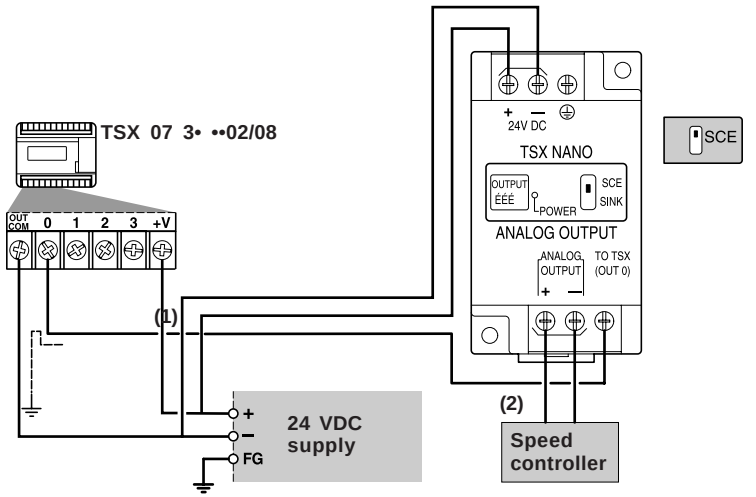
3.10 Connection of analog outputs (TSX ASN •••)

3.10-1 Connection with source output 0 of the TSX Nano (positive logic)



(1) Maximum 2 meters with Ø 0.5 mm shielded cable
(2) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

3.10-2 Connection with sink output 0 of the TSX Nano (negative logic)



(1) Maximum 2 meters with Ø 0.5 mm shielded cable
(2) Maximum 50 meters with Ø 0.5 mm shielded cable (shielding connected at module end)

4.1 RUN/STOP input

Principle

The RUN/STOP input is used to start or stop program execution.

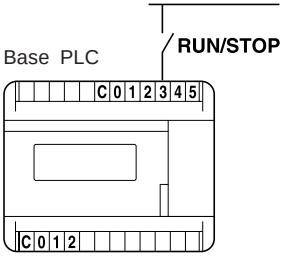
On a base or peer PLC, one of the first 6 inputs (%I0.0 to %I0.5) may, after configuration via the terminal, be assigned to this function.

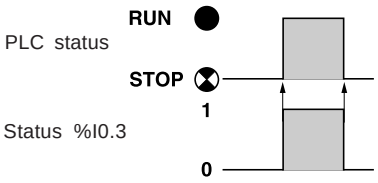
When powering up the PLC, the PLC state is determined by the RUN/STOP input (if configured) :

- PLC in STOP if the configured RUN/STOP input is at state 0,
- PLC in RUN if the configured RUN/STOP input is at state 1.

While the PLC is powered, a rising edge on the RUN/STOP input sets the PLC to RUN. The PLC is forced to STOP if the RUN/STOP input is at 0. If the RUN/STOP input is at state 0, a RUN command from a connected terminal will be ignored by the PLC.

Example : RUN/STOP switch on input %I0.3





4.2 PLC status (SECURITY) output

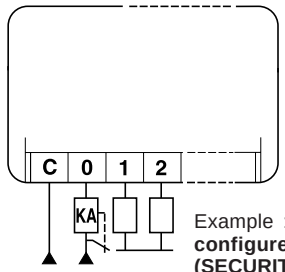
Principle

When the PLC is powered up and if no PLC error (blocking fault see appendix A.6) is detected, the PLC status (security) output changes to 1. It can be used in safety circuits external to the PLC, for example to control :

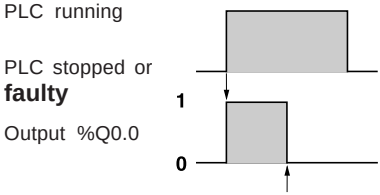
- The power supply to output devices.
- The PLC power supply.

On a base or peer PLC, one of the first 4 outputs (%Q0.0 to %Q0.3) may, after configuration via the terminal, be assigned to the PLC status (SECURITY) function.

Base PLC



Example : **output %Q0.0 configured as PLC status (SECURITY) output**

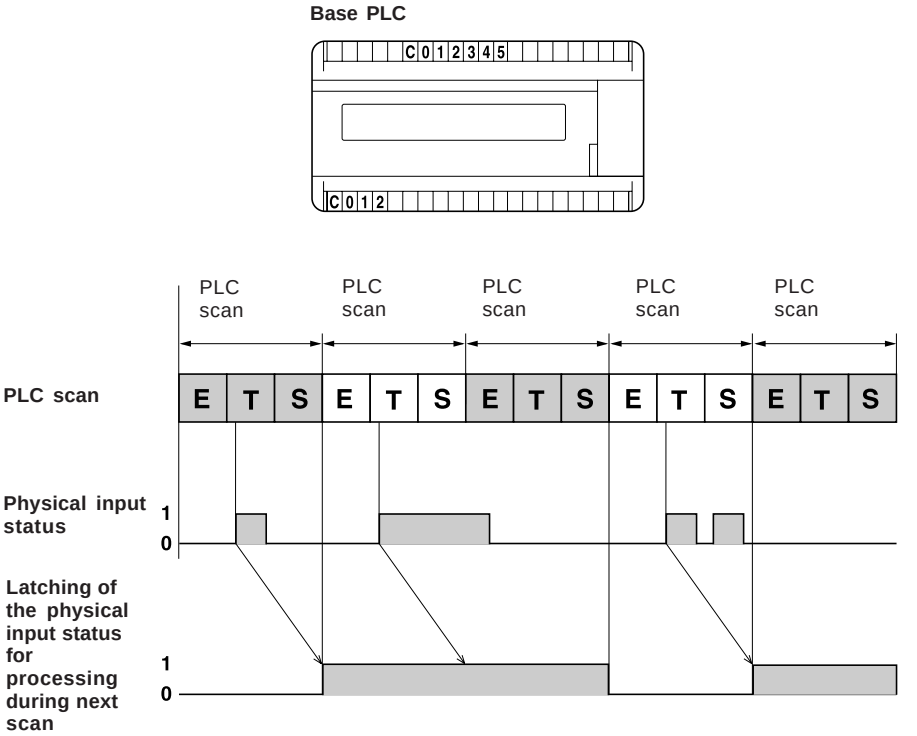


Appearance of a fault which stops the PLC

4.3 Latching inputs

Principle

When a pulse is shorter than one scan and has a value greater than or equal to 100 µs (1), the PLC latches the pulse, which is then updated in the next scan.
On a base PLC or peer PLC, each of the first 6 inputs (%I0.0 to %I0.5) can be assigned to the special latching function, after configuration via the terminal.



Key:
- I : read inputs
- P : process program
- O : update outputs.

Note :
A pulse which lasts longer than one scan will be treated as a pulse received at a standard input.

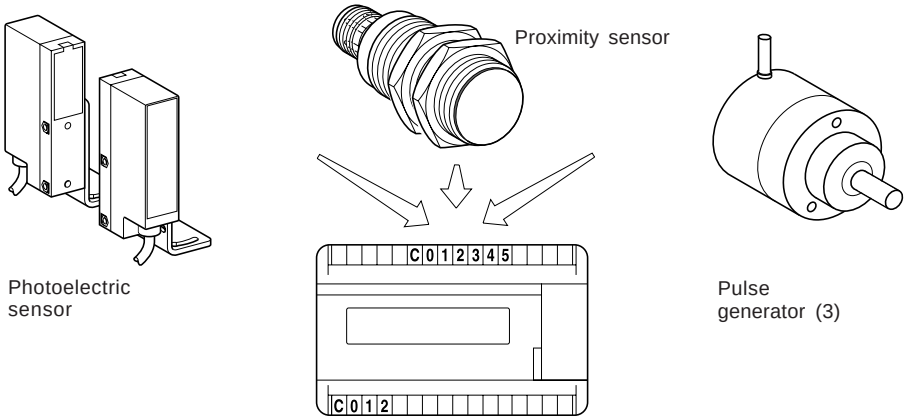
(1) see Part A, Section 1.7-1

4.4 I/O associated with fast counting

The TSX Nano PLC has a fast counter which can be used in three different ways :

- As a fast counter (maximum frequency 10 kHz).
- As a frequency meter (maximum frequency 10 kHz) which can, in particular, be used to manage an analog input module.
- As a fast up/down counter (maximum frequency 1 kHz).

Sensors with transistor outputs must be used with up/down counter inputs %I0.0 and %I0.3. NOTE : Do not use sensors with contact outputs with fast counter inputs, as this kind of sensor is susceptible to bounce because of low noise immunity.



Photoelectric sensor

Pulse generator (3)

Base PLC

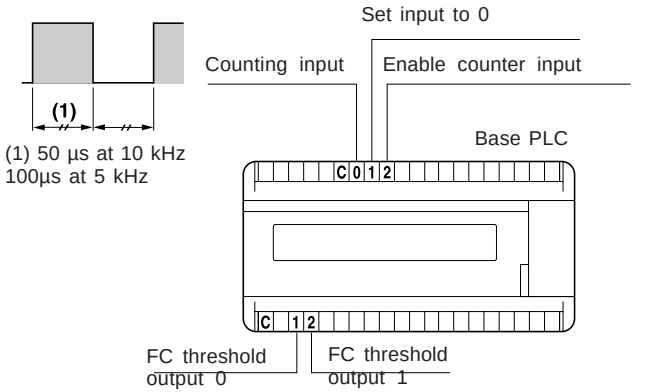
Functions	**Inputs**					**Outputs**	
	%I0.0	**%I0.1**	**%I0.2**	**%I0.3**	**%I0.4**	**%Q0.1**	**%Q0.2**
Upcounter	•	_	_	_	_	_	_
Counter set (reset counter to 0)	_	• (1)	_	_	_	_	_
Enable	_	_	• (1)	_	_	_	_
Downcounter		_	_	_	•	_	_
_							
Read the current value (2)	_	_	_	_	• (1)	_	_
Threshold output 0 (HSC_Out)	_	_	_	_	_	• (1)	_
Threshold output 1 (HSC_Out)	_	_	_	_	_	_	• (1)

The parameters of the function to be performed (fast counter, frequency meter, up/down counter) are set using a special %FC function block (see Part B, Section 3.4-5).

(1) These I/O are optional. Their use must be declared during configuration.
(2) Available only for the up/down configuration of the fast counter.

4.4-1 Use as a fast counter

The fast counter (FC) function enables counting at a maximum frequency of 10 kHz or 5 kHz depending on the selection made at configuration, with the possibility of counting from 0 to 65535 pulses.



The counter receives the pulses to be counted at the PLC input (**%I0.0**). If the counter enable input (**%I0.2**) is configured and at state 1, the pulses are counted by the counter and the counting value (current value FC.V) is constantly compared to 1 or 2 thresholds, FC.S0 and FC.S1, which are defined during configuration but can be modified by the program.

Resetting the counter to 0 is prompted by the rising edge of the input (**%I0.1**).

The FC threshold outputs 0 and 1 (**%Q0.1 and %Q0.2**) are controlled directly by the fast counter (without waiting for the outputs to be updated at the end of the scan) according to a matrix defined during configuration.

Output	FC.V < thresh. 0 < thresh. 1	thresh. 0 ≤ FC.V ≤ thresh. 1	thresh. 0 < thresh. 1 < FC.V
%Q0.1	0 or 1	0 or 1	0 or 1
%Q0.2	0 or 1	0 or 1	0 or 1

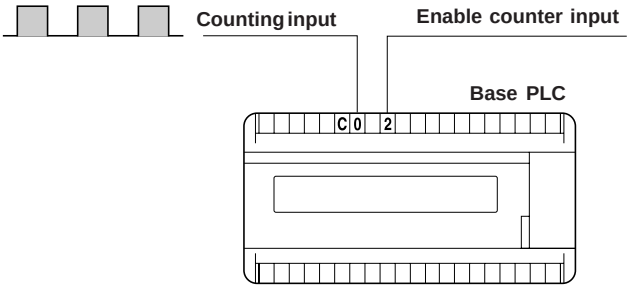
Certain commands (enable counter, set current value to 0), can also be executed from the user program via specific instructions.

Note :
The software setup (configuration, special instructions, etc) and the timing diagram of the fast counter operation are explained in Part B, Section 3.4-5.

4.4-2 Use as a frequency meter

The frequency meter function measures the frequency of a periodic signal (in Hz). The principle of this measurement is to count the number of pulses received within a specified period or time base. The frequency range which can be measured is from 1 Hz to 10 kHz. This function can, in particular, be used to manage an analog input module.

Time base	Measurement range	Accuracy	Refresh
100ms	10Hz-10kHz	0.1% for 10kHz 10% for 100Hz	10 times per second
1s	1Hz-10kHz	0.01% for 10kHz 10% for 10Hz	Once per second

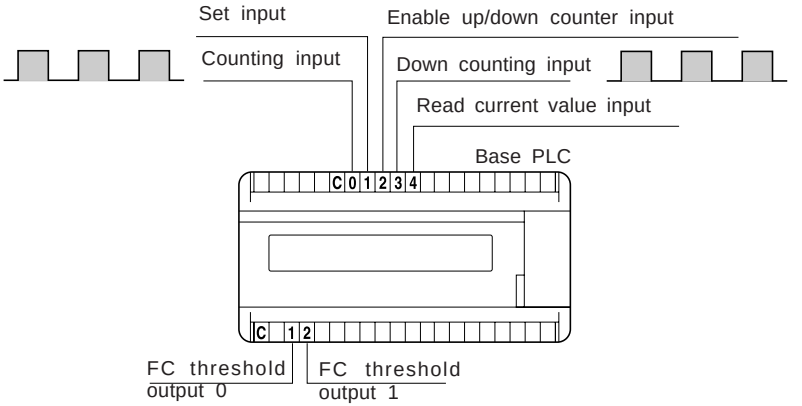


The frequency meter receives pulses at the PLC input (**%I0.0**). If the counter enable input (**%I0.2**) is configured and at state 1, the pulses are accepted by the counter, and the counting value (current value FC.V) changes throughout the measuring period. At the end of the measuring period, the current value FC.V corresponding to the frequency is read.
Input (**%I0.1**) resets the current value FC.V to 0.
This command (enable counter) can also be executed from the user program.

Note :
The software setup (configuration, special instructions) and the timing diagram of the frequency meter operation are explained in Part B, Section 3.4-5.

4.4-3 Use as an up/down counter

The fast up/down counter function counts up or down at a maximum frequency of 1 kHz with the possibility of counting from 0 to 65535 pulses.



The counter receives the pulses to be upcounted at PLC input (**%I0.0**) and the pulses to be downcounted at PLC input (**%I0.3**). If the up/down counting enable input (**%I0.2**) is configured and at state 1, the pulses are taken into account and the up/down counting value (current value FC.V) is constantly compared to 1 or 2 thresholds FC.S0 and FC.S1, which are defined during configuration and can be modified by the program. Information about whether the counter is counting up or down is available as a bit in system word SW111.

The preset value (0 to 65535), which is defined during configuration and can be changed by the program, is loaded into the current value on the rising edge of the input (**I0.1**).

Input (**%I0.4**) is used to read the current value FC.V on the fly.

Fast counter threshold outputs 0 and 1 (**%Q0.1 and %Q0.2**) are controlled directly by the fast up/down counter (without waiting for the outputs to be updated at the end of the scan).

The user can program outputs to be either 0 or 1 under any of the following conditions :

- when FC.V < threshold 0 < threshold 1Z
- when threshold 0 - FC.V - threshold 1
- when threshold 0 < threshold 1 < FC.V

Certain commands (up/down counter enable, set) can also be executed from the user program.

Note :

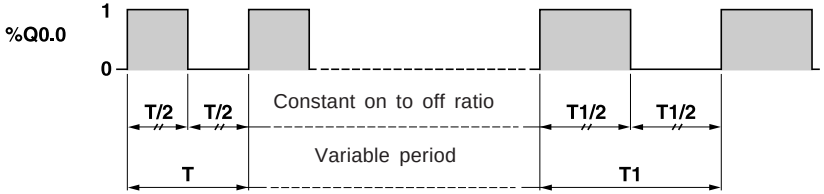
The software setup (configuration, special instructions) and the timing diagram of the up/down counter operation are explained in Part B, Section 3.4-5.

4.5 PULSE output : generating a pulse train

On a base or peer PLC, output %Q0.0 can be assigned to the special "PULSE" function, after configuration via the terminal.

Principle

A user-defined function block (%PLS) can generate a signal on output %Q0.0. This signal has a variable period but has a constant duty cycle, or on to off ratio, of 50% of the period.



The value of period T and the number of pulses to be generated are defined by configuration of function block %PLS.

Configuration parameters :

- Definition of the period : T = TB x %PLS.P
  TB = time base
  - 0.1 ms (can only be used on a PLC with transistor outputs)
  - 10 ms (default value), or 1s.

  %PLS.P = preset value :
  - 0 - %PLS.P < 32767 with TB = 10ms or 1s
  - 0 - %PLS.P < 255 with TB = 0.1ms
- Definition of number of pulses to be generated on output %Q0.0 : %PLS.N
  The number of pulses to be generated in period T (%PLS.N) can be limited or unlimited depending on the definition made at configuration :
  - 0 < %PLS.N <32767
  - %PLS.N=0 : unlimited generation

Range of periods available include :

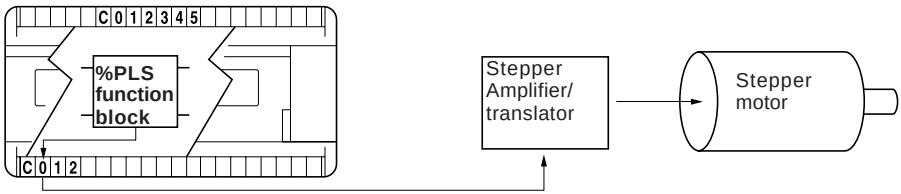
- 0.2 ms to 26 ms in steps of 0.1ms (38 Hz to 4.9 kHz).
- 20 ms to 5.45 min in steps of 10 ms
- 2 s to 9.1 hours in steps of 1s.

Note :

The complete parameter setting for function block %PLS is defined in Part B, Section 3.4-4.

Example of application : control of a stepper motor

Base PLC

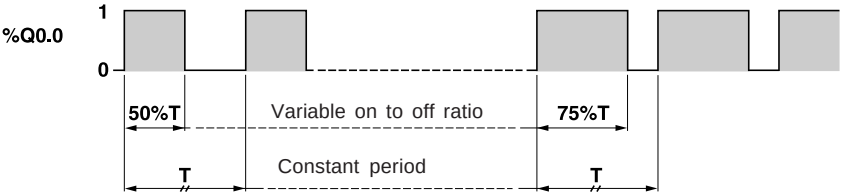


4.6 PWM output : pulse width modulation

On a base or peer PLC, output %Q0.0 can be assigned to the specific "PWM" function, after configuration via the terminal. This function, from TSX07 version V3 or later, can be used, in particular, to manage analog output modules.

Principle

A user-definable function block (%PWM) generates a signal on output %Q0.0. This signal has a constant period with the possibility of varying the duty cycle, or on to off ratio.



The value of period T and the percentage of the signal at state 1 in a period are defined in the configuration of function block %PWM :

Configuration parameters :

- Definition of period : T = TB x %PWM.P
  TB = time base :
  - 0.1ms (obligatory for analog output management; can only be used on a PLC with transistor outputs),
  - 10ms (default value), or 1s

  %PWM.P = preset value (not used for analog output management) :
  - 0 < %PWM.P - 32767 with time base 10ms or 1s
  - 0 < %PWM.P - 255 with time base 0.1ms
- Definition of period ratio : %PWM.R = Tx(%PWM.R/100)
  %PWM.R gives the percentage of signal at state 1 in a period (0<%PWM.R-100).

Range of periods available :

- 0.2 ms to 26 ms in steps of 0.2 ms (38 Hz to 4.9 kHz).
- 20 ms to 5.45 min in steps of 10 ms.
- 2 s to 9.1 hours in steps of 1s.

Note :

The complete parameter setting for function block %PWM is defined in Part B, Section 3.4-3.

Example of application : control of a dimmer switch

Base PLC

5.1 Power supply characteristics

PLC		TSX 07 •• ••08/28/48 TSX AMN 4000	TSX 07 •• ••02/12/22 TSX AMN 4001
Power supply		∿ (AC)	⎓ (DC)
Supply voltage	nominal	100 to 240V	24V
	limit	85 to 264V	19.2 to 30V
Frequency	nominal	50/60 Hz	_
	limit	47 to 63 Hz	_
Power required		-30VA	-14W
Inrush current	typical 20 A for 1 ms, max 40 A		
Integral and electronically fused (1) sensor supply		⎓ 24V /150 mA	_
Micro-cuts	duration	-10ms	-1 ms
Primary/ground isolation		2000V rms -50/60 Hz	2000V rms -50/60 Hz
IEC 1131-2 conformity		yes	yes

(1) Except on TSX 07 •1 1648 PLC

5.2 24VDC and 115VAC discrete input characteristics

PLC			TSX 07 •• ••02/08/12/22/28		TSX 07 •1 1648
Type			⎓ **24V** (resistive)		∿ **115V** (capacitive)
Logic (1)			pos. (sink)	neg. (source)	_
Sensor common			to + of supply	to - of supply	_
Type			isolated		isolated
Nominal input values	voltage		24V	24V	110/120V
	current		7 mA (2)	7mA (2)	10 mA
	peak current on activation		_	_	300 mA (at 110V)
	sensor supply		19.2 to 30V (including ripple)		_
	frequency		_	_	50 to 60 Hz
	current limitation		N/A	- 1.5A	N/A
Input limit values	state 1 voltage		•11V	- 8V	•79V
		current	•2.5mA at 11V	• ⎩2.5⎭ mA at 8V	• 4mA at 79V
	state 0 voltage		- 5V	Vps - 5V	- 20V
		current	- 1.2 mA	- ⎩1.2⎭ mA	- 2 mA
	frequency		_	_	47 to 63 Hz
Filtering	default values	0 to 1	12 ms	12 ms	12 ms
		1 to 0	12 ms	12 ms	12 ms
	program-mable values	0 to 1	100µs/3ms/12ms on %I0.0 to %I0.7		_
			375µs/3ms/12ms on %I0.8 to %I0.13		_
		1 to 0	100µs/3ms/12ms on %I0.0 to %I0.7		_
			375µs/3ms/12ms on %I0.8 to %I0.13		_
Isolation	between input and grounding		1500V rms -50/60 Hz		1500V rms -50/60 Hz
Compatibility 2-wire prox sens			yes (TE)	yes (TE)	yes (TE)
Compatibility 3-wire prox sens			yes	yes	_
IEC 1131-2 conformity			yes (type 1)	N/A	yes (type 1)

Note : The I/O characteristics are given for a load rate of 100% for PLCs with 10 I/O and 80% for PLCs with 16 and 24 I/O.
Load = total number of I/O simultaneously at state 1 divided by total number of I/O in the PLC.

(1) Positive or negative logic depending on the wiring (sink or source).
(2) 13 mA (or -13 mA) for %I•.0 Input.

5.3 24VDC discrete transistor output characteristics

PLC		**TSX 07 •• •• 02/08**	**TSX 07 •• •• 12**
Type		Unfused transistors (sink)	Protected transistor transistors (source)
DC load	nominal voltage	⎓ 24V	∿ 24V
	nominal current	0.5 A	0.5 A
	tungsten filament lamp	- 10W	- 10W
Limit values	voltage (including ripple)	19.2 to 30V	19.2 to 30V
	current	0.625 A at 30V	0.625 A at 30V
Logic		negative (sink)	positive (source)
Load common		to **+** of power supply	to **-** of power supply
Leakage current at state 0		- 1 mA	- 1 mA
Voltage drop at state 1		- 1.5 V (for I = 0.5 A)	- 2V (for I = 0.5 A)
Response time on resistive load	0 to 1	- 1 ms	- 1 ms
	1 to 0	- 1 ms	- 1 ms
Electronic fusing for	short-circuits and overloads	none (fit 1 fuse per output common)	yes
	overvoltages	yes	yes
	polarity inversions	yes	yes
Isolation	betw. output and grounding	1500V rms -50/60 Hz	1500V rms -50/60 Hz
IEC 1131-2 conformity		N/A	yes

Notes :

• The I/O characteristics are given for a load rate of 100% for PLCs with 10 I/O and 80% for PLCs with 16 and 24 I/O.

Load rate = total number of I/O simultaneously at state 1 divided by total number of I/O in the PLC.

• In this case, it is possible to place 2 outputs in parallel. A discharge diode must be placed on the PLC output terminals (not the load terminals).

5.4 Discrete relay output characteristics

PLC		TSX 07 •• •• 22/28/48			
Type		relay			
AC load	AC12 resistive duty	voltage	∿ 24V	∿ 48V	∿ 110V	∿ 220V
		current	2A (1) 1A (2)	1A (2)	1A (2) 0.5A (4)	1A (3) 0.5A (4)
	AC15 inductive (continuous) duty	voltage	∿ 24V	∿ 48V	∿ 110V	∿ 220V
		current	1A (1) 0.5A (3)	0.5A (3) 0.2A (5)	0.45A (3) 0.090A (5)	0.22A (3) 0.045A (5)
DC load	DC12 resistive duty	voltage	⎓ 24V	_	_	_
		current	1A (1)	_	_	_
	DC13 inductive duty	voltage	⎓ 24V	_	_	_
		current	0.4A (3)	_	_	_
Response time	opening	- 5 ms			
	closing	- 10 ms			
Electronic fusing for	overloads and short-circuits	none, fit 1 fuse per I/O point or group of I/O points			
	inductive ∿ overvoltages	none, an RC circuit or an MOV (ZNO) suppressor must be fitted in parallel with the terminals of each output			
	inductive ⎓ overvoltages	none, a freewheel diode must be fitted in parallel with the terminals of each output			
Isolation	betw. output and grounding	1500V rms - 50/60 Hz			
IEC 1131-2 conformity		yes	yes	yes	yes

Note :
The I/O characteristics are given for a load rate of 100% for PLCs with 10 I/O and 80% for PLCs with 16 and 24 I/O.
Load rate = total number of I/O simultaneously at state 1 divided by total number of I/O in the PLC

(1) 0.3 x 10^6 operations
(2) 0.5 x 10^6 operations
(3) 1 x 10^6 operations
(4) 2 x 10^6 operations
(5) 10 x 10^6 operations

5.5 Characteristics of analog I/O (TSX AMN 4000/4001)

5.5-1 Characteristics of analog inputs

Input	channel number	3 (high level)
	input range	0..10 V, +/-10 V 0..20 mA, 4..20 mA
	input impedance	100 KΩ (1) 125 Ω (2)
	permissible voltage without damage	+/- 30 V as voltage +/- 7.5 V as current
Resolution	channel 0	11 bits (+ sign at +/- 10 V) (4000 points at +/- 10 V)
	channel 1	7 bits (+ sign at +/- 10 V) (4) 11 bits (+ sign at +/- 10 V) (3) (250 points at +/- 10 V) (4) (4000 points at +/- 10 V) (3)
	channel 2	7 bits (+ sign at +/- 10 V) (250 points at +/- 10 V)
Conversion	conversion time	10 ms per channel
	maximum error from 0 to 60 °C	0.5% of the full scale
Isolation	voltage between supply and input channels	∼ 2000 V
	voltage between input channels	∼ 1000 V
	resistance between input channels and ground	> 10 MΩ

(1) Voltage type input
(2) Current type input
(3) Value when channels 0 and 1 are configured
(4) Value when channels 0, 1 and 2 are configured

5.5-2 Characteristics of the analog output

Output	channel number	1 (high level)
	output range	0..10 V, +/-10 V 0..20 mA, 4..20 mA
	load impedance	> 2 KΩ (1), < 600 Ω (2)
	permissible voltage without damage	+/- 30 V (1), +/- 12 V (2)
Resolution		11 bits (+ sign at +/- 10 V) (4000 points at +/- 10 V)
Conversion	conversion time	2 ms
	max. error from 0 to 60 °C as voltage	1% of the full scale
	max. error from 0 to 60 °C as current	1.5% of the full scale
Isolation	voltage between supply and the output channel	∿ 2000 V
	voltage between I/O channels	∿ 1000 V
	resistance between output channel and ground	> 10 MΩ

(1) Voltage type output
(2) Current type output

5.6 Characteristics of analog I/O (TSX AEN/ASN •••)

5.6-1 Characteristics common to analog inputs and outputs

Supply	nominal voltage	⎓ 24V
	limit voltages	⎓ 21V to 30V
	nominal current at 24V	104 mA
	inrush current	10 A max
	power drawn	2.5 W
Isolation	voltage between supply and ground	∿ 1500 V
	voltage between input or output and ground	∿ 1500 V
	resistance between supply and ground (500V)	> 10 MΩ
	resistance between input or output and ground (500V)	> 10 MΩ
Shocks	300 m/s^2, 3 shocks per axis, 3 axes	
Vibrations	5 to 55Hz, 60 m/s^2, 2 hours per axis, 3 axes	
Climatic environment	operating temperature	0 to 60 °C
	storage temperature	-25 °C to 70 °C
	relative humidity (without condensation)	45 to 85 %
	altitude	0 to 2000m

5.6-2 Characteristics of analog inputs

Input	channel number	1 (high level)
	input impedance	6.6 MΩ (1) 250 Ω (2)
	permissible voltage without damage	+/- 16 V
Conversion	conversion method	U (V) ∅ F (HZ)
	resolution	10 bits or 12 bits
	conversion time	125 ms (3) /500ms (4)
	precision (full scale)	+/- 1% 0 to 60 °C
Frequency output	nominal voltage	⎓ 24V
	logic (switch on front panel)	positive or negative
	protection against short-circuits	no
Isolation	voltage between input and frequency output	∿ 500 V
	resistance between input and frequency output	>10 MΩ

(1) 0/10V and -10/+10V modules
(2) 4/20 mA module
(3) 10-bit resolution
(4) 12-bit resolution

5.6-3 Characteristics of analog outputs

Output	channel number	1 (high level)
	output signal value	0/10 V
	load impedance	-5 KΩ (1) -250 Ω (2)
	permissible voltage without damage	+/- 12 V
	protection against short-circuits	yes (continuous)
Conversion	conversion method	F (Hz) ∅ U (V)
	resolution	8 bits
	conversion time	from 0 to 90%: 500ms max
	precision (full scale)	+/- 1% 0 to 60 °C
Frequency input	nominal voltage	⎓ 24V
	logic (switch on front panel)	positive or negative
	input frequency	312.5 Hz
Isolation	voltage between output and frequency input	∿ 500 V
	resistance between output and frequency input	>10 MΩ

(1) 0/10V and -10/+10V modules

(2) 4/20 mA module

5.7 Characteristics of the analog input (TSX 07 32/33 ••28)

Analog input	channel number		1 (high level)
	input signal value		0 / 10 V
	value of an LSB		40 mV
	input impedance		16 KΩ < Z < 18 KΩ
	maximum permissible voltage without damage		+/- 16 V
	protection against polarity inversion		yes
Conversion	conversion method		approx. successive
	resolution		8 bits
	conversion time		1 PLC scan
	precision at full scale		+/- 0.8% at 25 °C
			+/- 2% at 60 °C
	drift		0.34% / 10 °C
	repeatability at full scale		+/- 0.8% from 0 to 60 °C
Isolation	between analog channel and CPU		none
	distance between sensor and analog input :		
		isolated sensor	<30m with shielded cable
		non-isolated sensor	<10m with shielded cable

Important

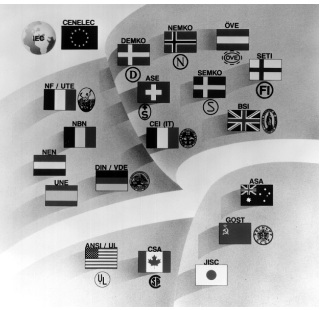
If using a sensor which is not isolated from ground, the - of the analog input must be grounded.

5.8 Service conditions

5.8-1 Standards

TSX Nano PLCs have been developed to comply with the main national and international standards concerning electronic industrial control devices.

- Specific PLC requirements : operating characteristics, filtering, ruggedness, safety, etc.
  EN61131-2 (IEC1131-2), CSA 22.2, UL508
- Strict limitation of electromagnetic interference generated : CE marking (European Directives on Low Voltage and EMC) **EN 50081-1, class B.**



- Electrical qualities and self-extinguishing capability of insulating materials : UL 746C, UL 94, etc.
- Marine certification
  - Bureau Veritas
  - DNV
  - GL (pending)

5.8-2 Environment, normal service conditions

- **Climatic environment, normal conditions**

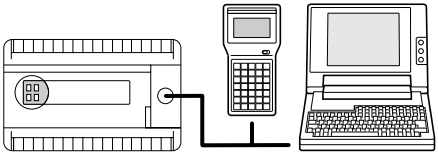
Temperature	Humidity and altitude
Operating temperature : 0 to 60°C (1)	Relative humidity **:** 5% to 95% (2) (non-condensing)
Storage temperature : - 25°C to + 70°C	Altitude : 0 to 2000 meters (0-6500 feet)

- **Resistance to vibration :** In accordance with IEC 68-2-6 FC tests
- **Resistance to mechanical shocks :** In accordance with IEC 68-2-27 EA tests

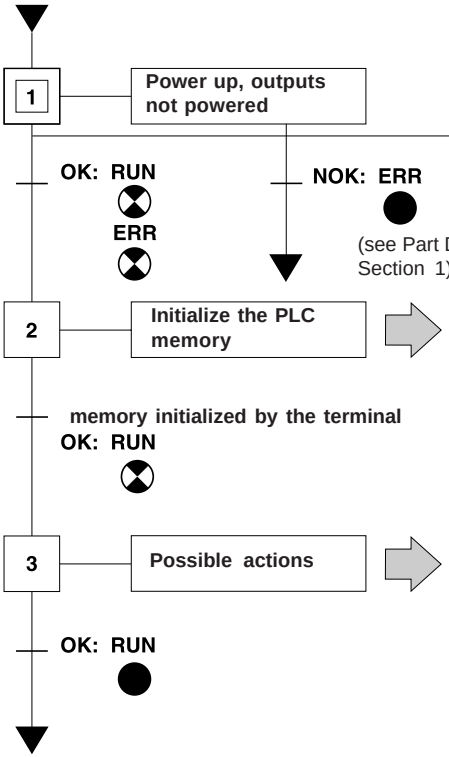
(1) 0 to 55°C with TSX 07 •• •• 12 when mounting vertically (on vertical plane)
(2) 45% to 85% for TSX AMN 4000/4001 modules

6.1 Procedure for first power-up

The TSX Nano incorporates several self-tests which continually monitor its operation.
The results of these self-tests are displayed on the front panel of the PLC. They can be made more detailed using the terminal. (1)



The diagram below shows the procedure for powering up the PLC for the first time.



- Connect the terminal.
- Initialize PLC memory by clearing the whole memory (Prg menu, Clr command).

Note : if using the PL7-07 software, create a "new" application and transfer it to the PLC.

- Check I/O connections.
- Configure and enter the program from the terminal :

- either directly into PLC memory (TSX mode)
- or into terminal memory (FTX mode), then transfer to the PLC memory.

- Set to RUN using RUN/STOP input %10.• if it is configured, or via the terminal.

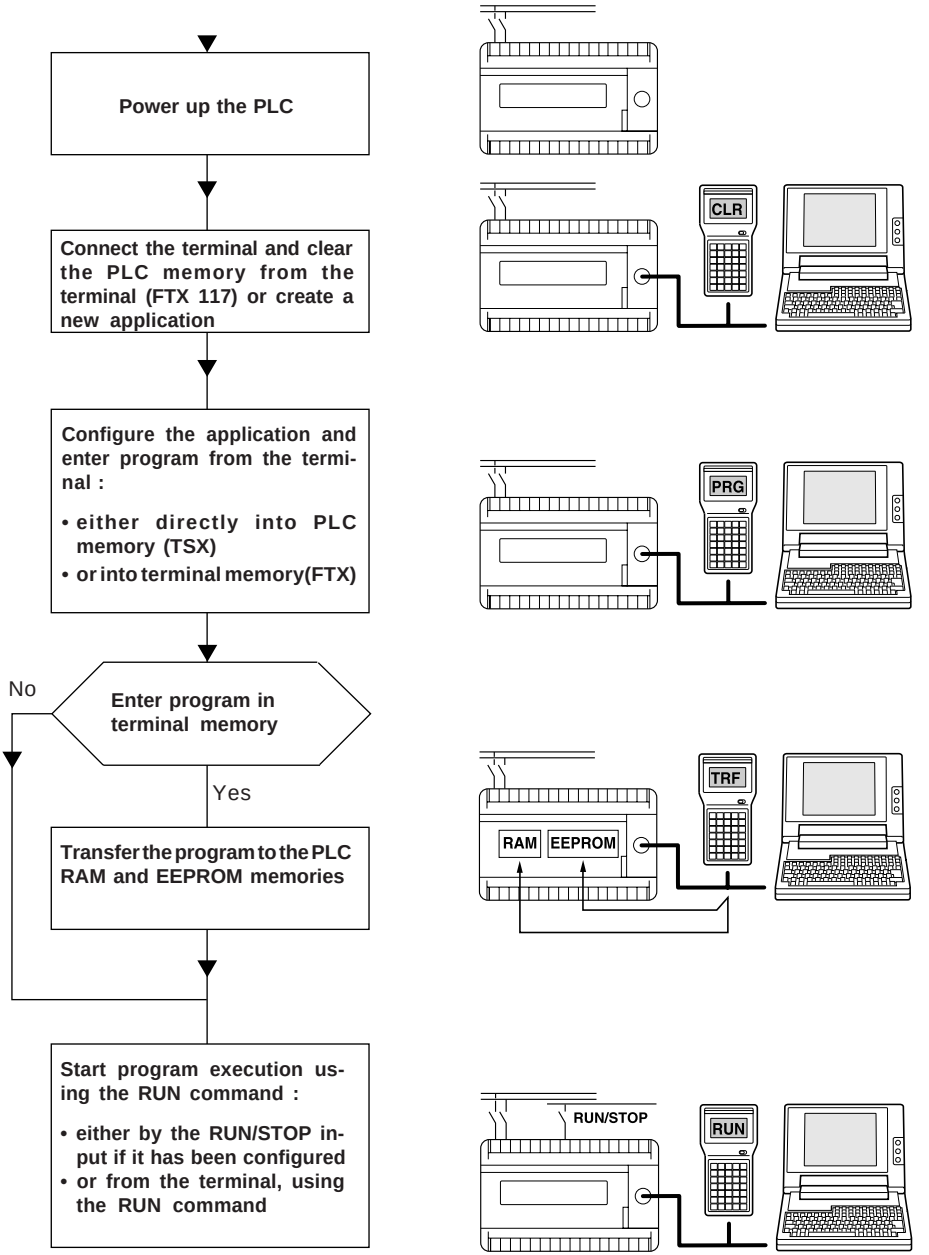
Key

LED flashing

LED lit continuously

(1) herein, the word "terminal" represents the FTX 117 Handheld terminal, or a PC compatible terminal or FTX 417/517/FT 2000 running PL7-07 PC programming software for the TSX Nano PLC.

Summary


Power up the PLC
Connect the terminal and clear the PLC memory from the terminal (FTX 117) or create a new application
CLR
Configure the application and enter program from the terminal :
• either directly into PLC memory (TSX)
• or into terminal memory(FTX)
PRG
No
Enter program in terminal memory
Yes
Transfer the program to the PLC RAM and EEPROM memories
TRF
RAM
EEPROM
Start program execution using the RUN command :
• either by the RUN/STOP input if it has been configured
• or from the terminal, using the RUN command
RUN/STOP
RUN

6.2 Checking the I/O connections

- **Principle**

  This check ensures that :
  - data from the sensors is received by the inputs and transmitted to the processor,
  - commands from the processor activate the outputs and are transmitted to the corresponding output devices.

- **Recommendations**

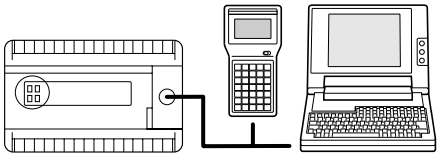
  In order to avoid any random movement of the machine, it is recommended that :
  - power fuses be removed from the motor controls,
  - pneumatic and hydraulic inputs be closed.

- **Procedure**

  - Power up for the first time, as shown in Section 6.1. Ensure that the I/O LED is not lit continuously.

  - Leave the PLC in STOP.

  - Select the Data Editor on the FTX 117 terminal, or the PL7-07 software.



TSX Nano PLC **Terminals**

  - Change the state of system bit %S8 (PLC status output) to state 0.

  - Check the inputs by activating each sensor :
    - Check that the LED on the front panel for the corresponding input changes state.
    - Check on the terminal screen that the corresponding bit also changes state.

  - Check the outputs using the terminal :
    - Force the bit corresponding to each output to state 1.
    - Check that the LED for the corresponding output changes state, as well as the associated output device.

  - Using the terminal :
    - Remove all forcing
    - Reset system bit %S8 to state 1 (the outputs are set at 0).

Note :

If the PLC does not contain the application program, this entire procedure can be performed in RUN. In this case, bit %S8 can remain at state 1 (default state).

7.1 Power outages and returns

- **Characteristics of power outages**
  - If power outages < power supply ridethrough : normal execution of the program
  - If power outages > power supply ridethrough : processor powered down with context (data & program) saved.

- **Warm restart : PLC restarts with the contents of the data memory in the same state as at the time of the power outage.**

  List of possible causes :
  - PLC restart after power outage > power supply ridethrough
  - System bit %S1 set to state 1 by the program or the terminal

  Consequences upon return of power
  - System bit %S1 set to state 1,
  - **Non-forced input bits set to state 0 (warning : setting to zero generates a "false" rising edge on an input physically at 1 during a warm restart,** see part B, section 2.1-2).
  - All I/O bits set to 0,
  - All unsaved internal bits set to state 0 (%M64 to %M127)
  - The state of the stored internal bits (%M0 to %M63) and current values of the function blocks (timers, counters, etc) are maintained
  - The scan is restarted from the point at which it stopped at the power outage, without updating the outputs at the end of the scan. The scan is then restarted normally : Reading the inputs ∅ program processing ∅ updating the outputs and setting system bit %S1 to 0

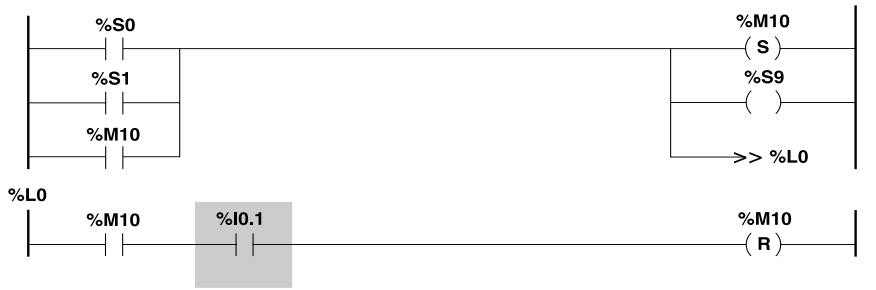
- **Cold restart : PLC restarts with loss of contents of the data memory**

  List of possible causes :
  - Back-up battery defective or absent
  - System bit %S0 set to 1 by the program or the terminal
  - Initialization of the PLC by the terminal

  Consequences upon return of power
  - System bit %S0 set to state 1
  - All internal bits, I/O bits and internal words set to state 0
  - The current values of the function blocks (timers, counters, etc), registers and step counters set to state 0
  - Adjustment values modified by the terminal lost, and preset values defined during configuration recalled
  - System bits and words initialized (except %S0 and the schedule block (RTC) data)
  - Forcing cancelled
  - Grafcet reinitialized
  - Scan restarted from the beginning :
    Reading the inputs ∅ program processing ∅ updating the outputs and setting system bit %S0 to 0.

Programming example

In order to avoid automatic restarting of the control system when power returns, the program below enables manual intervention by pressing a "RESTART" button and keeps the outputs at 0 during the power outage. This program must be added to the application program.
One part of this program must be at the start of the program (pre-processing zone), the other part must be at the end of the program (post-processing zone).



End of program

000	LD %S0	If %S0 is at logic level 1: cold restart	in preprocessing zone (start of program)
001	OR %S1	OR %S0 is at logic level 1: warm restart	
002	OR %M10	OR %M10 is at logic level 1: latch	
003	S %M10	Set internal bit %M10 to 1	
004	ST %S9	Set outputs to 0	
005	JMPC %L0	Jump to label %L0 if %S9 is at 1	
006			Application program
007			

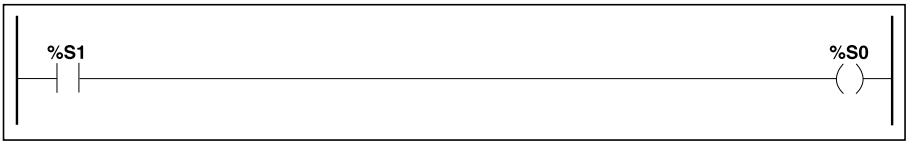
098			
099			
100	%L0 :	Jump address	in post-processing zone (end of program)
101	LD %M10	If %M10 is at logic level 1	
102	AND %I0.1	AND input 1 is at logic level 1: RESTART	
103	R %M10	Set internal bit %M10 to 0	
104	END	End of program	

7.2 Initializing the PLC

Presentation

The PLC can be initialized by the program by setting system bit %S0 to 1, which corresponds to a cold restart (see Section 7.1). During a warm restart, it may be necessary to initialize the PLC. The example below shows how this can be programmed. Initialization can also be requested from the terminal using the INIT command.

Programming




```
LD   %S1
ST   %S0
```

If %S1 = 1 (warm restart), set %S0 to 1 to initialize the PLC. These two bits are reset to 0 by the system at the end of the following scan.

IMPORTANT

System bit %S0 must not be set to 1 for more then one PLC scan.

7.3 Saving the program and data

Saving in the RAM

The user program and the data are contained in the PLC RAM memory. A battery in the PLC provides the memory with independent operation for 30 days.

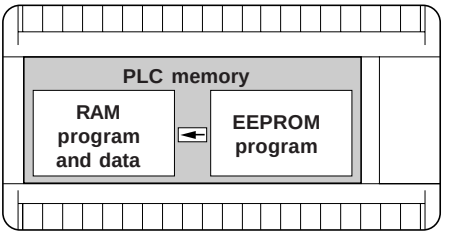
WARNING

This independence is ensured only if the battery is charged for at least 6 consecutive hours before the PLC is stopped.

EEPROM storage

It avoids the risk of corrupting the program held in the RAM during a battery failure or a power outage lasting longer than 30 days.

On power-up, the PLC checks the integrity of the programs contained in the RAM and EEPROM memories. In the event of data corruption, the program contained in the EEPROM is automatically transferred to the RAM if the MST option (autoloading) has been selected (see part C, section 12.4).



WARNING

Once the application has been debugged, it is advisable to transfer it into EEPROM with the MST option selected

Part B contains 2 levels of information :

- That which enables the user to carry out simple functions. In this case, it is not necessary to read the entire manual, only to refer to the paragraphs in gray.
- That which enables the user to carry out all the functions offered by the TSX Nano. In this case, the whole manual is relevant.

1.1 General

The control decisions a PLC makes are based on a control program that you create. Creating a TSX Nano PLC control program consists of writing a program in one of the control languages supported by the TSX07 PLCs.

The FTX 117 Handheld Terminal supports the List Programming Language. The List language is a line based textual boolean-like language which can also process numerical operations.

The PL7-07 PC Programming Software supports the List and Ladder programming languages. Ladder is a rung-based graphical Boolean language. Further, PL7-07 allows you to reverse a program from Ladder to List and from List to Ladder.

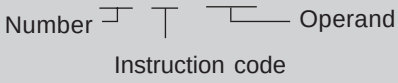
In addition, the TSX Nano PLCs support Grafcet List instructions.

1.2 Instruction List (List or IL)

Program structure
A program in PL7 language comprises a series of instructions (up to 1000 instructions) of various types.

Each program line has a number which is generated automatically, an instruction code and a bit or word type operand.

Example of an instruction : 003 LD %I0.1



Ladder diagrams are a graphical means of displaying a logical expression. An alternative to a Ladder diagram is an Instruction List program. An Instruction List program is a series of logical expressions written as a sequence of Boolean instructions. All the Boolean instructions, except for LOAD, STORE and NOT, operate on two operands, which can be either TRUE or FALSE, and produce a single value, either TRUE or FALSE.

As written, List instructions have only one explicit operand; the other is implied. The implied operand is the Boolean accumulator. In operation, a List instruction performs the specified operation on the contents of the accumulator and the operand, and replaces the contents of the accumulator with the result.

For example, the operation AND %I1.2 would perform a logical AND between the contents of the accumulator and Input 1.2, and would replace the contents of the accumulator with that result.

B

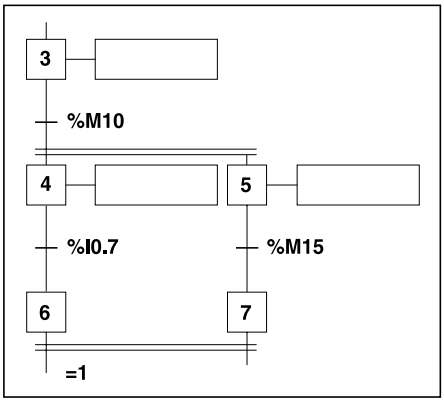
The LOAD and STORE instructions either load the accumulator with the value of the operand, or store the accumulator to the operand, respectively. The NOT instruction has no explicit operands and simply inverts the state of the accumulator.

Instructions

Types of instruction	**Example**	
• Bit instruction	004 LD %M10	Reads internal bit %M10
• Block instruction	008 IN %TM0	Starts the timer %TM0
• Word instruction	010 [%MW10 := %MW50+100]	Addition
• Program instruction	015 SR5	Calls up subroutine #5
• Grafcet instruction	020 -*-8	Step #8

1.3 Grafcet

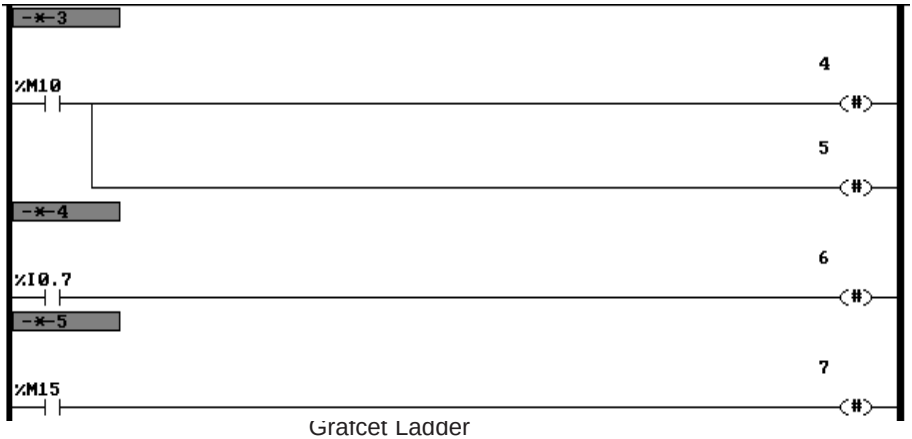
Grafcet is an analytical method which divides any sequential control system into a series of steps, with which actions, transitions and conditions are associated.
PL7-07 language has specific Grafcet instructions.
Graphical Grafcet is not supported in PL7-07 PC Programming Software.



Graphical Grafcet

```
-*-    3
LD     %M10
#      4
#      5
-*-    4
LD     %I0.7
#      6
-*-    5
LD     %M15
#      7
...
```

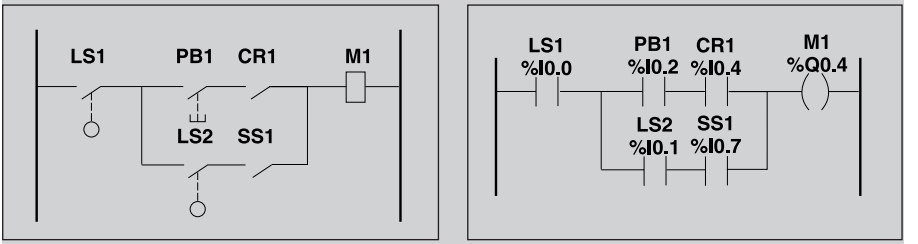
Grafcet list



Grafcet Ladder

1.4 Ladder diagrams

Ladder diagrams are similar to relay logic diagrams used to represent relay control circuits. The main differences between the two are that in ladder programming, all inputs are represented by contact symbols (-| |-) and all outputs are represented by coil symbols (-( )-), and that numerical operations are included in the graphical Ladder instruction set.

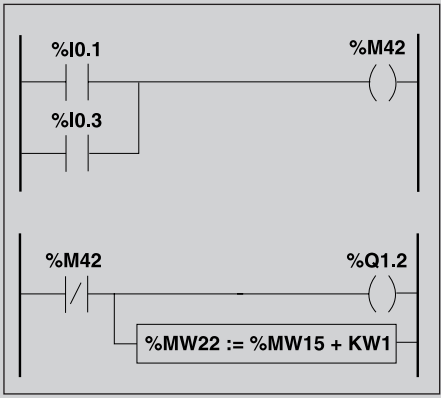


The figure above illustrates a simplified wiring diagram of a relay logic circuit and its equivalent Ladder diagram. Notice that in the Ladder diagram, all inputs associated with a switching device in the relay logic diagram are shown as contacts. The M1 output coil in the relay logic diagram is represented with an output coil symbol in the Ladder diagram. The address numbers appearing above each contact/coil symbol in the Ladder diagram reference the location of the external input/output connections to the PLC.

A program written in Ladder is composed of rungs—sets of specific graphical instructions drawn between the two vertical outer bars representing potential—that are executed sequentially by the PLC. The set of graphical instructions represent :

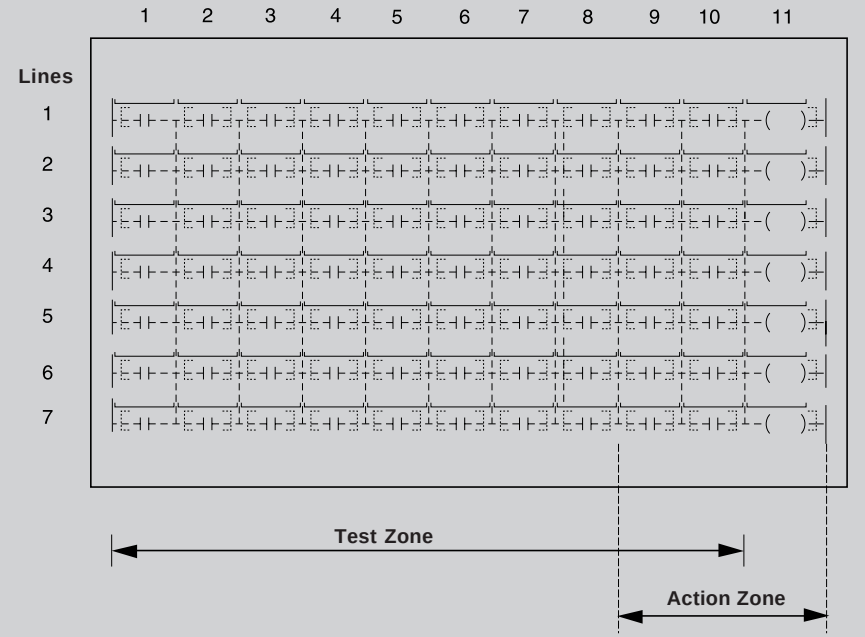
- the inputs/outputs of the PLC (push buttons, sensors, relays, pilot lights...),
- the functions of the PLC (timers, counters...),
- the math and logic operations (addition, division, and, xor...),
- the comparison operators and other numerical operations (A < B, A = B, shift, rotate...),
- the internal variables in the PLC (bits, words...).

These graphical instructions are arranged with vertical and horizontal connections leading ultimately to one or several outputs and/or actions.
A rung cannot support more than one group of linked instructions.
For example, this Ladder program is composed of two rungs.



1.4-1 Programming principles

Each Ladder rung is comprised of 7 lines and 11 columns and are organized in two zones :



- a test zone which contains the conditions that must be true for an action to take place.
- an action zone which contains the output or operation that results from the tests connected to it.

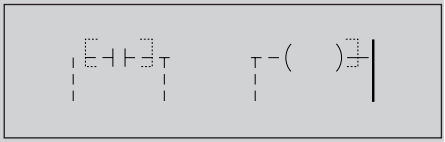
The rung is visualized by a 7 by 11 programming grid, starting in the upper-leftmost cell of the grid. You program test instructions, comparisons, and functions in the test zone and these test instructions are left-justified. This test logic then provides continuity to the action zone where you program coils, numerical operations and program flow control instructions. These action instructions are right-justified. The rung is solved, or executed (tests made and outputs assigned), from top to bottom and from left to right.

In addition to the rung, there is the Rung Header that appears directly above the rung. The rung header is used to document the logical intent of the rung. The rung header contains the rung number, any labels (%Li:) or subroutine declarations (SRi:), the rung title, and rung comments. For more information about the Rung Header, and its correspondence to List Line Comments, see section B.1.4-3.

B

- **Contacts, coils and program flow instructions**

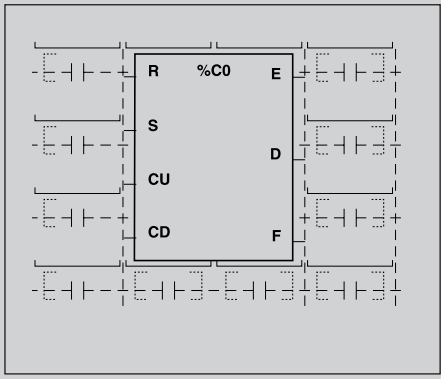
Contacts, coils and program flow (jump and call) instructions occupy a single cell of the programming grid. Function blocks, Comparison blocks and Operate blocks occupy multiple cells.



- **Function Blocks**

Function blocks are placed in the Test Zone of the programming grid. The block must appear in the first line; no Ladder instructions or lines of continuity may appear above or below the Function block. Ladder test instructions lead to the Function block's input side, and test instructions and/or action instructions lead from the block's output side.

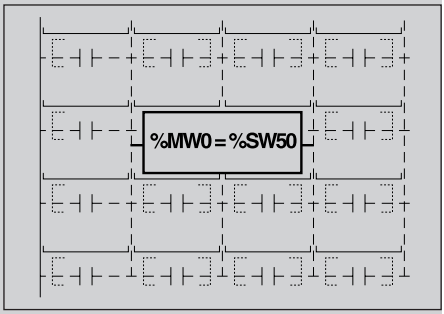
Function blocks are vertically orientated and occupy 2 columns by 4 rows of the programming grid.



- **Comparison Blocks**

Comparison blocks are placed in the Test Zone of the programming grid. The block may appear in any line or column in the test zone as long as the entire length of the instruction resides in the test zone.

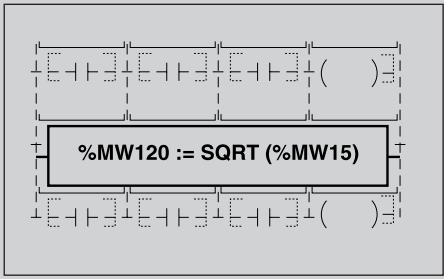
Comparison blocks are horizontally orientated and occupy 2 columns by 1 row of the programming grid.



- **Operate Blocks**

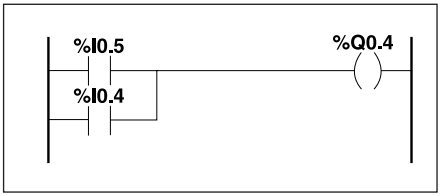
Operate blocks are placed in the Action Zone of the programming grid. The block may appear in any line in the action zone. The instruction is right-justified; therefore, it appears on the right and ends in the last column.

Operate blocks are horizontally orientated and occupy 4 columns by 1 row of the programming grid.



1.4-2 Reversibility

In this manual, the term reversibility refers to the ability of the PL7-07 Programming software for the TSX Nano to convert TSX Nano application programs from the Ladder programming language to the List language and back again. PL7-07 programs can be displayed in either format at will by setting a preference for one or the other. You can even convert (reverse) individual Ladder rungs to List and back simply by selecting the Tools menu's Toggle Ladder/List option.




```
LD    %I0.5
OR    %I0.4
ST    %Q0.4
```

Understanding of reversibility lies in the relationship of the "Rung"—the collection of Ladder programming instructions that constitute a logical expression—and the "Sequence"—the collection of List programming instructions that accomplish the same. The figure above demonstrates an example of a common rung in Ladder in any user program. Next to the Ladder rung is the equivalent program logic expressed as a List sequence.

In essence, the program you write, whether it be in Ladder or List, is kept internally as List. PL7-07 takes advantage of the program structure similarities between the two languages and uses this internal List image of your program to display it in the List / Ladder Viewer and Editors as either a List program (its basic form), or graphically as a Ladder diagram, depending upon the preference you select. Therefore, everything created in Ladder can always be reversed to List, but some List logic may not reverse to Ladder. To ensure reversibility from List to Ladder, it is important to follow the set of List programming conventions contained in section B.1.4-3.

1.4-3 Reversible List programming conventions

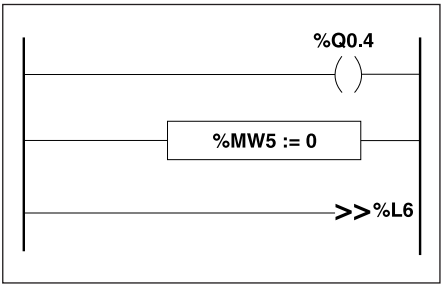
The structure of a reversible function block in List language requires the use of certain specific instructions. They are :
- BLK which marks the block start, and defines the beginning of the rung and the start of the input portion to the block,
- OUT_BLK (marks the beginning of the output portion of the block)
- END_BLK (obviously, marks the end of the block and the rung).

The use of the reversible Function Block instructions are not mandatory for a properly functioning List program. For some instructions it is possible to program in List which is not reversible. Non-reversible List programming is described in the Function Block descriptions in section B2.2.

Another important convention is to avoid the use of certain List instructions, or certain combinations of instructions and operands, which have no equivalence in Ladder. For example, the N instruction (bit Not inverses the boolean accumulator) has no equivalent Ladder instruction. In the table below is a listing of all the List programming instructions that will not reverse to Ladder.

List Instruction	Operand	Description
JMPCN	%Li	Jump Conditional Not
N	none	Negation (NOT)
ENDCN	none	End Conditional Not
XORN	any	XORN preceded by OR logic

Unconditional rungs also follow a List programming convention to ensure List-to-Ladder reversibility. An unconditional rung is a rung in which there are no tests or conditions; the output and/or action instruction(s) is (are) energized, or executed, all the time. The figure below illustrates unconditional rungs and List equivalent sequences.




```
LD      1
ST      %Q0.4
LD      1
[%MW5 := 0]
JMP     %L6
```

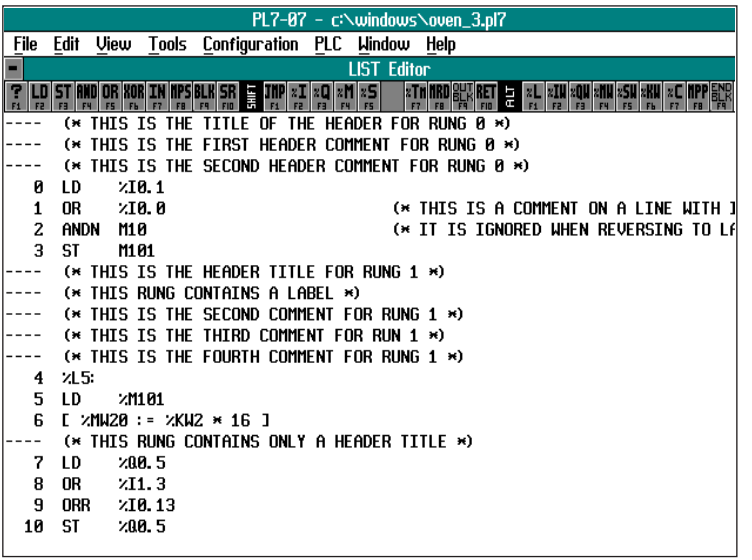
Notice that each of the unconditional List sequences, with one exception, begin with a Load instruction followed by the number 1. The combination sets the boolean accumulator value to one, and therefore sets the coil (store instruction) to one and sets %MW5 to zero on every scan of the program. The exception is the unconditional jump instruction. This List instruction is executed regardless of the value of the accumulator, and consequently does not need the accumulator set to one as do the previous two examples.

If you try to reverse a List program that is not completely reversible, the reversible portions are displayed in Ladder and the irreversible portions are displayed in “Ladder List Rungs.” Ladder List Rungs function just like a small list editor, enabling you to view and modify the irreversible parts of a Ladder program.

Program Documentation

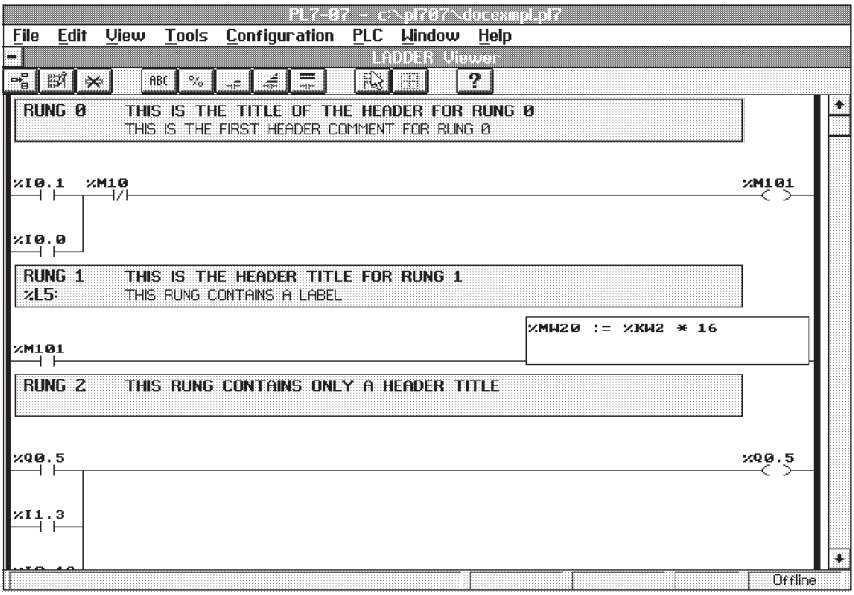
The List editor allows you to document your program with List Line Comments. These comments may appear on the same line as programming instructions, or they may appear on lines of their own. The Ladder Editor allows you to document your program with Rung Headers found directly above the rung.

PL7-07 uses these comments for reversibility. When reversing a program from List to Ladder, PL7-07 uses some of the List comments to construct a Rung Header. Simply, the comments that may be found between List sequences are used for the Rung Headers.



The first comment found on a line by itself (no instruction appears on the line with it) after the end of a List sequence is used for the Rung Title. Then, the next comments found after the first, if any exist, are used for the body of the rung. Once the body lines of the header are occupied, then the rest of the line comments between List sequences are ignored, as are any comments that are found on List lines that also contain List instructions.

When a Ladder rung that contains a Rung Header is reversed to List, the Rung Header documentation is inserted between the List sequences. Any labels or subroutine declarations (%Li: or SRi:) are placed on the next line following the header, immediately prior to the beginning of the List Sequence. If the rung that is reversed was originally written in List, and there were comments ignored when List was reversed to Ladder, those comments will reappear in the List Editor.



2.1 Boolean processing

2.1-1 Definition of the main bit objects

• I/O bits

The addressing system for these bits is described in Part A, Section 1.5.
These bits are the "logical images" of the electrical state of the I/O. They are stored in the data memory and updated during each scan of the program logic.

• Internal bits

Internal bits are internal storage areas available to the user for program use.

Note : I/O bits which are not in use cannot be used as internal bits.

• System bits

System bits %S0 to %S127 are used to monitor the correct operation of the PLC as well as the correct execution of the application program. The role and use of these bits are explained in Part B, Section 6.

• Step bits

Bits %X1 to %X62 are the bits associated with the Grafcet steps. Step bit Xi is set to 1 when the corresponding step is active, and to 0 when the step is deactivated.

• Bits extracted from words : see Section 3.1-1

List of operand bits

The following table lists all types of operand bits.

Type	Address (or value)	Maximum number	Write access (1)	See Sect.	
Immediate value		0 or 1	_	_	_
Bits	input	%I0.i or %I1.i (2)	28	no	1.5
	output	%Q0.i or %Q1.i (2)	20	yes	Pt. A
Bits	internal	%Mi	128 (3)	yes	
Bits	system	%Si	128	depends on i	5.1
Bits	of Grafcet steps (5)	%Xi	62	yes	2.3-1
Bits	of function blocks	%TMi.Q, %DRi.F, etc		no (4)	2.2-1
Bits	of reversible function blocks	E,D,F,Q,TH0,TH1		no	3.3-1
Bits	word extracts	varies	varies	varies	3.1-1

(1) Written by the program or in the Data editor by the terminal.
(2) Where i = 0 for a PLC base or a peer PLC, i = 1 for an I/O extension, and j = I/O point. I/O bits can be forced to 0 or 1 in the Data editor.
(3) The first 64 are saved in the event of a power outage.
(4) Except for %SBRi.j and %SCi.j, these bits can be read and written.

2.1-2 Introduction to Boolean instructions

Boolean instructions can be compared to Ladder language elements.

Test elements, example : the LD (LOAD) instruction is equivalent to an open contact.

LD %I0.0	Contact closed when the controlling bit is at state 1.

Action elements, example : the ST (STORE) instruction is equivalent to a coil.

ST %Q0.0	The associated bit object takes the logical value of the bit accumulator (result of previous logic).

Boolean equation :

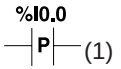
LD %I0.0 AND %I0.1 ST %Q0.0	The Boolean result of the test elements is applied to the action element.

Rising and falling edge

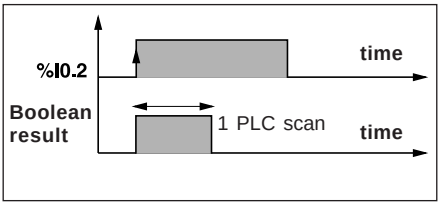
Test instructions can be used to detect rising or falling edges on the PLC inputs. An edge is detected when the state of an input has changed between scan n-1 and the current scan n, and it remains detected during the current scan.

The LDR instruction (R : Rising edge) is equivalent to a rising edge detection contact :

LDR %I0.0

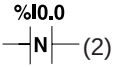


Rising Edge : detects a change of the controlling input from 0 to 1.

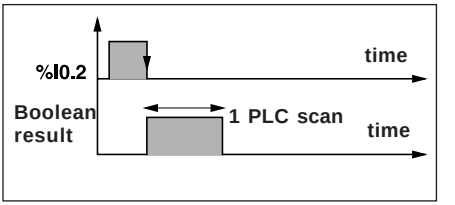


The LDF instruction (F : Falling edge) is equivalent to a falling edge detection contact :

LDF %I0.0

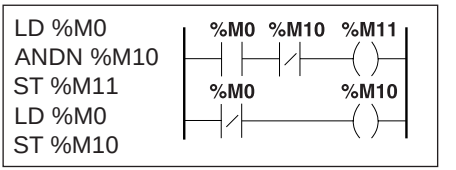


Falling Edge : detects a change of the controlling input from 1 to 0.



Instructions on a rising or falling edge apply to inputs %I, but it is possible to detect edges on any other bit (or Boolean result) using 2 internal bits.

In the example opposite, bit %M11 records the rising edge of bit %M0.



(1) **P**ositive transition sensing contact

(2) **N**egative transition sensing contact

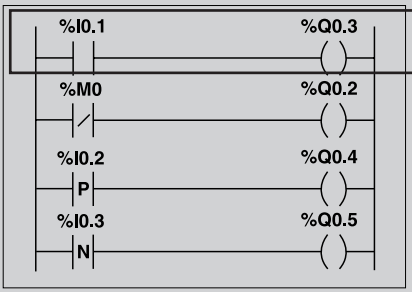
(3) On a cold or warm restart, the application detects a rising edge even if the input has remained at 1. This phenomenon can be masked by starting the program on LD %S1 and ENDC instructions.

Instruction format

The Boolean instruction described appears in bold type. Each equation is illustrated by the corresponding Ladder diagram.

Load instructions LD, LDN, LDR, LDF

The instructions LD, LDN, LDR and LDF correspond respectively to the open, closed, rising edge and falling edge contacts.



LD %I0.1
ST %Q0.3
LDN %M0
ST %Q0.2
LDR %I0.2
ST %Q0.4
LDF %I0.3
ST %Q0.5

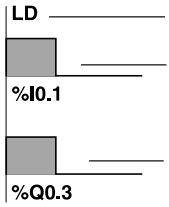
Code	Operand
LD	0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDR	%I
LDF	%I

List of operands

0/1	immediate value 0 or 1
%I	PLC input %Ii.j
%Q	PLC output %Qi.j
%M	internal bit %Mi
%S	system bit %Si
%X	step bit %Xi
%BLK.x	function block bit, e.g. %TMi.Q
%•:Xk	word bit, e.g. %MWi:Xk
[	Comparison expression e.g. [%MWi<1000]

Timing diagram

The 4 timing diagrams have been grouped together.



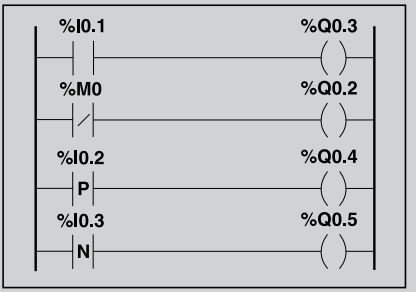
Timing diagram for the LD instruction

Input state

Output state

2.1-3 Load instructions LD, LDN, LDR, LDF

The instructions LD, LDN, LDR and LDF correspond respectively to the open, closed, rising edge and falling edge contacts (LDR and LDF only on PLC inputs).

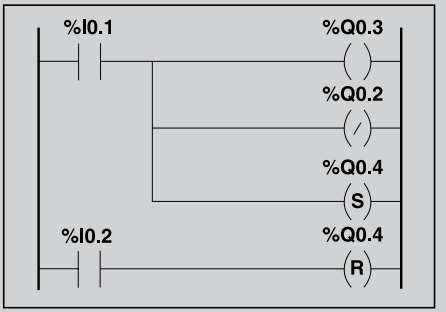


LD %I0.1
ST %Q0.3
LDN %M0
ST %Q0.2
LDR %I0.2
ST %Q0.4
LDF %I0.3
ST %Q0.5

Code	Operand
LD	0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
LDR	%I
LDF	%I

2.1-4 Assignment instructions ST, STN, S, R

The instructions ST, STN, S and R correspond respectively to the direct, inverse, set and reset coils.



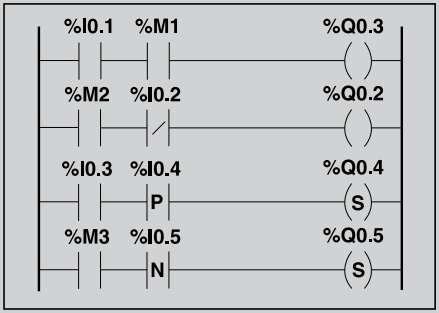
LD %I0.1
ST %Q0.3
STN %Q0.2
S %Q0.4

LD %I0.2
R %Q0.4

Code	Operand
ST	%Q,%M,%S,%BLK.x,%•:Xk
STN	%Q,%M,%S,%BLK.x,%•:Xk
S	%Q,%M,%S,%X,%BLK.x,%•:Xk
R	%Q,%M,%S,%X,%BLK.x,%•:Xk

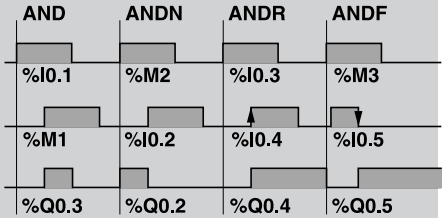
2.1-5 Logic AND instructions : AND, ANDN, ANDR, ANDF

These instructions perform a logic AND between the operand (or its inverse, or its rising or falling edge) and the Boolean result of the preceding instruction.



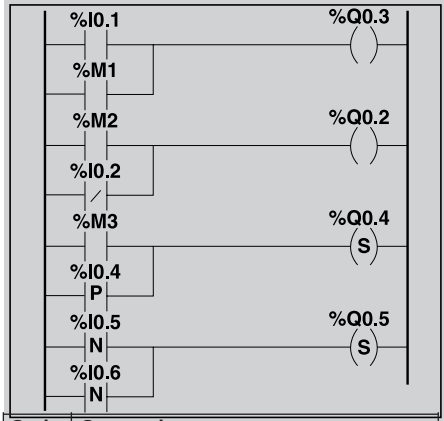

```
LD    %I0.1
AND   %M1
ST    %Q0.3
LD    %M2
ANDN  %I0.2
ST    %Q0.2
LD    %I0.3
ANDR  %I0.4
S     %Q0.4
LD    %M3
ANDF  %I0.5
S     %Q0.5
```

Code	Operand
AND	0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
ANDN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk,[
ANDR	%I
ANDF	%I



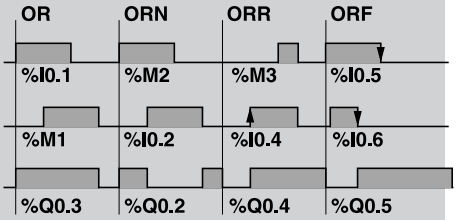
2.1-6 Logic OR instructions : OR, ORN, ORR, ORF

These instructions perform a logic OR between the operand (the inverse of the operand, a rising or falling edge) and the Boolean result of the preceding instruction.



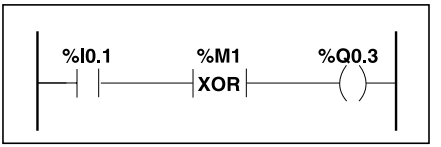

```
LD    %I0.1
OR    %M1
ST    %Q0.3
LD    %M2
ORN   %I0.2
ST    %Q0.2
LD    %M3
ORR   %I0.4
S     %Q0.4
LDF   %I0.5
ORF   %I0.6
S     %Q0.5
```

Code	Operand
OR	0/1,%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
ORN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
ORR	%I
ORF	%I

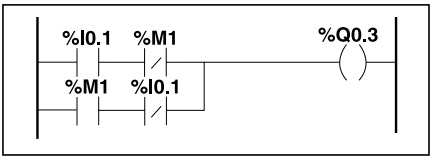


2.1-7 Exclusive OR instructions : XOR, XORN, XORR, XORF

These instructions perform an exclusive OR between the operand (the inverse of the operand, a rising or falling edge) and the Boolean result of the preceding instruction. The XOR instruction can be used in either of the following ways, for example :

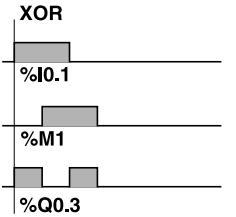



```
LD    %I0.1
XOR   %M1
ST    %Q0.3
```




```
LD %I0.1
ANDN %M1
OR( %M1
ANDN %I0.1
)
ST %Q0.3
```

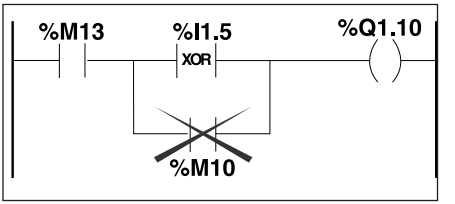
Code	Operand
XOR	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
XORN	%I,%Q,%M,%S,%X,%BLK.x,%•:Xk
XORR	%I
XORF	%I



Special cases

- In Ladder, XOR contacts cannot be
- situated at the left (first position) of a rung,
- placed in parallel.

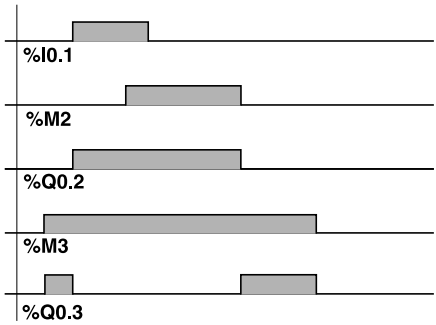
For example, attempting to enter the following rung will generate a Validation error.



2.1-8 NOT instruction : N

This instruction negates the Boolean result of the preceding instruction.

```
LD    %I0.1
OR    %M2
ST    %Q0.2
N
AND   %M3
ST    %Q0.3
```



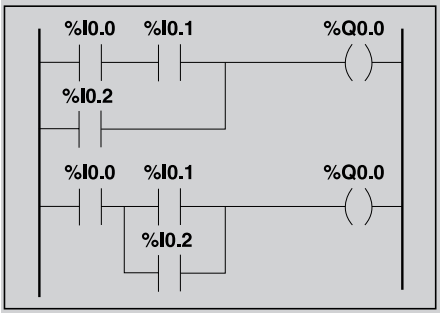
Code	Operand
N	_

Note : The instruction N is not reversible.

2.1-9 Using parentheses

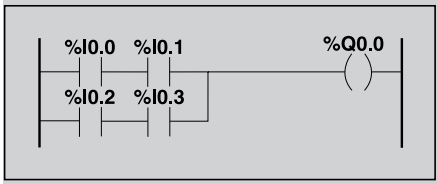
The instructions AND and OR can use parentheses. These parentheses are used for parallel branches in Ladder diagrams. Opening the parenthesis is associated with the AND or OR instruction. Closing the parenthesis is an instruction, which must be given for each open parenthesis.

Example : AND(...)



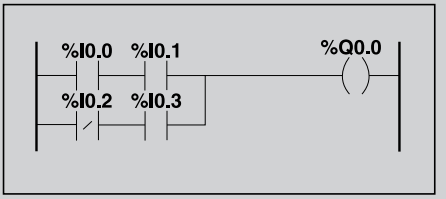

```
LD    %I0.0
AND   %I0.1
OR    %I0.2
ST    %Q0.0

LD    %I0.0
AND(  %I0.1
OR    %I0.2
)
ST    %Q0.0
```

Example : OR(...)




```
LD    %I0.0
AND   %I0.1
OR(   %I0.2
AND   %I0.3
)
ST    %Q0.0
```

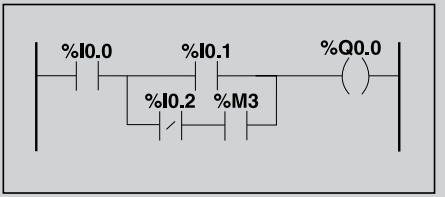
The following modifiers can be assigned to parentheses N, F, R or [:
• N negation, for example, AND(N or OR(N
• R rising edge, for example, AND(R or OR(R
• F falling edge, for example, AND(F or OR(F
• [comparison, see Section 3.1-5




```
LD    %I0.0
AND   %I0.1
OR(N  %I0.2
AND   %I0.3
)
ST    %Q0.0
```

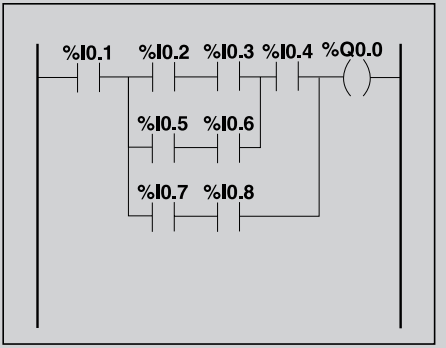
Nesting parentheses
It is possible to nest up to 8 levels of parentheses.

Example




```
LD    %I0.0
AND(  %I0.1
OR(N  %I0.2
AND   %M3
)
)
ST    %Q0.0
```

Example



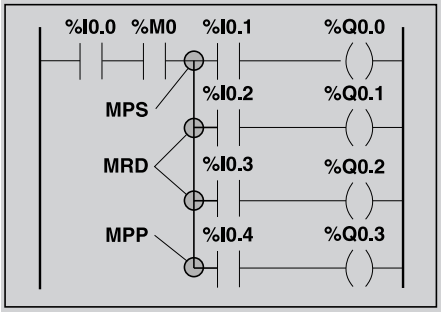

```
LD    %I0.1
AND(  %I0.2
AND   %I0.3
OR(   %I0.5
AND   %I0.6
)
AND   %I0.4
OR(   %I0.7
AND   %I0.8
)
)
ST    %Q0.0
```

Note :
• Each parenthesis opened must be closed again.
• %Li: labels and SRi: subroutines must not be placed in expressions between parentheses. This is also the case for jump (JMP), subroutine (SRi) and function block instructions. See Section 2.4-3 for more information.
• Assignment instructions ST, STN, S and R must not be programmed between parentheses.

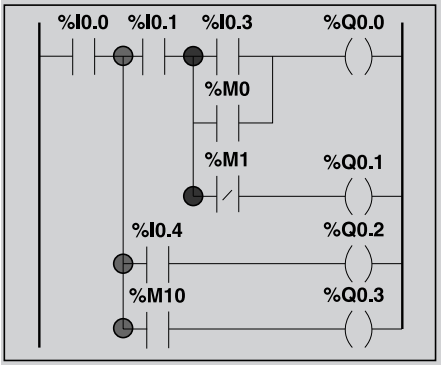
2.1-10 Instructions MPS, MRD, MPP

These 3 types of instruction are used to process the routing to the coils.
They use a temporary storage area referred to as the stack which can store up to 8 Boolean expressions.
The instruction MPS pushes the accumulator (the result of the last logical operation) onto the top of the stack and shifts the other values towards the bottom of the stack.
The instruction MRD reads the top of the stack into the accumulator.
The instruction MPP pops the top of the stack into the accumulator and shifts the other values towards the top of the stack.

Examples :




```
LD    %I0.0
AND   %M0
MPS
AND   %I0.1
ST    %Q0.0
MRD
AND   %I0.2
ST    %Q0.1
MRD
AND   %I0.3
ST    %Q0.2
MPP
AND   %I0.4
ST    %Q0.3
```



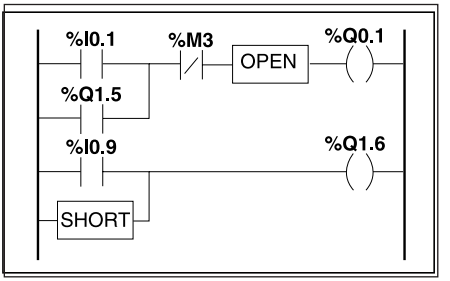

```
LD    %I0.0
MPS
AND   %I0.1
MPS
AND(  %I0.3
OR    %M0
)
ST    %Q0.0
MPP
ANDN  %M1
ST    %Q0.1
MRD
AND   %I0.4
ST    %Q0.2
MPP
AND   %M10
ST    %Q0.3
```

Note : These instructions cannot be used within an expression between parentheses.

2.1-11 Special Ladder instructions OPEN and SHORT

As a special convenience to the Ladder programmer, there are two instructions that can be used during, debugging and troubleshooting. The OPEN and SHORT instructions provide a means to alter the logic of a rung by either "shorting" the continuity of the rung (allow continuity to pass through the rung regardless of the result of the last logical operation), or by "opening" the continuity of the rung (discontinuing the continuity of the rung regardless of the result of the last logical operation).

In list, the OR and AND instructions accompanied by the immediate values of 1 and 0 are used to create the OPEN and SHORT instructions.



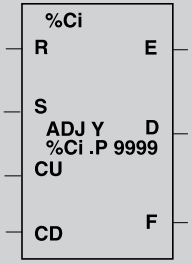

```
LD    %I0.1
OR    %Q1.5
ANDN  %M3
AND   0
ST    %Q0.1
LD    %I0.9
OR    1
ST    %Q1.6
```

2.2 Standard function blocks

2.2-1 Bit and word objects associated with standard function blocks

Function blocks set up bit objects and specific words.

- **Bit objects :**
  These correspond to the block outputs. They can be accessed by Boolean test instructions.
  They can be addressed :
  - either directly (for example, LD E) if they are wired to the block in reversible programming (see Section 2.2-2),
  - or by specifying the block type (for example, LD %Ci.E),
  Inputs can be accessed in the form of instructions.



Up/down counter block

- **Word objects :**
  These correspond to :
  - The block configuration parameters. These parameters can be accessed (for example, preset parameter) or not (for example, time base) by the program.
  - The current values (for example, %Ci.V current count value).

List of function block bit and word objects which can be accessed by the program

Standard function blocks	Associated words and bits		Address	Write access	See Sect.
Timer	Word	Current value	%TMi.V	no	2.2-3
%TMi (i=0 to 31)		Preset value	%TMi.P	yes	
	Bit	Timer output	%TMi.Q	no	
Up/down	Word	Current value	%Ci.V	no	2.2-4
counter		Preset value	%Ci.P	yes	
%Ci (i=0 to 15)	Bit	Underflow output (empty)	%Ci.E	no	
		Preset output reached	%Ci.D	no	
		Overflow output (full)	%Ci.F	no	
LIFO/FIFO register	Word	Access to register	%Ri.I	yes	2.2-5
%Ri (i= 0 to 3)		Register output	%Ri.O	yes	
	Bit	Register output full	%Ri.F	no	
		Register output empty	%Ri.E	no	
Drum controller	Word	Current step number	%DRi.S	yes	2.2-6
%DRi (i=0 to 3)	Bit	Last step equals current step	%DRi.F	yes	

2.2-2 Programming principles

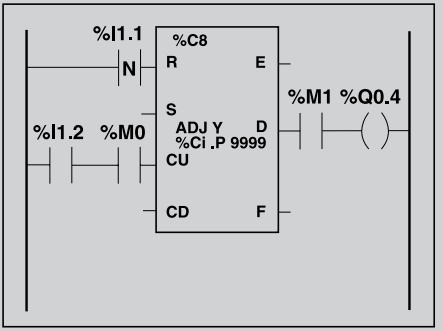
Standard function blocks can be programmed in two different ways :

- Using function block instructions (for example, BLK %TM2): This reversible method of programming for the Ladder language enables operations to be performed on the block in a single place in the program.
- Using specific instructions (for example, CU %Ci): This non-reversible method enables operations to be performed on the block's inputs in several places in the program (for example, line 100 CU %C1, line 174 CD %C1, line 209 LD %C1.D).

Principles of reversible programming of standard function blocks

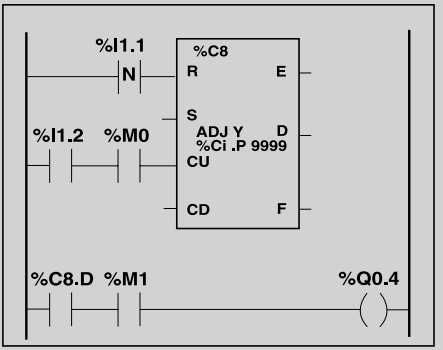
This programming uses instructions BLK, OUT_BLK and END_BLK.
BLK indicates the start of the function block.
OUT_BLK is used to directly "wire" the block outputs.
END_BLK indicates the end of the block.

Example of reversible programming with wired outputs




```
BLK   %C8
LDF   %I1.1
R                    Input processing
LD    %I1.2
AND   %M0
CU
OUT_BLK
LD    D              Output
AND   %M1            processing
ST    %Q0.4
END_BLK
```

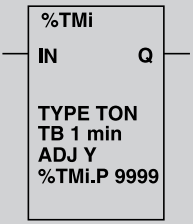
Example of reversible programming without output wiring




```
BLK   %C8
LDF   %I1.1
R                    Input processing
LD    %I1.2
AND   %M0
CU
END_BLK
LD    %C8.D          Output
AND   %M1            processing
ST    %Q0.4
```

Note :
Only test and input instructions on the relevant block can be placed between BLK and OUT_BLK instructions (or between BLK and END_BLK, when OUT_BLK is not programmed).

2.2-3 Timer function blocks %TMi



Timer block

There are 3 types of timer :

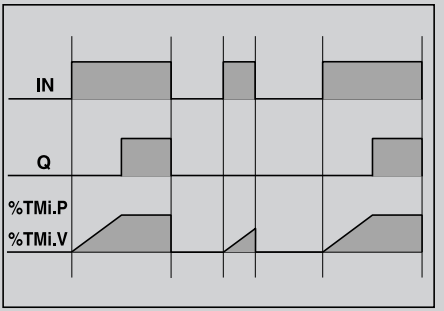
- **TON** : this type of timer is used to control on-delay actions. This delay is programmable and can be modified via the terminal.
- **TOF** : this type of timer is used to control off-delay actions. This delay is programmable and can be modified by the terminal.
- **TP** : this type of timer is used to create a pulse of an exact duration. This duration is programmable and can be modified by the terminal.

Note : The larger the preset value, the greater the timer accuracy.

Characteristics		
Timer number	**%TMi**	0 to 31
Type	**TON** **TOF** **TP**	• on-delay (by default) • off-delay • pulse (monostable)
Time base	**TB**	1min (by default), 1s, 100ms, 10ms, 1ms (for TM0 and TM1).
Current value	**%TMi.V**	Word which increments from 0 to %TMi.P when the timer is running. Can be read and tested but not written by the program (1).
Preset value	**%TMi.P**	0-%TMi.P-9999. Word which can be read, tested and written by the program. It is set to 9999 by default. The length or delay generated is equal to %TMi.P x TB.
Data Editor	**Y/N**	Y : possibility of changing the preset value %TMi.P in the Data editor. N : no access in the Data editor.
Setting input (or instruction)	**IN**	The timer starts on a rising edge (type TON or TP) or a falling edge (type TOF).
Timer output	**Q**	Associated bit %TMi.Q is set to 1 depending on the function performed : TON, TOF or TP

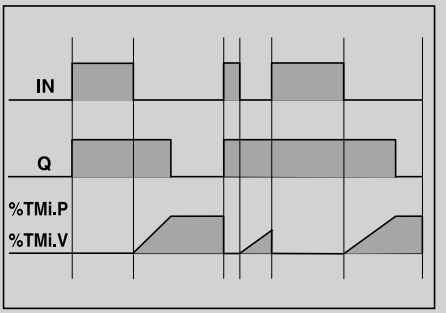
(1) %TMi.V can be modified via the terminal in the Data editor.

Using as an on-delay timer : type TON



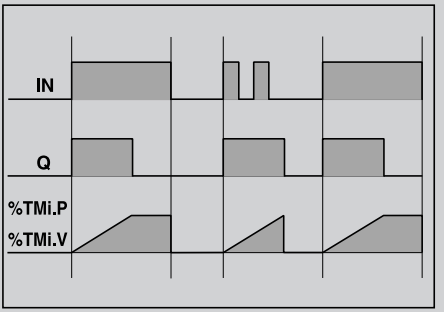
The timer is started on a rising edge at input IN (1) : the current value %TMi.V increases from 0 to %TMi.P by one unit for each pulse of the time base TB. Output bit %TMi.Q changes to 1 when the current value reaches %TMi.P and then remains at 1 as long as a falling edge is not detected at input IN.
When a falling edge is detected at input IN (2), the timer is stopped, even if the timer has not reached % TMi.P.

Using as an off-delay timer : type TOF



The current value %TMi.V is set to 0 on a rising edge at input IN (1) (even if the timer is still running). When the falling edge is detected on input IN, the timer is started. The current value increases to %TMi.P by one unit on each pulse of the time base TB. Output bit %TMi.Q changes to 1 when a rising edge is detected on input IN and the timer returns to 0 when the current value reaches %TMi.P.

Using as a pulse : type TP

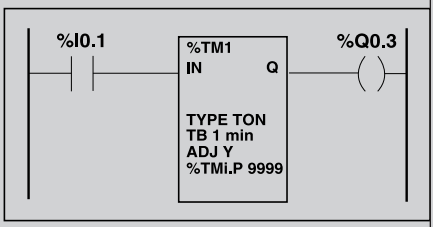


The timer is started on a rising edge at input IN (1) : the current value %TMi.V is set to 0 if the timer has not already started.
The current value increases from 0 to %TMi.P by one unit for each pulse of the time base TB. Output bit %TMi.Q changes to 1 when the timer is started and returns to 0 when the current value reaches %TMi.P.
This timer cannot be reset. Once %TMi.V equals %TMi.P, and the input IN is at state 0, %TMi.V is set to 0.

(1) Or activation of instruction IN.
(2) Or deactivation of instruction IN.

Programming and configuration

Timer function blocks are programmed in the same way irrespective of how they are to be used. The function TON, TOF, and TP is selected during configuration.



Reversible programming

```
BLK   %TM1
LD    %I0.1
IN
OUT_BLK
LD    Q
ST    %Q0.3
END_BLK
```

Configuration

The following parameters must be entered during configuration.

- Type : TON, TOF or TP
- TB : 1min, 1s, 100ms, 10ms or 1ms
- %TMi.P : 0 to 9999
- Adjust : Y or N

Non-reversible programming

```
LD    %I0.1
IN    %TM1
LD    %TM1.Q
ST    %Q0.3
```

Special cases

- **Effect of a cold restart :** (%S0=1)
  - forces the current value to 0,
  - sets output %TMi.Q to 0,
  - the preset value is reset to the value defined during configuration.
- **Effect of a warm restart :** (%S1=1) has no effect on the current value of the timer, nor on the preset value. The current value does not change during a power outage.
- **Effect of a PLC stop :** stopping the PLC does not freeze the current value.
- **Effect of a program jump :** Jumping over a timer block does not freeze the timer. The timer will continue to increment until it reaches the preset value (%TMi.P). At that point, the done bit (%TMi.Q) assigned to output Q of the timer block changes state; however, the associated output wired directly to the block output is not activated and not scanned by the PLC.
- **Testing of bit %TMi.Q (done bit) :** it is advisable to test bit %TMi.Q only once in the program.
- **Effect of master control relay instructions MCS/MCR** : a timer block programmed between 2 MCS/MCR instructions is reset when the MCS instruction is active.
- **Effect of modifying the preset %TMi.P :** modifying the preset value via an instruction or by adjusting it only takes effect when the timer is next activated.

Timers with 1ms time base (TSX 07 3• ••••)

The 1 ms time base is only available on %TM0 and %TM1 timers.
If the user so requires, he can use the four system words %SW76, %SW77, %SW78 and %SW79 as "hourglasses".

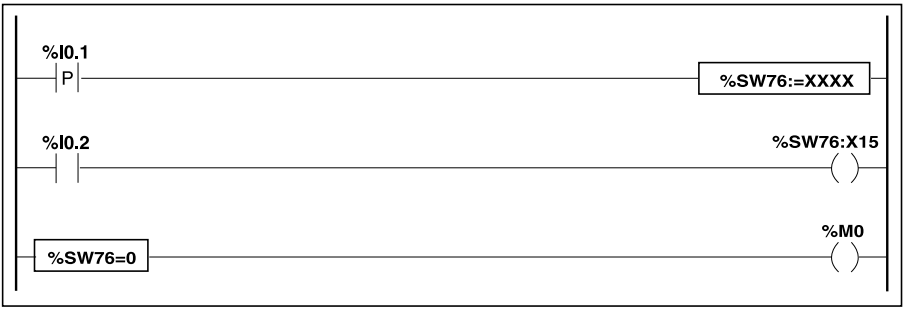
These four system words are decremented individually by the system every millisecond **if they have a positive value**.

Multiple timing can be achieved by successive loading of one of these words or by testing the intermediate values.

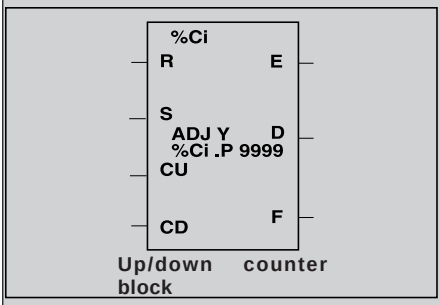
If one of these four system words is less than 0, it will not be modified, a timer can thus be "frozen" by setting corresponding bit 15 to 1 then "unfrozen" by resetting it to 0.

Programming example :

```
LDR%I0.1          (launching the timer on rising edge of %I0.1)
[%SW76:=XXXX]     (XXXX= required value)
LD   %I0.2        (optional management of the freeze, input I0.2 freezes)
ST   %SW76:X15
LD   [%SW76=0]    (timer end test)
ST      %M0
.....
```



2.2-4 Up/down counter function blocks %Ci



The up/down counter function block is used to up/down count events. These two operations can be simultaneous.

Characteristics		
Counter number	**%Ci**	0 to 15
Current value	**%Ci.V**	Word incremented or decremented according to inputs (or instructions) CU and CD. Can be read and tested but not written by the program (1).
Preset value	**%Ci.P**	0-%Ci.P-9999. Word can be read, tested and written (default value: 9999).
Edit via terminal in	**Y/N**	Y : possibility of modifying the preset value the Data editor. N : no access in the Data editor.
Reset input (or instruction)	**R**	At state 1 : %Ci.V = 0.
Set input (or instruction)	**S**	At state 1: %Ci.V = %Ci.P.
Upcount input (or instruction)	**CU**	Increments %Ci.V on a rising edge.
Downcount input (or instruction)	**CD**	Decrements %Ci.V on a rising edge.
Underflow output	**E** (Empty)	The associated bit %Ci.E=1, when downcounter %Ci.V changes from 0 to 9999 (set to 1 when %Ci.V reaches 9999, and reset to 0 if the counter continues to count down).
Preset output reached	**D** (Done)	The associated bit %Ci.D=1, when %Ci.V=%Ci.P.
Overflow output	**F** (Full)	The associated bit %Ci.F =1 when %Ci.V changes from 9999 to 0 (set to 1 when %Ci.V reaches 0, and reset to 0 if the counter continues to count up).

(1) %Ci.V can be modified via the terminal in the Data editor.

Operation

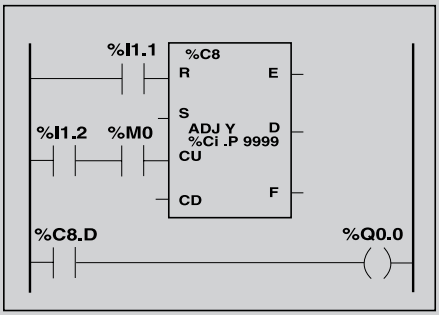
- **Upcount :** when a rising edge appears at the upcounting input CU (or instruction CU is activated), the current value is incremented by one unit. When this value is equal to the preset value %Ci.P, the "preset reached" output bit %Ci.D assigned to output D changes to state 1. Output bit %Ci.F (upcount overflow) changes to state 1 when %Ci.V changes from 9999 to 0, and is reset to 0 if the counter continues to count up.
- **Downcount :** when a rising edge appears at the downcounting input CD (or instruction CD is activated), the current value %Ci.V is decremented by one unit. Output bit %Ci.E (underflow) changes to state 1 when %Ci.V changes from 0 to 9999, and is reset to 0 if the counter continues to count down.
- **Up/downcount :** to use both the upcount and the downcount functions simultaneously (or to activate instructions CD and CU), the two corresponding inputs CU and CD must be controlled. These two inputs are then scanned in succession. If they are both at 1, the current value remains unchanged.
- **Reset :** when this input is set to state 1 (or the instruction is activated), the current value %Ci.V is forced to 0, outputs %Ci.E, %Ci.D and %Ci.F are at 0. The reset input has priority.
- **Set :** if input S is at state 1 (or instruction S is active) and the reset input is at state 0 (or instruction R is inactive), the current value %Ci.V takes the value of %Ci.P, and output %Ci.D is set to 1.

Special cases

- **Effect of a cold restart :** (%S0=1)
  - The current value %Ci.V is set to zero.
  - Output bits %Ci.E, %Ci.D and %Ci.F are set to zero.
  - The preset value is initialized with the value defined during configuration.
- **Effect of a warm restart (%S1=1), of a PLC stop :** this has no effect on the current value of the counter (%Ci.V).
- **Effect of modifying the preset %Ci.P :** modifying the preset value via an instruction or by adjusting it takes effect when the block is processed by the application (activation of one of the inputs).

Configuration and programming

Example : Count a number of items up to 5000. Each pulse on input %I1.2 (when internal bit %M0 is at 1) increments the upcounter %C8 up to its final preset value (bit %C8.D=1). The counter is reset by input %I1.1.



Configuration

The following parameters must be entered during configuration :

- %Ci.P, set to 5000 in this example
- Adjust : Y

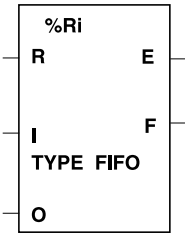
Reversible programming

```
BLK   %C8
LD    %I1.1
R
LD    %I1.2
AND   %M0
CU
END_BLK
LD    %C8.D
ST    %Q0.0
```

Non-reversible programming

```
LD    %I1.1
R     %C8
LD    %I1.2
AND   %M0
CU    %C8
LD    %C8.D
ST    %Q0.0
```

2.2-5 LIFO/FIFO register function blocks %Ri



Register block

A register is a memory block which is used to store up to 16 words of 16 bits in two different ways :

- Queue (First In, First Out) known as FIFO.
- Stack (Last In, First Out) known as LIFO.

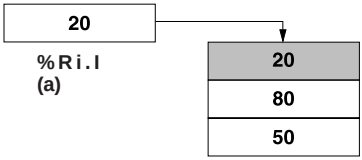
Characteristics		
Register number	**%Ri**	0 to 3
Type	**FIFO** **LIFO**	Queue (default selection) Stack
Input word	**%Ri.I**	Register input word. Can be read, tested and written.
Output word	**%Ri.O**	Register output word. Can be read, tested and written.
Storage input (or instruction)	**I** (In)	On a rising edge, stores the contents of word %Ri.I in the register.
Retrieval input (or instruction)	**O** (Out)	On a rising edge, loads a data word into word
Reset input (or instruction)	**R** (Reset)	%Ri.O. At state 1 initializes the register.
Empty output	**E** (Empty)	The associated bit %Ri.E indicates that the register is empty. Can be tested.
Full output	**F** (Full)	The associated bit %Ri.F indicates that the register is full. Can be tested.

Operation

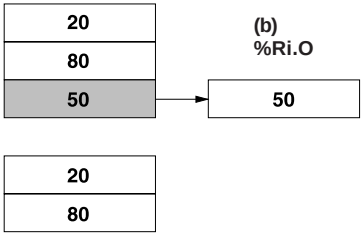
FIFO (First In, First Out)

The first data item entered is the first to be retrieved.
When a storage request is received (rising edge at input I or activation of instruction I), the contents of input word %Ri.I (which have already been loaded) are stored at the top of the queue (Fig a).
When the queue is full (output F=1), no further storage is possible.
When a retrieval request is received (rising edge at input O or activation of instruction O) the data word lowest in the queue is loaded into output word %Ri.O and the contents of the register are moved down one place in the queue (Fig b).
When the register is empty (output E=1) no further retrieval is possible. Output word %Ri.O does not change and retains its value.
The queue can be reset at any time (state 1 at input R or activation of instruction R).

Example :
Storage of the contents of %Ri.I at the top of the queue.



Retrieval of the first data item which is then loaded into %Ri.O

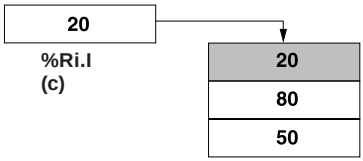


LIFO (Last In, First Out)

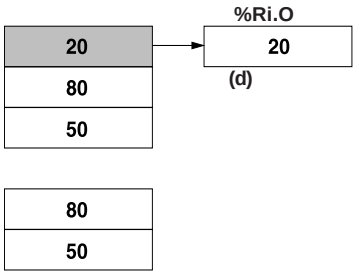
The last data item entered is the first to be retrieved.

When a storage request is received (rising edge at the input or activation of instruction I), the contents of input word %Ri.I (which have already been loaded) are stored at the top of the stack (Fig c).
When the stack is full (output F at 1), no further storage is possible.
When a retrieval request is received (rising edge at input O or activation of instruction O) the highest data word (last word to be entered) is loaded into word %Ri.O (Fig d).
When the register is empty (output E=1), no further retrieval is possible. Output word %Ri.O does not change and retains its last value. The stack can be reset at any time (state 1 at input R or activation of instruction R). The element indicated by the pointer is then the highest in the stack.

Example :
Storage of the contents of %Ri.I at the top of the stack.



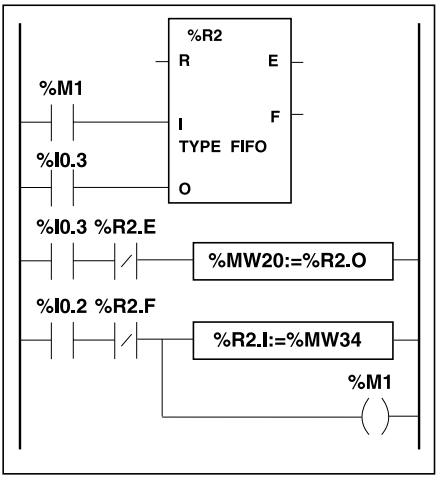
Retrieval of the data word highest in the stack.



Special cases

- **Effect of a cold restart** : (%S0=1) initializes the contents of the register Output bit %Ri.E, assigned to output E, to 1 and resets %Ri.I and %Ri.O words to 0.
- **Effect of a warm restart :** (%S1=1) has no effect on the contents of the register, nor on the state of its output bits.

Programming and configuration



The programming example shows word %MW34 being loaded into %R2.I at the storage request %I0.2, if register R2 is not full (%R2.F=0). The storage request in the register is made by %M1. The retrieval request is made by input %I0.3 and %R2.O is loaded into %MW20 if the register is not empty (%R2.E=0).

Configuration

The only parameter which must be entered during configuration is the type of register : FIFO (default) or LIFO.

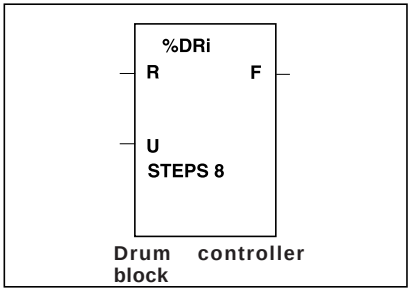
Reversible program

```
BLK    %R2
LD     %M1
I
LD     %I0.3
O
END_BLK
LD     %I0.3
ANDN   %R2.E
[%MW20:=%R2.O]
LD     %I0.2
ANDN   %R2.F
[%R2.I:=%MW34]
ST     %M1
```

Non-reversible program

```
LD     %M1
I      %R2
LD     %I0.3
O      %R2
ANDN   %R2.E
[%MW20:=%R2.O]
LD     %I0.2
ANDN   %R2.F
[%R2.I:=%MW34]
ST     %M1
```

2.2-6 Drum controller function block %DRi



The drum controller operates on a similar principle to an electromechanical drum controller, which changes step according to external events. On each step, the high point of a cam gives a command which is executed by the control system. In the case of a drum controller, these high points are symbolized by state 1 for each step and are assigned to output bits %Qi.j or internal bits %Mi, known as control bits.

Characteristics		
Number	**%DRi**	0 to 3
Current step number	**%DRi.S**	0-%DRi.S-7. Word which can be read and tested. Can only be written in the program with a decimal immediate value.
Number of steps		1 to 8 (default)
Return to step 0 input (or instruction)	**R** (RESET)	At state 1, sets the drum controller to step 0.
Advance input (or instruction)	**U** (UP)	On a rising edge, causes the drum controller to advance by one step and updates the control bits.
Output	**F** (FULL)	Indicates that the current step equals the last step defined. The associated bit %DRi.F can be tested (for example, %DRi.F=1 if %DRi.S= number of steps configured - 1).
Control bits		Outputs or internal bits associated with the step (16 control bits) and defined in the Configuration editor.

Operation

The drum controller comprises :

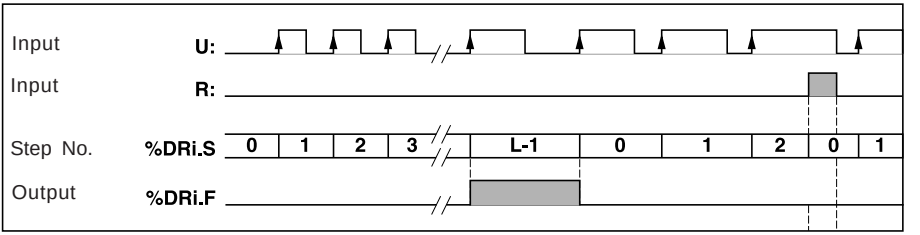
- A matrix of constant data (the cams) organized in 8 steps (0 to 7) and 16 data bits (state of the step) arranged in columns numbered 0 to F.
- A list of control bits (1 per column) corresponding either to outputs %Q0.i or %Q1.i, or to internal bits %Mi. During the current step, the control bits take on the binary states defined for this step.

The table below summarizes the main characteristics of the drum controller.

Column Control bits	0 %Q0.1	1 %Q0.3	2 %Q1.5		D %Q0.6	E %Q0.5	F %Q1.0
Step 0	0	0	1		1	1	0
Step 1	1	0	1		1	0	0
Step 5	1	1	1		0	0	0
Step 6	0	1	1		0	1	0
Step 7	1	1	1		1	0	0

In the example above, step 5 is the current step, control bits %Q0.1; %Q0.3 and %Q1.5 are set to state 1; control bits %Q0.6; %Q0.5 and %Q1.0 are set to state 0.
The current step number is incremented on each rising edge at input U (or on activation of instruction U). The current step can be modified by the program.

Operating diagram

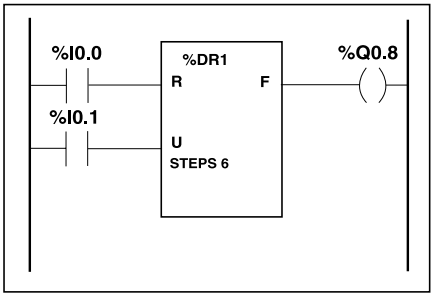


Special cases

- **Effect of a cold restart :** (%S0=1) resets the drum controller to step 0 (updating the control bits).
- **Effect of a warm restart :** (%S1=1) updates the control bits, after the current step.
- **Effect of a program jump :** If the drum controller is not scanned, the control bits are not reset to 0.
- **Updating the control bits :** only occurs when there is a change of step or in the case of a warm or cold restart.
- **Effect of master control relay instructions MCS/MCR :** Having a drum controller between 2 MCS/MCR instructions means that the control bits are reset to 0 if the Boolean result of the instruction preceding the MCS instruction is 0.

Programming and configuration

In this example, the first 6 outputs %Q0.0 to %Q0.5 are activated in succession, each time input %I0.1 is set to 1.
Input I0.0 resets the outputs to 0.



Reversible programming

```
BLK   %DR1
LD    %I0.0
R
LD    %I0.1
U
OUT_BLK
LD    F
ST    %Q0.8
END_BLK
```

Configuration

The following information is defined during configuration :

- The number of steps : 6
- The output states (control bits) for each drum controller step.

  Q0. 1: 2: 3: 4: 5:
  Step 1 : 0 0 0 0 0
  Step 2 : 1 0 0 0 0
  Step 3 : 0 1 0 0 0
  Step 4 : 0 0 1 0 0
  Step 5 : 0 0 0 1 0
  Step 6 : 0 0 0 0 1

- Assignment of the control bits

  1 : %Q0.0 4 : %Q0.1
  2 : %Q0.2 5 : %Q0.3
  3 : %Q0.4 6 : %Q0.5

Non-reversible programming

```
LD    %I0.0
R     %DR1
LD    %I0.1
U     %DR1
LD    %DR1.F
ST    %Q0.8
```

2.3 Grafcet instructions

2.3-1 Description

B

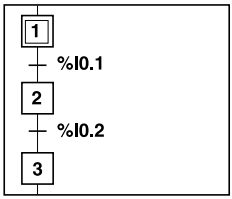
Grafcet instructions in PL7 language offer a simple method of translating a control sequence (Grafcet chart).
The PL7 language comprises a maximum of 62 steps, including the initial step(s). The number of steps active at any one time is limited only by the total number of steps.
The table below lists all the instructions and objects required to program a Grafcet chart.

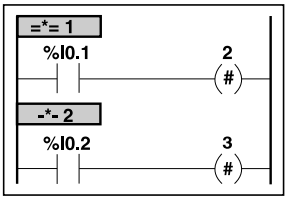
Graphic representation	Transcription in PL7 language	Role
 Initial step Transition Step	**=*= i**	Start the initial step (1)
	# i	Activate step i after deactivating the current step
	-*- i	Start step i and validate the associated transition (1)
	#	Deactivate the current step without activating any other steps
	#Di	Deactivate step i and the current step
	=*= POST	Start post-processing and end sequential processing
	%Xi	Bit associated with step i, can be tested and written (max. no. of steps : 62).
	LD %Xi, AND%Xi, ANDN%Xi OR%Xi, XOR%Xi, XORN%Xi	LDN %Xi, Test the activity of step i ORN%Xi
	S **%Xi**	Activate step i
	R **%Xi**	Deactivate step i

(1) The first step =*=i or -*-i written indicates the start of sequential processing and thus the end of preprocessing.

Examples

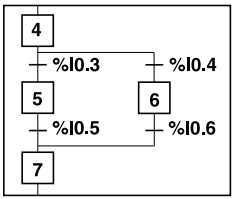
Linear sequence

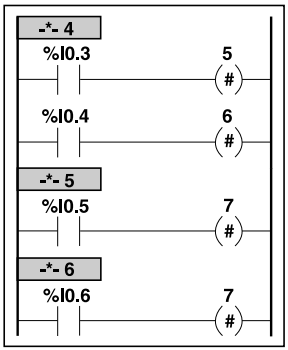





```
=*=        1
LD %I0.1
#  2
-*- 2
LD %I0.2
#  3
```

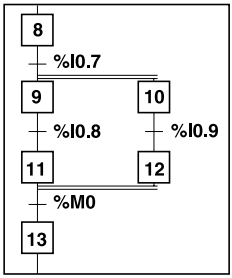
Alternative sequence

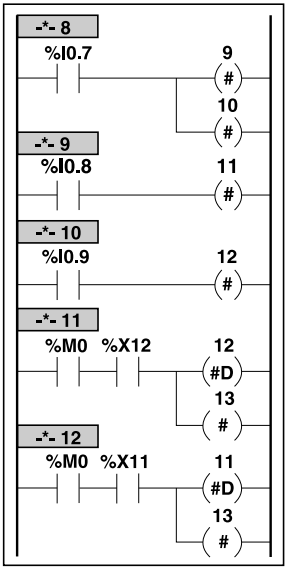





```
-*- 4
LD %I0.3
#  5
LD %I0.4
#  6
-*- 5
LD %I0.5
#  7
-*- 6
LD %I0.6
#  7
```

Simultaneous sequences






```
-*- 8
LD %I0.7
#  9
#  10
-*- 9
LD %I0.8
#  11
-*- 10
LD %I0.9
#  12
-*- 11
LD %M0
AND %X12
#D 12
#  13
-*- 12
LD %M0
AND %X11
#D 11
#  13
```

Note :

For a Grafcet Chart to be operational, at least one active step must be declared using the =*=i instruction (initial step) or the chart should be prepositioned during preprocessing using system bit %S23 and the instructions S %Xi.
See example of prepositioning a chart in appendix A.10 part G.

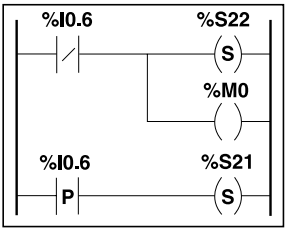
2.3-2 Program structure

A PL7 Grafcet program has 3 parts, each of which has a specific role.

Processing	Example

Preprocessing

This comprises a series of instructions for processing :
- power returns
- faults
- changes of operating mode
- prepositioning Grafcet steps
- input logic



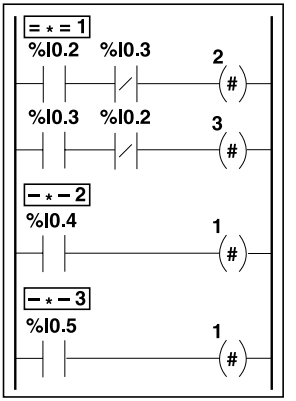

```
000  LDN   %I0.6
001  S     %S22
002  ST    %M0
003  LDR   %I0.6
004  S     %S21
```

Preprocessing ends with the first =*= or -*- instruction which occurs.

Sequential processing

This comprises the chart (instructions representing the chart) :
- steps
- actions associated with steps (see Appendix A.11 Part G)
- transitions
- transition conditions

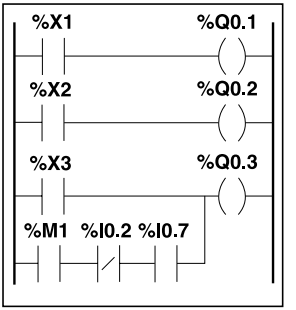
Sequential processing ends with execution of the =*=POST instruction




```
005  =*=   1
006  LD    %I0.2
007  ANDN  %I0.3
008  #     2
009  LD    %I0.3
010  ANDN  %I0.2
011  #     3
012  -*-   2
013  LD    %I0.4
014  #     1
015  -*-   3
016  LD    %I0.5
017  #     1
```

Post-processing

This comprises a series of instructions for processing :
- commands from the sequential processing for controlling the outputs
- safety interlocks specific to the outputs.




```
018  =*=   POST
019  LD    %X1
020  ST    %Q0.1
021  LD    %X2
022  ST    %Q0.2
023  LD    %X3
024  OR(   %M1
025  ANDN  %I0.2
026  AND   %I0.7
027  )
028  ST    %Q0.3
```

Note :
The scan cycle is that defined in Part A, Section 1.3. In sequential processing, only active steps at the start of the scan and their associated instructions are executed.

2.4 Program instructions

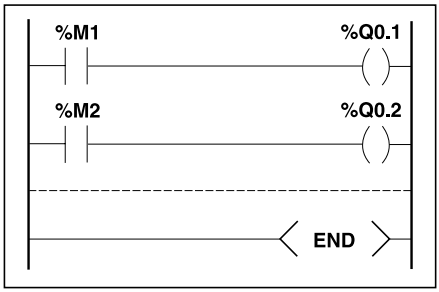
2.4-1 Program end instructions END, ENDC, ENDCN

The end of the execution of a program scan is defined using the instructions END, ENDC and ENDCN :

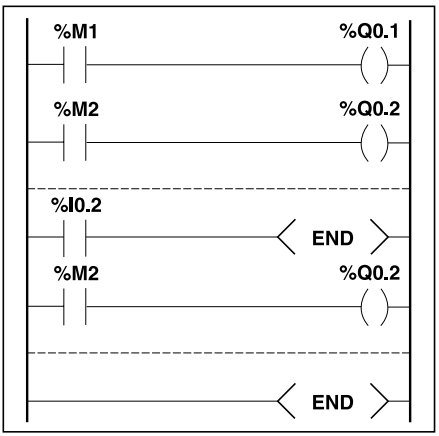
END : unconditional end of program
ENDC : end of program if Boolean result of preceding test instruction is 1
ENDCN : end of program if Boolean result of preceding test instruction is 0.

By default (normal mode) when the end of program is activated, the outputs are updated and the next scan is started.
If scanning is periodic, when the end of period is reached the outputs are updated and the next scan is started.

Example :




```
LD  %M1
ST  %Q0.1
LD  %M2
ST  %Q0.2
.....
END
```




```
LD  %M1
ST  %Q0.1
LD  %M2
ST  %Q0.2
.....
LD  %I0.2
ENDC        ⟶ If %I0.2 =1, end of program scanning.
LD  %M2
ST  %Q0.2
.....
END         If %I0.2 =0, continues program scanning until new END instruction.
```

2.4-2 NOP Instruction

The **NOP** instruction does not perform any operation. It is used for "reserving" lines in a program which allow the user to insert instructions later without modifying the line numbers.

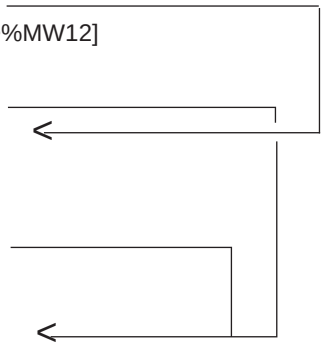
2.4-3 Jump instructions JMP, JMPC, JMPCN to a label %Li:

Instructions JMP, JMPC and JMPCN cause the execution to be interrupted immediately and the program to be continued from the line after the program line containing label %Li (i = 0 to 15).

JMP : unconditional program jump
JMPC : program jump if Boolean result of preceding logic is 1
JMPCN : program jump if Boolean result of preceding logic is 0.

Examples :

```
000   LD     %M15
001   JMPC %L8
002   LD     [%MW24>%MW12]
003   ST %M15
004   JMP   %L12
005   %L8:
006   LD  %M12
007   AND   %M13
008   ST     %M2
009   JMPCN  %L12
010   OR     %M11
011   S      %Q0.0
012   %L12:
013   LD %I0.0
.....
```



Jump to label %L8 if %M15 is at 1

Unconditional jump to label %L12 :

Jump to label %L12 if %M2 is at 0

Note :

• These instructions are not permitted between parentheses, and they must not be placed between the instructions AND(, OR(, and a close parenthesis instruction ")".

• The label can only be placed before a LD, LDN, LDR, LDF or BLK instruction.

• The label number of label %Li must be defined only once in a program.

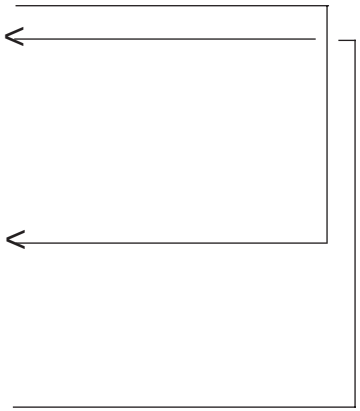
• The program jump is performed to a line of programming which is downstream or upstream. When the jump is upstream, attention must be paid to the program scan time. Extended scan time can cause the watchdog timer to expire.

2.4-4 Subroutine instructions SRn, SRn:, RET

The **SRn** instruction calls the subroutine referenced by label **SRn**: if the result of the preceding Boolean instruction is 1.
The **RET** instruction placed at the end of the subroutine commands the return to the main program.
The subroutine is referenced by a label SRn: with n=0 to 15.

Example :

```
000   LD     %M15
001   AND    %M5
002   ST     %Q0.0
003   LD     [%MW24>%MW12]
004   SR8
005   LD     %I0.4
006   AND    %M13
007   _
008   _
009   _
010   END

011   SR8:
012   LD  1
013   IN     %TM0
014   LD     %TM0.Q
015   ST     %M10
016   RET
.....
```



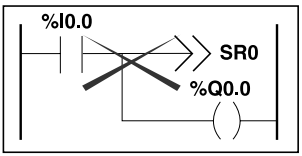
Note :

• A subroutine should not call up another subroutine.

• This instruction is not permitted between parentheses, and thus it must not be placed between the instructions AND(, OR(, and a close parenthesis instruction ")".

• The label can only be placed before a LD or BLK instruction marking the start of a Boolean equation (or rung).

• Calling the subroutine must not be followed by an assignment (e.g. output or transfer) instruction, example :



~~LD %I0.0~~
~~SR0~~
~~ST %Q0.0~~

LD %I0.0
ST %Q0.0
SR0

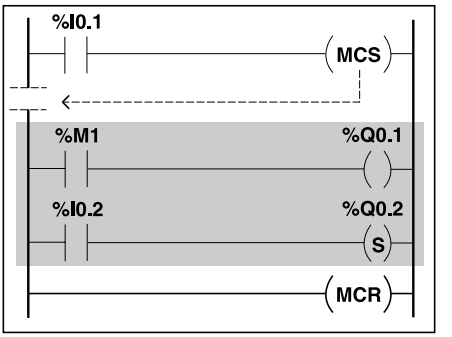
2.4-5 Master control relay instructions MCR and MCS

When the Boolean result of the instruction preceding the MCS instruction is 0, the execution of the program lines following this instruction are modified as shown in the table below until the MCR instruction is (unconditional) executed.

Instructions / blocks	Behavior
ST, STN	Associated object set to 0
S, R	Instructions not executed
SRi, JMP, JMPC, JMPCN	Not executed
%TMi	Reinitialized
%DRi	Control bits set to 0
%FC	Counter stopped and threshold outputs reset to 0
% PWM, %PLS	Generation of output signals stopped
Other function blocks	Not executed (maintain their current state)
Operation blocks	Not executed

Example :

```
.....
002   LD    %I0.1
003   MCS
004   LD    %M1
005   ST    %Q0.1
006   LD    %I0.2
007   S     %Q0.2
007   MCR
.....
```



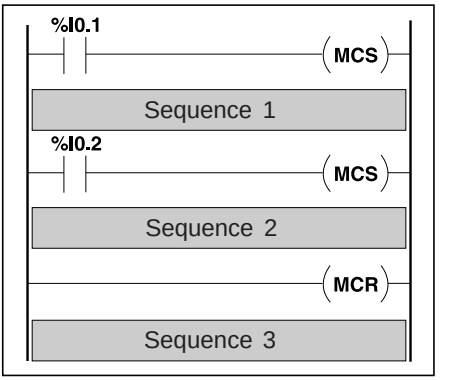
When %I0.1 is at 0, the MCS instruction is activated, %Q0.1 is forced to 0, and output %Q0.2 is maintained.

It is possible to use several MCS instructions for a single MCR instruction. All the MCS instructions are deactivated by a single MCR instruction.

When %I0.1 is at 0, sequences 1 and 2 are modified, sequence 3 is executed normally.

When %I0.1 is at 1 and %I0.2 is at 0, sequence 2 is modified, and sequences 1 and 3 are executed normally.

If no MCR instruction is programmed after an MCS instruction, the MCS instruction is effective until the END instruction or the end of the program.



Important

Do not use MCS and MCR instructions in subroutines, transition conditions or Grafcet actions.

B

3.1 Numerical processing

3.1-1 Definition of word objects

Word objects are addressed in the form of words of 16 bits, which are stored in the data memory, and which can contain an integer value between -32768 and 32767 (except for the fast counter which is between 0 and 65535).

Immediate values

These are integer values of the same format as the 16-bit words, which enable values to be assigned to these words. They are stored in the program memory and contain a value between -32768 and 32767.

Word formats

The contents of the words or values are stored in the user memory in 16-bit binary code (two's complement) using the following convention :

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
	0	1	0	1	0	0	1	0	0	1	0	0	1	1	0	1	Bit position Bit state
I+	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1		Bit value

In signed binary notation, bit 15 is allocated by convention to the sign of the coded value :

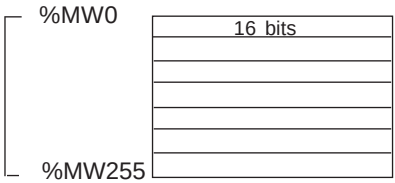
- Bit 15 at 0 : the content of the word is a positive value.
- Bit 15 at 1 : the content of the word is a negative value (negative values are expressed in two's complement logic).

Words and immediate values can be entered or retrieved in the following format :

- **Decimal** 1579 (max : 32767, min :-32768)
- **Hexadecimal** 16#A536 (max : 16#FFFF, min : 16#0000)
  alternate syntax : #A536.

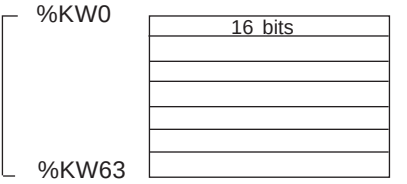
Internal words

Internal words are used to store values during the operation.
They are stored in the data memory.
Words %MW0 to %MW255 are read or written to directly by the program. They are used as working words.



Constant words

Constant words store constant values or alphanumeric messages. Their contents can only be written or modified by the terminal (in the CONFIGURATION editor). Constant words %KW0 - %KW63 have read-only access by the program.



I/O exchange words

I/O exchange words %IW/QW are assigned to PLCs connected by extension cable. They are used for exchanges between PLCs (see Section 3.5).

System words

These 16-bit words have several functions. They provide access to data coming directly from the PLC by reading %SWi words (for example, potentiometers). They are used to perform operations on the application (for example, adjusting the schedule blocks). The role of each of these words is explained in Section 6.

Bit objects extracted from words

It is possible to extract one of the 16 bits from some words. The position of the extracted bit separated by a colon is added to the word address.

Syntax : % Word object : Xk where k = 0 to 15 position of the bit in the object word.
Example : %MW5:x6 : bit position 6 of internal word %MW5.

List of word operands

Type	Address (or value)	Maximum number	Write access	See Sect.
Immediate values • base 10 • base 16	 example : 2103 example : 16#AF0D		No	
Internal words	%MWi	256	Yes	_
Constant words	%KWi	64	No (1)	_
System words	%SWi	128	Depends on i	5.2
Function block words	%TMi.P %Ci.P etc.			2.2-1 3.3
Exchange words Input words Output words	 %IWi.j %QWi.j	 8 8	 No Yes	3.5
Bits extracted from words				
• internal	%MWi:Xk	256 x 16	Yes	
• system	%SWi:Xk	128 x 16	Depends on i	
• constant	%KWi:Xk	64 x 16	No	
• input	%IWi.j:Xk	8 x 16	No	
• output	%QWi.j:Xk	8 x 16	Yes	

(1) constant words are entered in configuration mode.

3.1-2 Structured objects

Bit string

Bit strings are series of adjacent object bits of the same type and of a defined length : L.

Example of bit strings :

%M8 %M9 %M10 %M11 %M12 %M13

%M8:6 (1)

Type	Address	Maximum size	Write access
Discrete input bits	%I0:L or %I1:L	0<L<17	No
Discrete output bits	%Q0:L or %Q1:L	0<L<17	Yes
System bits	%Si:L with i multiple of 8	0<L<17 and i+L-128	Depending on i
Grafcet Step bits	%Xi:L with i multiple of 8	0<L<17 and i+L-63	Yes (by program)
Internal bits	%Mi:L with i multiple of 8	0<L<17 and i+L-128	Yes

Bit strings can be used with the assignment instruction := (see assignment instruction, Section 3.1-4).

Word tables

Word tables are series of adjacent words of the same type and of a defined length : L.

Example of word tables :

%KW10:7

%KW10

%KW16

16 bits

Type	Address	Maximum size	Write access
Internal words	%MWi:L	0<L<256 and i+L-256	Yes
Constant words	%KWi:L	0<L and i+L-64	No
System words	%SWi:L	0<L and i+L-128	Depending on i

Word tables can be used with the assignment instruction := (see assignment instruction, Section 3.1-4).

(1) %M8:6 is acceptable (8 is a multiple of 8)
%M10:16 is unacceptable (10 is not a multiple of 8)

Indexed words

- **Direct addressing**

The addressing of an object is said to be direct when the address of the objects is fixed and defined when the program is written.

Example : %M26 (internal bit with the address 26)

- **Indexed addressing**

In indexed addressing, an index is added to the direct address of an object : the content of the index is added to the object address. The index is defined by an internal word %MWi. The number of "index words" is unlimited.

Example : %MW108[%MW2] : word with direct address 108 + contents of word %MW2.
If word %MW2 has a value of 12, writing to %MW108[%MW2] is equivalent to writing to %MW120.

Type	Address	Maximum size	Write access
Internal words	%MWi[%MWj]	0-i+%MWj<256	Yes
Constant words	%KWi[%MWj]	0-i+%MWj<64	No

Indexed words can be used with the assignment instruction := (see assignment instruction, Section 3.1-4) and in comparison instructions (see comparison instructions Section 3.1-5).

This type of addressing enables series of objects of the same type (internal words, constant words, etc) to be scanned in succession, by modifying the content of the index word via the program.

- **Index overflow, system bit %S20**

There is an overflow of the index when the address of an indexed object exceeds the limits of the memory zone containing the same type of object, that is to say when :
- The object address + the content of the index is less than 0,
- The object address + the content of the index is greater than 255 (for words %MWi) or 63 (for words %KWi).

In the event of an index overflow, the system sets system bit %S20 to 1 and the object is assigned an index value of 0.

Note : The user is responsible for monitoring any overflow : bit %S20 must be read by the user program for possible processing. The user must see that it is reset to 0.

%S20 (initial state = 0) :
- On index overflow : set to 1 by the system.
- Acknowledgment of overflow : set to 0 by the user, after modifying the index.

B

3.1-3 Introduction to numerical instructions

Numerical instructions generally apply to 16-bit words (see Section 3.1-1). They are written between square brackets. If the result of the preceding logical operation was true (the Boolean accumulator = 1), the numerical instruction is executed. If the result of the preceding logical operation was false (the Boolean accumulator = 0), the numerical instruction is not executed and the operand remains unchanged.

Note : Numerical instructions are displayed in 2 or 3 lines on an FTX117 terminal.

3.1-4 Assignment instructions

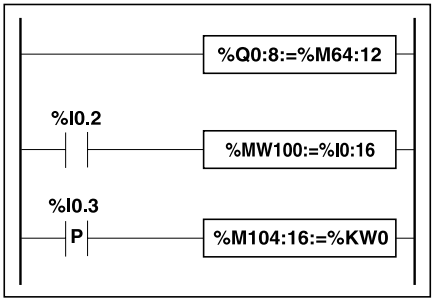
These are used to load operand Op2 into operand Op1

Syntax : [Op1:=Op2] <=> Op2->Op1

The following assignment operations can be performed :
- On bit strings
- On words
- On word tables

Assignment of bit strings (see bit string object in Section 3.1-2)
Operations can be performed on the following bit strings :
bit string -> bit string (ex 1)
bit string -> word (ex 2)
word -> bit string (ex 3)
immediate value -> bit string

Examples




```
LD     1
[%Q0:8:= %M64:12]          (ex 1)

LD     %I0.2
[%MW100:= %I0:16 ]         (ex 2)

LDR    %I0.3
[%M104:16:=%KW0]           (ex 3)
```

Usage rules
- For bit string -> word assignment : the bits in the string are transferred to the word starting on the right (first bit in the string to bit 0 in the word), and the word bits which are not involved in the transfer (length<16) are set to 0.
- For word -> bit string assignment : the word bits are transferred from the right (word bit 0 to the first bit in the string).

Syntax

Operator	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
:=	[Op1: = Op2] Operand 1 (Op1) assumes the value of operand 2 (Op2)	%MWi,%QWi, %SWi %MWi[MWi], **%Mi:L,%Qi:L,%Si:L,** **%Xi:L**	Immediate value, %MWi, %KWi,%IW,%QW,%SWi, %BLK.x,%MWi[MWi], %KWi[MWi]**, %Mi:L,%Qi:L,** **%Si:L,%Xi:L, %Ii:L**

Note : The abbreviation %BLK.x (e.g. %C0.P) is used to describe any function block word.

Assignment of words

Assignment operations can be performed on the following words :

word -> word (ex 1)
indexed word -> word
immediate value -> word (ex 3)
bit string -> word
word -> indexed word
indexed word -> indexed word (ex 2)
immediate value -> indexed word
word -> bit string

Examples :


```
LD    1
[%SW112 := %MW100]          (ex 1)

LD    %I0.2
[%MW0[%MW10] :=
      %KW0[%MW20] ]         (ex 2)

LDR   %I0.3
[%MW10:=100]                (ex 3)
```

Syntax

Operator	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
:=	[Op1: = Op2] Operand 1 (Op1) assumes the value of operand 2 (Op2)	**%MWi,%QWi, %SWi** **%MWi[MWi],** %Mi:L,%Qi:L,%Si:L, %Xi:L	**Immediate value, %MWi**, **%KWi, %IW, %QW, %SWi,** **%BLK.x, %MWi[MWi],** **%KWi[MWi]**, %Mi:L,%Qi:L, %Si:L,%Xi:L,%Ii:L

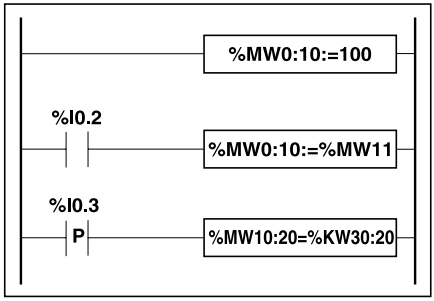
Notes :

• The abbreviation %BLK.x (e.g. R3.I) is used to describe any function block word.
• For bit strings %Mi:L, %Si:L and %Xi:L, the base address of the first of the bit string must be a multiple of 8 (0, 8, 16, …, 96, …).

Assignment of word tables (see word table object in Section 3.1-2)

Assignment operations can be performed on the following word tables :
immediate value -> word table (ex 1)
word -> word table (ex 2)
word table -> word table (ex 3)

Examples




```
LD      1
[%MW0 :10:= 100]                (ex 1)

LD      %I0.2
[%MW0:10 := %MW11]              (ex 2)

LDR     %I0.3
[%MW10:20=%KW30:20]             (ex 3)
```

Syntax

Operator	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
:=	[Op1: = Op2] Operand 1 (Op1) assumes the value of operand 2 (Op2)	**%MWi:L,%SWi:L**	**%MWi:L, %KWi:L, %SWi:L** Immediate value, %MWi, %KWi, %IW, %QW, %SWi, %BLK.x

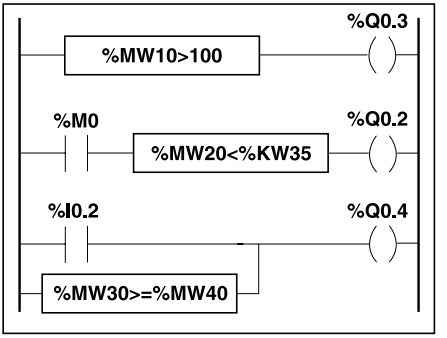
3.1-5 Comparison instructions

Comparison instructions are used to compare two operands.

> : test if operand 1 is greater than operand 2.
>= : test if operand 1 is greater than or equal to operand 2.
< : test if operand 1 is less than operand 2.
<= : test if operand 1 is less than or equal to operand 2.
= : test if operand 1 is equal to operand 2.
<> : test if operand 1 is different from operand 2.

Structure

The comparison is executed inside square brackets following instructions LD, AND and OR. The result is 1 when the comparison requested is true.




```
LD     [%MW10 > 100]
ST     %Q0.3
LD     %M0
AND    [%MW20 < %KW35]
ST     %Q0.2
LD     %I0.2
OR     [%MW30 >= %MW40]
ST     %Q0.4
```

Syntax

Operator	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
>,>=,<,<= =, <>	LD[Op1 Operator Op2] AND[Op1 Operator Op2] OR[Op1 Operator Op2]	%MWi,%KWi,%IW, %QWi,%SWi,%BLK.x	Immediate value, %MWi, %KWi,%IW,%QW,%SWi, %BLK.x,%MWi[%MWi], %KWi[%MWi]

Note

Comparison instructions can be used within parentheses.

Example :

```
LD     %M0
AND(   [%MW20>10]
OR     %I0.0
)
ST     %Q0.1
```

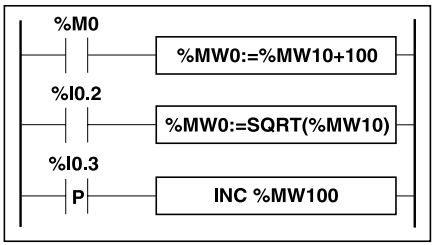
3.1-6 Arithmetic instructions

These instructions are used to perform arithmetic operations between two operands or on one operand.

+	: add two operands	**REM**	: remainder of division of the two operands
-	: subtract two operands	**SQRT**	: square root of an operand
*****	: multiply two operands	**INC**	: increment an operand
/	: divide two operands	**DEC**	: decrement an operand

Structure

Arithmetic operations are performed as follows :




```
LD      %M0
[%MW0 := %MW10 + 100]
LD      %I0.2
[%MW0 := SQRT(%MW10) ]
LDR     %I0.3
[INC %MW100]
```

Syntax

This depends on the operators used : see the table below.

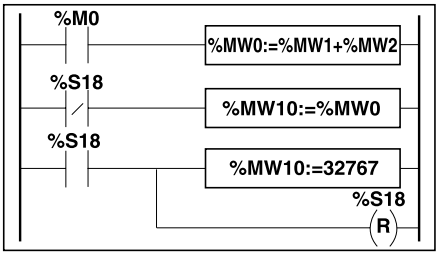
Operator	Syntax	Operand 1 (Op1)	Operands 2 and 3 (Op2 & 3)
+,-,*,/,REM (1)	[Op1: = Op 2 Operator Op3]	%MWi,%QWi, %SWi	Immed.value (2),%MWi,%KWi, %IW,%QW,%SWi,%BLK.x
SQRT	[Op1: = SQRT(Op2)]		
INC, **DEC**	[Operator Op1]		

Rules for use

• **Addition :** Overflow during operation

If the result exceeds the limits of -32768 or +32767, bit %S18 (overflow) is set to 1. The result is then, in itself, not correct (see next page). The user program manages bit %S18.

Example :




```
LD      %M0
[%MW0 := %MW1+ %MW2]
LDN     %S18
[%MW10 := %MW0 ]
LD      %S18
[%MW10 := 32767 ]
R       %S18
```

If %MW1 =23241 and %MW2=21853, the real result (45094) cannot be expressed in one 16-bit word, bit %S18 is set to 1 and the result obtained (-20442) is incorrect. In this example when the result is greater than 32767, its value is fixed at 32767.

(1) with TSX 07s V2.2 or lower, the result (Op1) of division (/) or of the remainder (REM) is not significant when the contents of operand 3 (Op3) is greater then 255.
(2) with SQRT, Op2 cannot be an immediate value.

Absolute overflow of the result (unsigned arithmetic) :
During certain calculations, it may be necessary to interpret an operand in unsigned arithmetic (bit 15 then represents the value 32768). The maximum value for an operand is 65535. Adding 2 absolute values (unsigned) whose result is greater than 65535 causes an overflow. This is signaled by system bit %S17 (carry) changing to 1, which represents the value 65536.

Example 1 : [%MW2:=%MW0 + %MW1] where %MW0 =65086, %MW1=65333
Word %MW2 contains the number 64883. Bit %S17 is set to 1 and represents the value 65536. The unsigned arithmetic result is then equal to :
65536 + 64883 = 130419.
Example 2 : [%MW2:=%MW0 + %MW1] where %MW0 =45736 (that is, a signed value of -19800), %MW1=38336 (that is, a signed value of 27200).
The two system bits %S17 and %S18 are set to 1. The signed arithmetic result (+18536) is incorrect. In unsigned arithmetic, the result (18536 + the value of %S17, that is 84072) is correct.

- **Subtraction :**

Negative result
If the result of a subtraction is less than 0, system bit %S17 is set to 1.

- **Multiplication :**

Overflow during operation
If the result exceeds the capacity of the result word, bit %S18 (overflow) is set to 1 and the result is not significant.

- **Division/Remainder :**

Division by 0
If the divider is 0, the division is impossible and system bit %S18 is set to 1. The result is then incorrect.

Overflow during operation
If the division quotient exceeds the capacity of the result word, bit %S18 is set to 1.

- **Square root extraction :**

Square root extraction is only performed on positive values. Thus, the result is always positive. If the square root operand is negative, system bit %S18 is set to 1 and the result is incorrect.

Note :
The user program is responsible for managing system bits %S17 and %S18. These are set to 1 by the PLC and must be reset by the program so that they can be reused (see previous page for example).

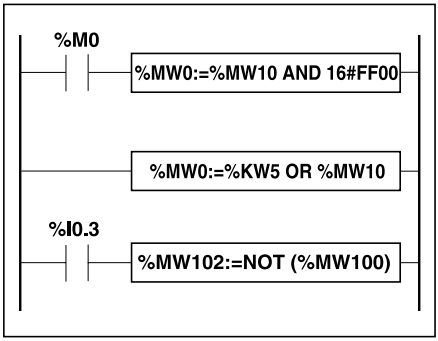
3.1-7 Logic instructions

The associated instructions are used to perform a logic operation between two word operands or on one word operand.

AND : AND (bit-wise) between two operands
OR : Logic OR (bit-wise) between two operands
XOR : Exclusive OR (bit-wise) between two operands
NOT : Logic complement (bit-wise) of an operand

Structure

Logic operations are performed as follows :




```
LD      %M0
[%MW0 := %MW10 AND 16#FF00]

LD      1
[%MW0 := %KW5 OR %MW10]

LD      %I0.3
[%MW102:= NOT (%MW100)]
```

Syntax

This depends on the operators used : see table below.

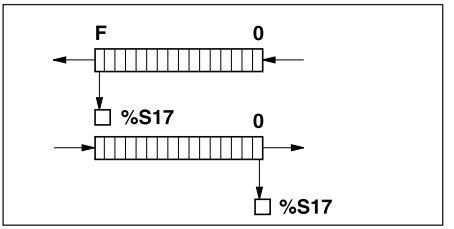
Operator	Syntax	Operand 1 (Op1)	Operands 2 and 3 (Op2 & 3)
AND, **OR**, **XOR**	[Op1: = Op2 Operator Op3]	%MWi,%QWi, %SWi	Immed. val. (1), %MWi,%KWi,
NOT	[NOT (Op2)]		%IW,%QW,%SWi, %BLK.x

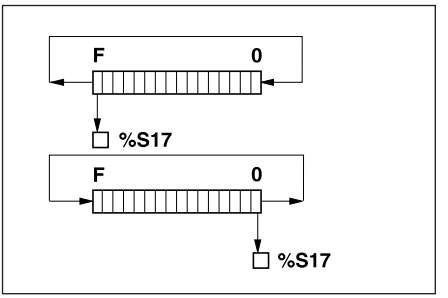
Example : [%MW15:=%MW32 AND %MW12]

(1) with NOT, Op2 cannot be an immediate value.

3.1-8 Shift instructions

Shift instructions consists of moving bits of an operand a certain number of positions to the right or to the left.
There are two types of shift operation :



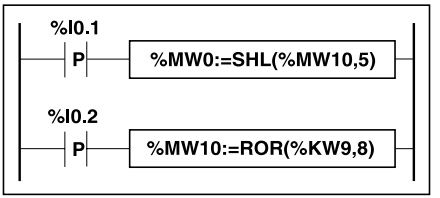


- **Logic shift** :
  - **SHL(**op2**,i)** logic shift of i positions to the left.
  - **SHR(**op2**,i)** logic shift of i positions to the right.

- **Rotate shift** :
  - **ROL(**op2**,i)** rotate shift of i positions to the left.
  - **ROR(**op2**,i)** rotate shift of i positions to the right.

As the operand to be shifted is a single length operand, the variable i must be between 1 and 16.
The state of the last output bit is stored in bit %S17.

Structure
Logic operations are performed as follows :



LDR %I0.1
[%MW0 := SHL(%MW10,5)]

LDR %I0.2
[%MW10 := ROR(%KW9,8)]

Syntax
This depends on the operators used : see the table below.

Operator	**Syntax**	**Operand 1 (Op1)**	**Operand 2 (Op2)**
SHL, **SHR** ROL**,** ROR	[Op1: = Operator(Op2,i)]	%MWi,%QWi, %SWi	%MWi,%KWi, **%IW,%QW,%SWi, %BLK.x**

3.1-9 Conversion instructions

There are two types of conversion instruction :
- **BTI** : BCD --> Binary conversion
- **ITB** : Binary --> BCD conversion

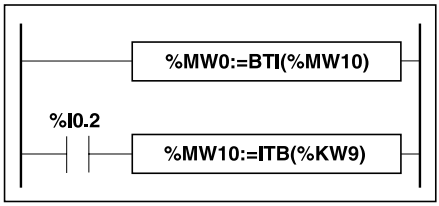
Review of the BCD code :
The BCD code (Binary Coded Decimal) represents a decimal digit (0 to 9) by coding 4 bits. A 16-bit word object can thus contain a number expressed in four digits (0 - N - 9999).
During a conversion, if the value isn't a BCD one, the system bit %S18 is placed at 1. This bit must be tested and reset to 0 by the user program.

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Example :
- Word %MW5 expresses the BCD value "2450" which corresponds to the binary value : 0010 0100 0101 0000
- Word %MW12 expresses the decimal value "2450" which corresponds to the binary value : 0000 1001 1001 0010

Word %MW5 is converted to word %MW12 by using instruction BTI.
Word %MW12 is converted to word %MW5 by using instruction ITB.

Structure
Conversion operations are performed as follows :



LD %M0
[%MW0 := BTI (%MW10)]

LD %I0.2
[%MW10 := ITB (%KW9)]

Syntax
This depends on the operators used : see the table below.

Operator	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
BTI, **ITB**	[Op1: = Operator (Op2)]	%MWi,%QWi, %SWi	%MWi,%KWi, %IW,%QW, %SWi, %BLK.x

Application examples
The BTI instruction is used to process a setpoint value at PLC inputs via BCD encoded thumbwheels.
The ITB instruction is used to display numerical values (for example, the result of a calculation, the current value of a function block) on BCD coded displays.

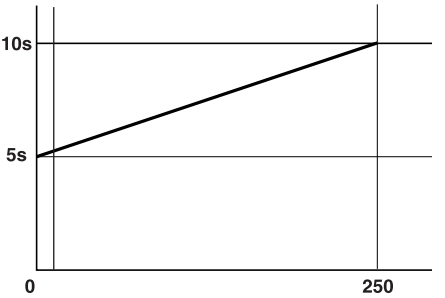
3.2 Potentiometers

Review from Part A, Section 1.8 :
"Base PLC" TSX Nano PLCs have on the front panel :
- One potentiometer for TSX Nano PLCs with 10, 14 and 20 I/O.
- Two potentiometers for TSX Nano PLCs with 16 and 24 I/O.

Programming
The numerical values from 0 to 255 corresponding to the analog values provided by these potentiometers are contained in the following two system words :
- %SW112 for the first potentiometer (leftmost, potentiometer no.0)
- %SW113 for the second potentiometer (rightmost, potentiometer no.1)

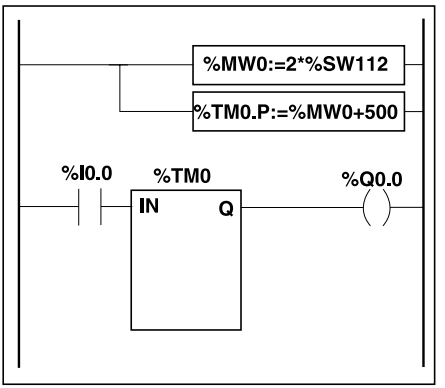
These words can be used in arithmetic operations. They can be used for any type of adjustment, for example, presetting a time-delay or a counter, adjusting the frequency of the pulse generator or machine preheating time.

Example :
Adjusting the duration of a time-delay from 5 to 10s using potentiometer no.0. For this adjustment practically the entire adjustment range of the potentiometer from 0 to 250 is used.



The following parameters are selected at configuration for the time-delay block %TM0 :
- Type TON
- Time base TB : 10ms

The preset value of the time-delay is calculated from the adjustment value of the potentiometer using the following equation %TM0.P :=2*%SW112+500.




```
LD      1
[%MW0:=2*%SW112]
[%TM0.P:=%MW0+500]
BLK     %TM0
LD      %I0.0
IN
OUT_BLK
LD      Q
ST      %Q0.0
END_BLK
```

3.3 Analog channel (TSX 07 32/33 •• ••)

TSX Nano **TSX 07 32/33 •• ••** PLCs include a non-isolated 0/10 V analog channel.

Principle

An analog to digital converter converts the input voltage 0-10 V to a digital value from 0 to 255 which is stored in system word %SW112.

Input voltage	%SW112
0 V	0
40 mV	1
80 mV	2
•	•
•	•
9.96 V	249
10 V	250
10.2 V	255

The digital value 255 is used to detect whether the maximum value of the input signal has been exceeded.

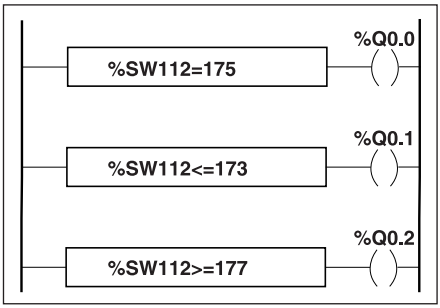
Potentiometer P0, located on the front panel, can be used to correct any errors caused by the analog measurement circuit in some applications.

Programming example

Controlling the temperature of an oven. The cooking temperature is set at 315 °C. A variation of +/- 3.6 °C results in tripping of outputs %Q0.1 and %Q0.2 respectively. Practically all of the possible setting range of the potentiometer from 0 to 250 is used in this example.

```
LD   [%SW112 = 175]
ST   %Q0.0
LD   [%SW112 <= 173]
ST   %Q0.1
LD   [%SW112 >= 177]
ST   %Q0.2
```

0 °C -> 0 V -> %SW112 = 0
311.4 °C -> 6.92 V -> %SW112 = 173
315 °C -> 7 V -> %SW112 = 175
318.6 °C -> 7.08 V -> %SW112 = 177
450 °C -> 10 V -> %SW112 = 250



3.4 Dedicated function blocks

3.4-1 Bit and word objects associated with dedicated function blocks

Dedicated function blocks use the same type of dedicated words and bits as standard function blocks (See Section 2.2).

Function block bit and word objects accessible by the program

Dedicated function blocks Sect.	Associated words and bits		Address	Write access	See
Pulse width modulation output %PWM	Word	% of pulse at 1 in relation to the total period	%PWM.R	yes	3.4-3
		Preset period	%PWM.P	No	
Pulse generator %PLS	Word	Preset value	%PLS.P	yes	3.4-4
		No. of pulses to generate	%PLS.N	yes	
	Bit	Current output	%PLS.Q	no	
		Generation done output	%PLS.D	no	
Fast counter %FC	Word	Threshold i (i = 0 or 1)	%FC.Si	yes	3.4-5
		Current value	%FC.V	no	
		Preset value	%FC.P	yes	
	Bit	Overrun output	%FC.F	no	
		Overrun threshold i output	%FC.THi	no	
Send message %MSG	Bit	Comm. fault output	%MSG.E	no	3.4-6
		Comm. done output	%MSG.D	no	
Shift bit register %SBRi (i=0 to 7)	Bit	Register bit, j=0 to 15	%SBRi.j	yes	3.4-7
Step counter %SCi (i=0 to 7)	Bit	Step counter bit, j=0 to 255	%SCi.j	no	3.4-8

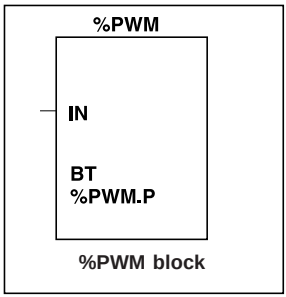
3.4-2 Programming principles

As with standard function blocks, special function blocks can be programmed in 2 different ways :

- Non-reversible programming : using special instructions.
- Reversible programming : by simulating Ladder language function blocks.

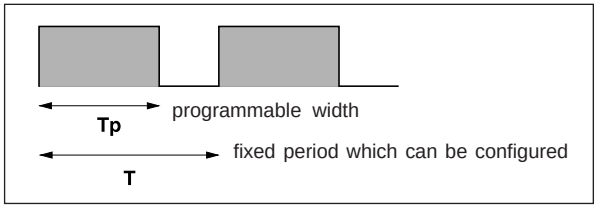
See Section 2.2-2.

3.4-3 Pulse width modulation output % PWM



%PWM block

The %PWM function block is used to generate a square wave signal on PLC output %Q0.0. The signal width (duty cycle) can be varied by the program (for description, see Part A, Section 4.6).
This function can also be used to control an analog output module wired to the output %Q0.0 (see Section B-4.4).



Characteristics

Time base	**TB**	0.1ms (1), 10ms, 1s (default value)
Preset period	**%PWM.P**	0<%PWM.P-32767 if time base is 10ms or 1s. 0<%PWM.P-255 if time base is 0.1ms (0= function not in use) The preset value and the time base can be modified during configuration. They are used to fix the signal period **T = %PWM.PxTB**. The lower the ratios to be obtained, the greater the selected %PWM.P must be. **Range of periods obtained :** • 0.2 to 26ms in steps of 0.1ms • 20ms to 5.45 min in steps of 10ms • 2s to 9.1 hours in steps of 1s
Ratio of the	**%PWM.R**	0-%PWM.R-100 (2), this word gives the period percentage of the signal at state 1 in a period (0 = default value). The "width" Tp is thus equal to : **Tp = T x (%PWM.R/100)** Word %PWM.R is written by the user program. It is this word which performs the width modulation.
Pulse generation input (or instruction)	**IN**	At state 1, the pulse width modulated signal is generated at output %Q0.0. At state 0, output %Q0.0 is set to 0.

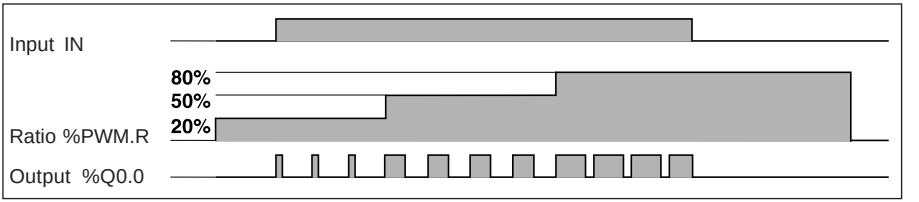
Warning :

The algorithm for processing width modulation %PWM has been refined between versions V3.0 and V3.1 of TSX Nano. For the same %PWM.R, this may result in a different duty cycle at the output between the two versions.

(1) This time base is not advisable for TSX Nano PLCs with relay outputs.
(2) Values greater than 100 will be considered as being equal to 100.

Operation

The frequency of output signal %Q0.0 is set during configuration by selecting the time base TB and the preset %PWM.P. The width of the signal is modulated by modifying the %PWM.R ratio in the program.



Programming and configuration

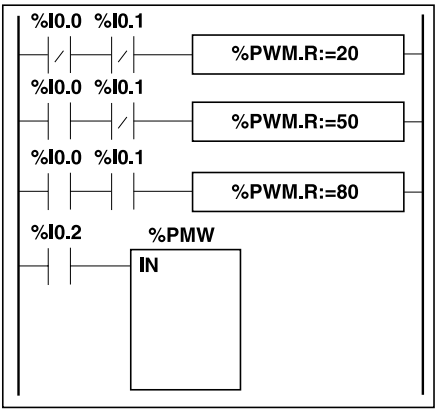
In this example, the signal width is modified by the program according to the state of PLC inputs %I0.0 and %I0.1.

The signal period is set at 500 ms during configuration. (See configuration note below.)

If %I0.0 and %I0.1 are set to 0, the %PWM.R ratio is set at 20%, the duration of the signal at state 1 is then : 20% x 500 ms = 100 ms.

If %I0.0 is set to 0 and %I0.1 is set to 1, the %PWM.R ratio is set at 50% (duration 250 ms).

If %I0.0 and %I0.1 are set to 1, the %PWM.R ratio is set at 80% (duration 400 ms).




```
LDN   %I0.0
ANDN  %I0.1
[%PWM.R:=20]
LD    %I0.0
ANDN  %I0.1
[%PWM.R:=50]
LD    %I0.0
AND   %I0.1
[%PWM.R:=80]
BLK   %PWM
LD    %I0.2
IN
END_BLK
```

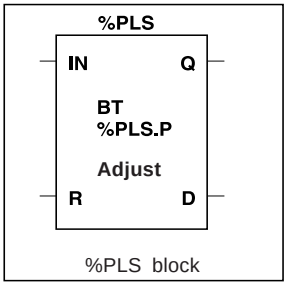
Configuration

Output %Q0.0 = output %PWM TB= 10ms %PWM.P=50

Special cases

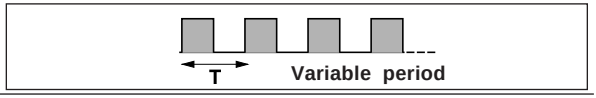
- **Effect of a cold restart :** (%S0=1) sets the %PWM.R ratio to 0
- **Effect of a warm restart :** (%S1=1) has no effect
- **Effect of a PLC stop :**
  The output %Q0.0 is set to 0 regardless of the state of system bit %S8. For PLCs, version 2.2 or lower, the output %Q0.0 is set to 0 if %S8 is at 1 or maintained (signal generation) if %S8 = 0.
- **With 0.1 ms time base**, forcing output %Q0.0 using a programming device does not stop the signal generation.

3.4-4 Pulse generator output % PLS



%PLS block

The %PLS function block generates a square wave signal (on to off ratio, or duty cycle, of 50%) at PLC output %Q0.0.
This signal can be :
• of limited length, the number of pulses and the period are written by the program (or during configuration).
• of unlimited length, the period is written by the program or during configuration.
(see description in Part A, Section 4.5).

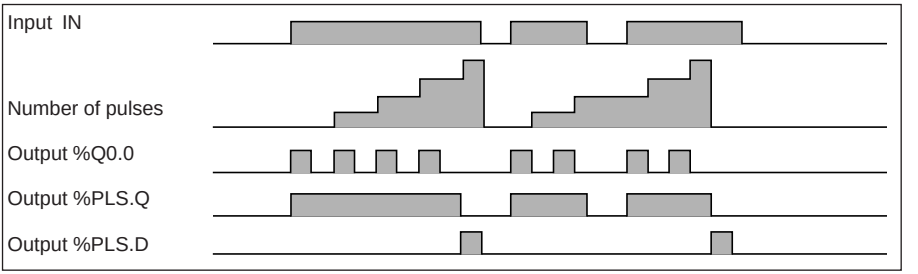


Characteristics

Time base	**TB**	0.1 ms (1), 10ms, 1s (default value)
Preset	**%PLS.P**	0<%PLS.P<32767 if TB =10 ms or 1s. 0<%PLS.P<255 if TB = 0.1ms (1)(2) (0 = output at 0, 1 = output at 1). The preset value is used to modulate the signal period **T = %PLS.PxTB**. **Range of periods obtained :** • 0.2 to 26ms in steps of 0.1ms • 20ms to 5.45 min in steps of 10ms • 2s to 9.1 hours in steps of 1s. **Note : %PLS.P must be an even number**
Number of pulses	**%PLS.N**	0<%PLS.N-32767, this word gives the number of pulses of the train to be generated. 0 = square wave signal of unlimited length (by default). %PLS.N is tested and written by the program.
Adjustment via the terminal	**Y/N**	Y : possible to modify the preset value %PLS.P.N in the Data editor. N : no access in the Data editor
Pulse generation input (or instruction)	**IN**	At state 1, generates the signal on output %Q0.0. At state 0 sets output %Q0.0 to 0.
Reset input (or instruction)	**R**	At state 1, resets the number of pulses of outputs %PLS.Q, and %PLS.D to zero
Current pulse generation output	**%PLS.Q**	State 1, pulse signal is generated at %Q0.0.
Pulse generation done output	**%PLS.D**	State 1, signal generation is done
PLS counter (2)		N=no, Y=yes. This option enables input %I0.0 to be used as a counting input.

(1) This time base is not advisable for TSX 07s with relay outputs.
(2) The PLS counting option is required when a time base of 0.1 ms is selected. In order for the PLS to work properly, it is necessary to physically loop output %Q0.0 to input %I0.0. In this type of operation, the %PLS.P must be greater than or equal to 6 (max. freq. =1.6 Khz) to ensure correct operation of the function.

Operation



Special cases

- **Effect of a cold restart :** (%S0=1) sets the value of %PLS.P to that defined during configuration
- **Effect of a warm restart :** (%S1=1) : no effect
- **Effect of a PLC stop :** see effect of a PLC stop page 3/18.
- **Effect of modifying the preset %PLS.P :** modifying the preset value %PLS.P via an instruction or by adjusting it takes effect instantly
- **With the time base of 0.1 ms**, forcing the loop input %I0.0 does not stop generation.

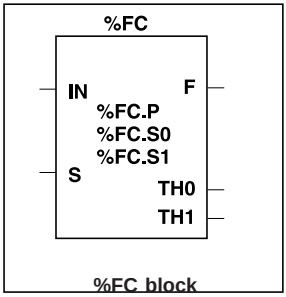
Comment :

Using a time base of 0.1 ms may create a discrepancy of one pulse between the number of pulses requested (%PLS.N) and the number actually generated.
To overcome this, the function must be reset once generation is complete :

```
LD   %Mi
IN   %PLS
N
R    %PLS
```

However, this action restarts generation of %PLS.N pulses once input IN changes to 1.

3.4-5 Fast counting functions, frequency meter and up/down counter %FC



%FC block

The %FC function block can be used to perform one of the three following functions :
- Fast counting
- Frequency meter
- Up/down counting

(see detailed description, Part A, Section 4.4). The function is selected during configuration.
Note : Fast counting functions can be used without any programming, simply by configuring the I/O and the parameters.

Characteristics of block %FC

The function block offers a group of words, input and output bits which are used to generate the 3 counting functions. To fully understand the role of each object in the function to be performed, refer to the detailed description of the function.

Current value	**%FC.V**	Word incremented or decremented according to the inputs and the function selected. Can be read and tested but not written.
Preset value	**%FC.P**	Only used by the up/down counting function : 0-%FC.P-65535. Word can be read, tested and written.
Threshold value S0 (1)	**%FC.S0**	0-%FC.S0-65535 word containing the value of threshold 0, defined during configuration can be read and written by the program.
Threshold value S1 (1)	**%FC.S1**	0-%FC.S1-65535 word containing the value of threshold 1, defined during configuration can be read and written by the program.
Enable input (or instruction)	**IN**	At state 1, validates the current function. At state 0, inhibits the current function.
Set input (or instruction)	**S**	At state 1 : • sets the current value to the preset value (up/down counting function) or resets the current value to 0. • initializes operation of the threshold outputs (HSC_Out) • takes into account threshold values %FC.S0 and %FC.S1 modified by the program.
Overflow output	**%FC.F**	State 1, when the current value %FC.V exceeds 65535. %FC.F can be cleared by a Preset (%I0.1 or S instruction) or a cold restart.
Threshold bit 0 (2)	**%FC.TH0**	State 1, when the current value is greater than or equal to the threshold value %FC.S0
Threshold bit 1 (2)	**%FC.TH1**	State 1, when the current value is greater than or equal to the threshold value %FC.S1

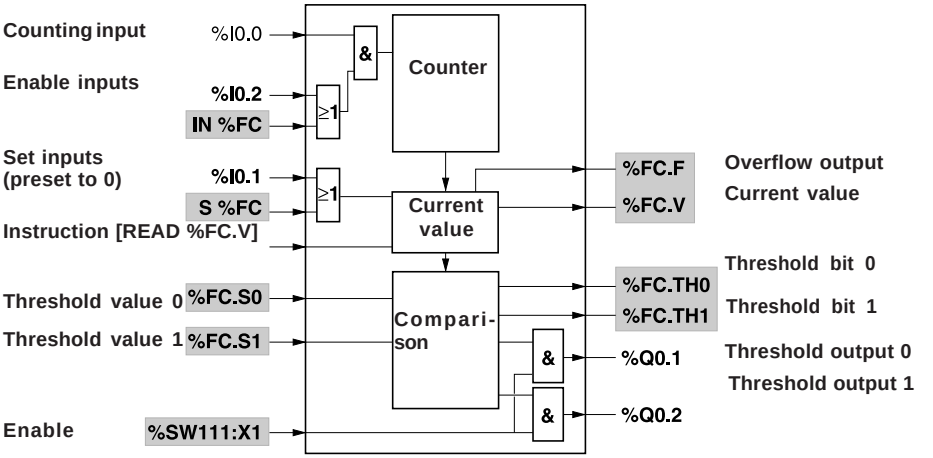
(1) Threshold value %FC.S0 must be less than threshold value %FC.S1.
(2) It is advisable to test bit %FC.THx only once in the program.

Fast counting function

The fast counting function counts at a maximum frequency of 10 kHz in fast mode (or 5 kHz in normal mode, selected during configuration), with a range of 1- 65535.
The counter receives the signals to be counted at PLC input %I0.0. The counting value (current value %FC.V) is compared to 2 thresholds %FC.S0 and %FC.S1, which are defined during configuration and can be modified by the program (modification is taken into account when the Set input is activated).

Fast counter

Diagram :



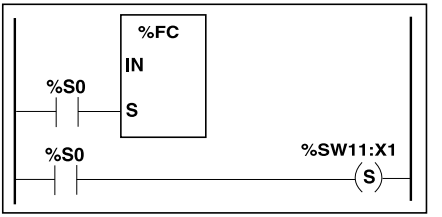
Comment : Apart from counting input %I0.0, all other function block discrete I/O are optional (they can be selected during configuration). The equivalent programming objects and instructions are available for each of these (marked by a box on the diagram).

Threshold outputs : Threshold outputs are controlled directly by the fast counter (without waiting for the outputs to be updated at the end of the scan) according to the settings established by the user during configuration.

Output	FC.V< thres.0<thres.1	thres.0-FC.V-thres.1	thres.0<thres.1<FC.V
%Q0.1	0 or 1	0 or 1	0 or 1
%Q0.2	0 or 1	0 or 1	0 or 1

When they are initialized the threshold outputs must be enabled by a fast counter preset command. Programming example :

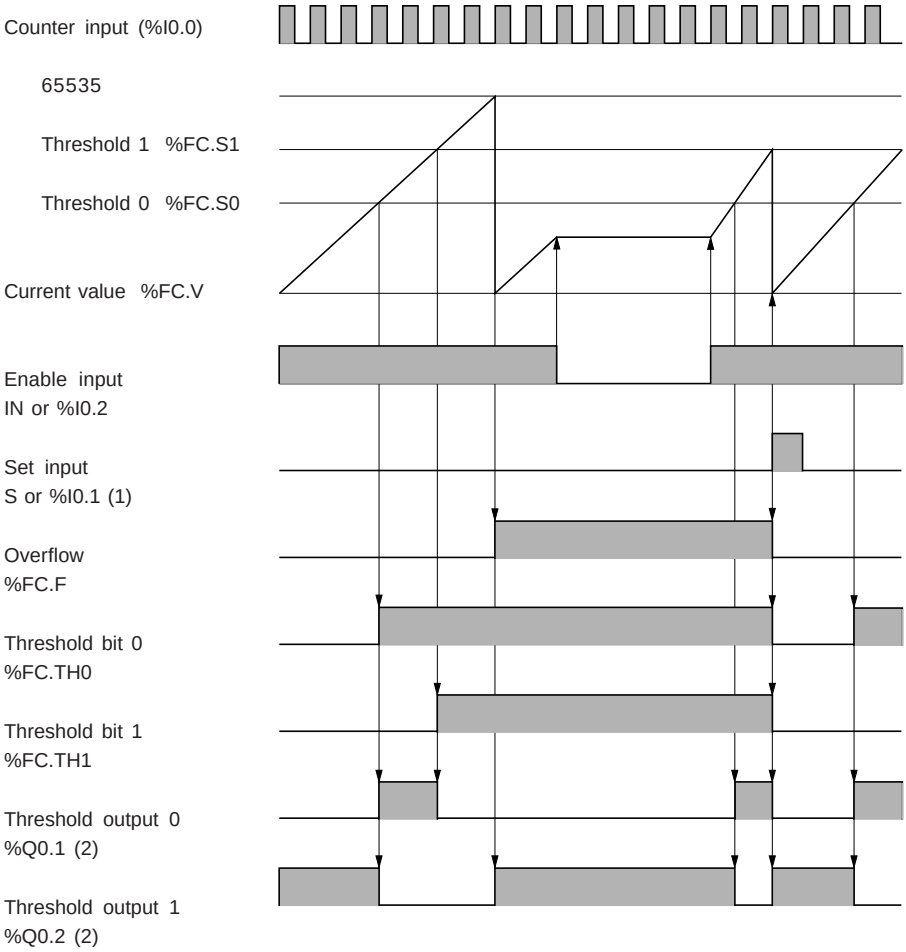
```
BLK  %FC
LD  %S0
S
END_BLK
LD  %S0
S   %SW111:X1  (enable bit %SW111:X
                is set to 1)
```



B

Reading the current value

The current %FC.V value is updated at the end of each PLC scan. %FC.V can also be updated by the READ instruction :
syntax : **[READ %FC.V] Timing diagram**



(1) Input %I0.1 operates on a rising edge (see timing diagram), while input S operates on a state.
(2) In this timing diagram the status matrix configured by the user is as follows :

Output	FC.V< thresh 0<thresh 1	thresh 0-FC.V-thresh	1 thresh0<thresh1<FC.V
%Q0.1	0	1	0
%Q0.2	1	0	0

Frequency meter function

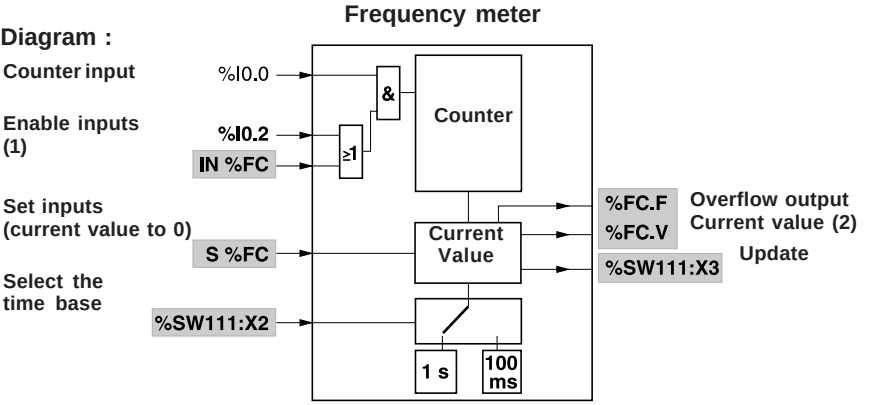
The frequency meter function is used to measure the frequency of a periodic signal in Hz on input %I0.0. There is a choice of two modes : fast (10kHz filtering) or normal (5kHz filtering).

The frequency range which can be measured is from 1Hz to 10kHz. The user can choose between 2 time bases, the choice being made by system bit %SW111:X2 (1 = time base of 100 ms, 0 = time base of 1s).

Time base	Measurement range	Precision	Update
100ms	100Hz-10kHz	0.1% for 10kHz 10% for 100Hz	10 times per second
1s	10Hz-10kHz	0.01% for 10kHz 10% for 10Hz	Once per second

System bit %SW111:X3 is set to 1 during an update of the current value. The user program must reset the system bit to 0.

Diagram :

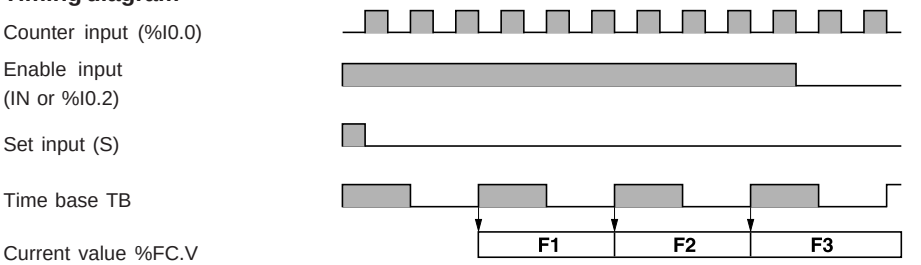


(1) Input %I0.2 is optional, its use is selected during configuration.
(2) Current value FC.V is expressed in Hz.

Note :

This function can also be used to obtain the value of the analog module wired to input %I0.0 (see part B Section 4.3 : managing analog I/O).

Timing diagram

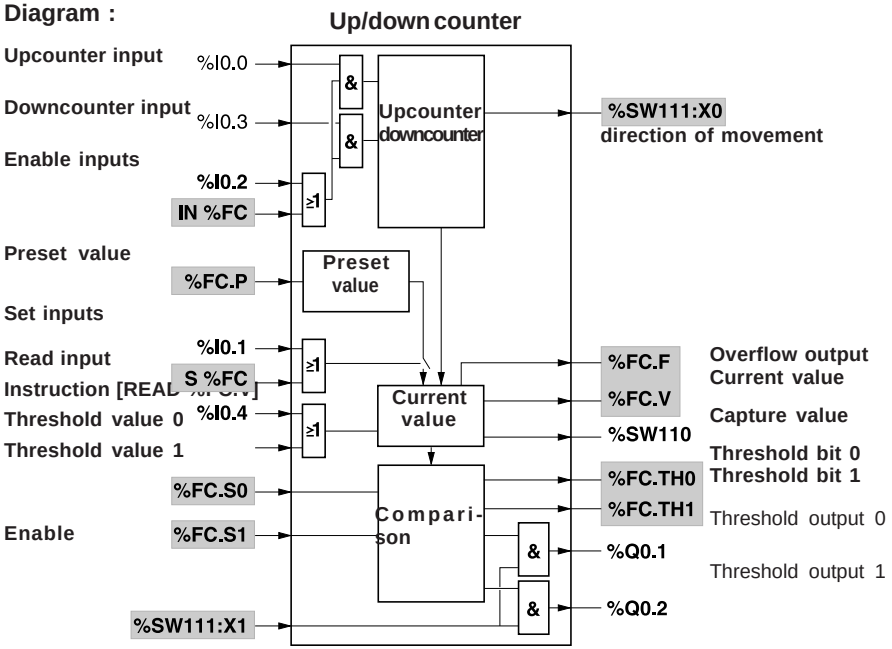


Up/down counting function

The up/down counting function counts up or down at a maximum frequency of 1 kHz, with an up/down counting range of 0 to 65535.

The counter receives the signals to be upcounted on PLC input %I0.0, and the signals to be downcounted on PLC input %I0.3. It provides an output indicating the direction of movement : whether the counter is upcounting %SW111:X0 = 1 or downcounting %SW111:X0 = 0. The up/down counting value (current value %FC.V) is compared with 2 thresholds, %FC.S0 and %FC.S1, which are defined during configuration and can be modified by the program (modification is taken into account when the preset input is activated).

Diagram :



Apart from upcounting input %I0.0 and downcounting input %I0.3, all other function block discrete I/O are optional (they can be selected during configuration). The equivalent programming instructions and objects are available for each of these (marked by a box on the diagram).

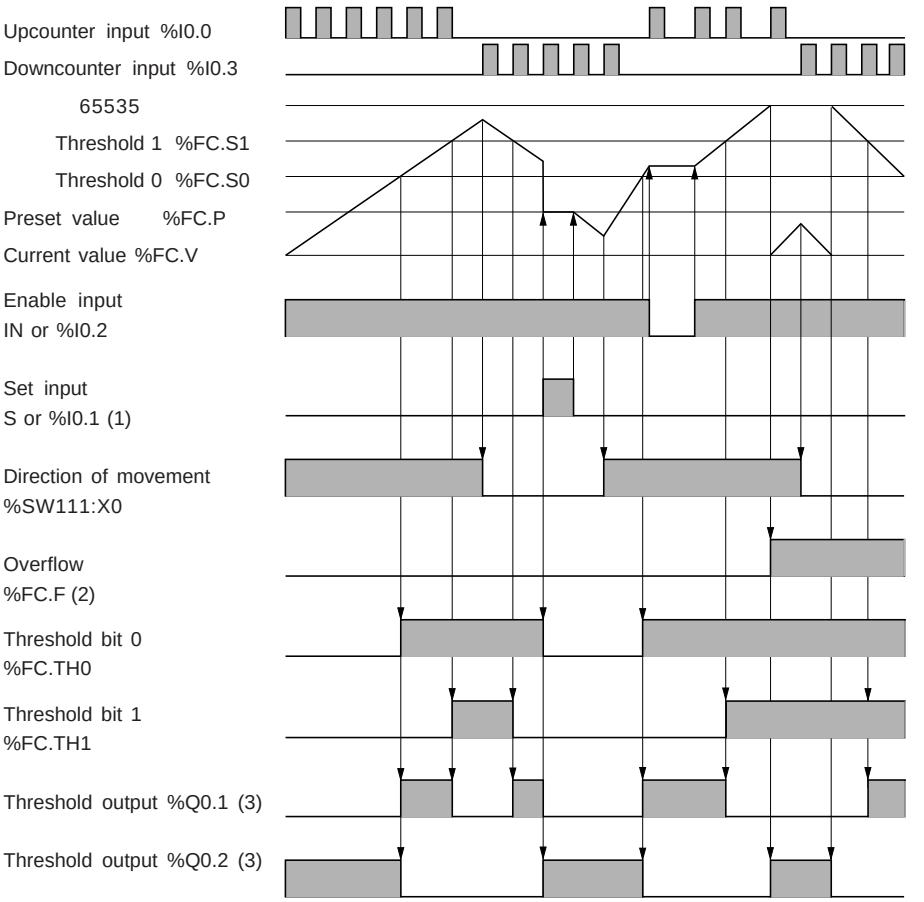
Preset : the preset value of 0 to 65535 is defined during configuration and can be modified by the program. The current value is loaded with the preset value on a rising edge at input %I0.1 or at state 1 of input S.

Threshold outputs : see fast counting function

Reading the current value

The value %FC.V is updated automatically at the end of the scan. %FC.V can also be updated by the READ instruction, syntax : **[READ %FC.V]**

A rising edge on Read input %I0.4 causes the value %FC.V to be copied to %SW110.



(1) Input %I0.1 operates on a rising edge, while input S operates on a state.
(2) The overflow output %FC.F is reset to zero after reinitialization of the counter.
(3) See status matrix for the fast counting function.

Important :

If one of the 4 inputs %I0.0, %I0.1, %I0.3, or %I0.4, on a rising or falling edge is at state 1, no action linked to the other 3 will be performed.

Special cases (fast counting, frequency meter and up/down counter)

- **Effect of a cold restart :** (%S0=1) changes the current value, outputs %FC.F, %FC.TH0, %FC.TH1 and enable bit of the threshold outputs (%SW111:X1) to 0, and copies the values defined during configuration to words %FC.P,%FC.S0/S1.

- **Effect of a warm restart :** (%S1=1) **and PLC stop :** has no effect on the current value.

  After an activated master control relay instruction MCS, high-speed outputs are set to 0, after deactivation of the MCS, they are only reset after the threshold is crossed in upcounting mode or after an up/down counting action in downcounting mode.

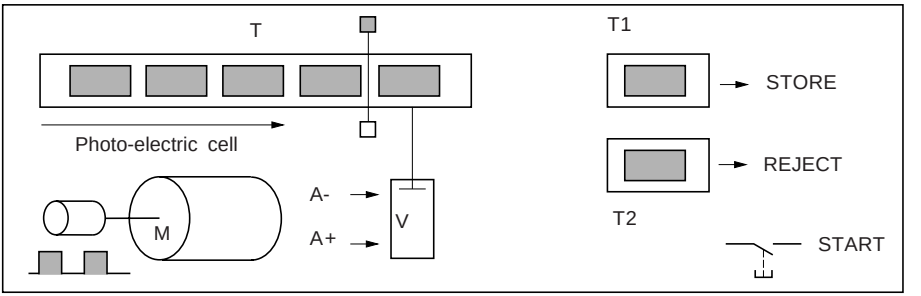
- **Displayed %FC values :**
  For all values related to the %FC function block, values are displayed :
  - by the FTX 117 terminal as two's complement numbers (Decimal) or as a direct value (Hexadecimal).
  - by PL7-07 as direct value (Decimal or Hexadecimal).

  For example :

	Displayed values		
%FC.V	**(PL7-07)**	**(FTX 117)**	**(PL7-07 - FTX 117)**
0	0	0	0000
1	1	1	0001
2	2	2	0002
•	•	•	•
•	•	•	•
•	•	•	•
32766	32766	32766	7FFE
32767	32767	32767	7FFF
32768		-32768	8000
•	•	•	•
•	•	•	•
•	•	•	•
65533	65533	-3	FFFD
65534	65534	-2	FFFE
65535	65535	-1	FFFF
0	0	0	0

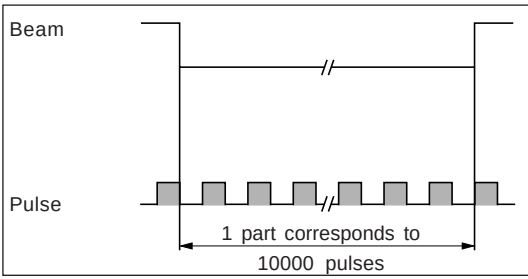
Example of the fast counting function

Description of the application



The parts to be measured are put on a conveyor belt driven continuously without slipping by a motor to which a rotary incremental encoder is attached. The measurement is made by counting the number of pulses during the time cell C detects the moving part. A cylinder V controls the lateral movement of belt T so as to position it opposite conveyor T1 or conveyor T2, according to the measurement results.

The number of pulses measured is compared to 2 extreme values (length measurement tolerance)



- If it is between these 2 values, the part is directed towards the STORE (belt T1, control A+ on cylinder V)
- If this is not the case, the part is directed towards the reject bin (belt T2, control A- on cylinder V)

The START pushbutton is used to start the whole operation.

Assigning the I/O

Inputs
- %I0.0 counter input connected to the incremental encoder
- %I0.1 reset input connected to the photoelectric cell
- %I0.2 enable input connected to the start button

Outputs
- %Q0.1 cylinder A+ control output
- %Q0.2 cylinder A- control output
- %Q0.0 belt control output

Processing the application

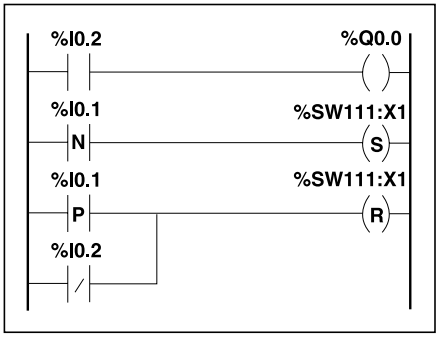
The fast counting function can be processed without the need to program the PLC by configuring only the %FC function block.

%FC : counter
Mode : fast
Counting input : %I0.0
Set : %I0.1
Enable input : %I0.2
%FC.S0 : 9950 threshold 0 corresponds to the minimum tolerance level
%FC.S1 : 10000 threshold 1 corresponds to the maximum tolerance level

Output matrix

Output	FC.V< thresh 0<thresh 1	thresh 0-FC.V-thresh 1	thresh 0<thresh 1<FC.V
%Q0.1	0	1	0
%Q0.2	1	0	1

Program




```
LD    %I0.2
ST    %Q0.0
LDF   %I0.1
S     %SW111:X1
LDR   %I0.1
ORN   %I0.2
R     %SW111:X1
```

Operation

Input %I0.0 counts the number of pulses sent from the incremental encoder as soon as the enable input %I0.2 (start button) is activated.

On a rising edge at input %I0.1, the current value of the counter is reset to 0.

When the cell (input %I0.1) detects that there are no more parts on the conveyor belt, outputs %Q0.1 and %Q0.2 are enabled (by bit %SW111:X1) and set to 0 or 1 at this moment depending on the current value of the counter %FC.V (according to the output matrix).

Output %Q0.1 is set to 1 when the part is within tolerance. It controls the output of positioning cylinder A on conveyor belt T1 (see the output matrix).

Output %Q0.2 is set to 1 when the part is outside tolerance (return cylinder A to belt T2).

3.4-6 Transmitting/Receiving messages and controlling data exchanges

The TSX Nano can communicate with a programming terminal (either the FTX117 or the PL707 software) and with other UNI-TELWAY devices connected to the terminal port. The TSX Nano can also be configured to send and/or receive messages in character mode (ASCII protocol).

PL7 language offers for this :

- **EXCH** instruction to transmit/receive messages,
- **%MSG** function block to control the data exchanges.

The TSX Nano determines the protocol according to the pinout of the cable being used, and provides this information to the system bit %S100 (management of /DPT).

An overview of UNI-TELWAY compatible products and configurations appears in Part F of this manual.

EXCH instruction

The SEND instruction allows the TSX Nano to send and/or receive information to/from UNI-TELWAY or ASCII devices. The user defines a table of words (%MWi:L or %KWi:L) containing the data to be sent and/or received (up to 64 data words in transmission and/or reception). The format for the word table is described in the paragraphs about each protocol (ASCII and UNITELWAY).

A message exchange is performed using the **EXCH** instruction.

Syntax: [EXCH %MWi:L] (1) or [EXCH %KWi:L]

Note : Devices such as XBT or CCX man-machine interfaces, or Inductel identification devices, which support UNI-TELWAY protocol, can send information to, or get information from the TSX Nano without any special TSX Nano programming.

The TSX Nano must finish the exchange from the first EXCH instruction before a second can be launched. The %MSG block must be used when sending several messages.

(1) L : Number of words in the word table.
Values of the internal word table %MWi:L are such as i+L - 255.

Note

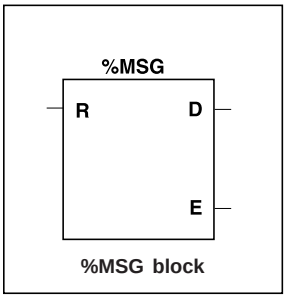
The PL7-07 V1 software can only be used for transmission, using the SEND instruction for TSX07 V1 and V2.
TSX Nano V3 PLCs (TSX 07 3• •• ••) authorize transmission and/or reception of messages using the EXCH instruction (the EXCH instruction in reception mode on TSX 07 3• 10 •• PLCs is only possible in version V3.1).
(For PL7-07 V1, the EXCH instruction is called SEND).

%MSG function block

The use of the %MSG function block is optional; it can be used to manage data exchanges. The %MSG function block has three purposes :

- **Communications error checking**
  The error checking verifies that the block length (word table) programmed with the EXCH instruction is large enough to contain the length of the message to be sent (compare with length programmed in the least significant byte of the first word of the word table). It also verifies that a UNI-TELWAY message was received by the UNI-TELWAY device.
- **Coordination of multiple messages**
  To ensure the coordination when sending multiple messages, the %MSG function block provides the information required to determine when a previous message is complete.
- **Transmission of priority messages**
  The %MSG function block allows current message transmissions to be stopped, in order to allow the immediate sending of an urgent message.

Exchange control block



%MSG block

The **%MSG** function block manages exchanges.
Programming it is optional.

Characteristics of the %MSG block

Reset input (or instruction)	**R**	At state 1, reinitializes communication, %MSG.E = 0 and %MSG.D = 1
Comm. done output	**%MSG.D**	State 1, comm. done if : • End of transmission if transmission, • End of reception (end character received), • Error, • Reset the block. State 0, request in progress
Fault (Error) output	**%MSG.E**	State 1 if : • Bad command, • Table incorrectly configured • Incorrect character received (speed, parity, etc) • Reception table full (not updated) State 0, message length OK, link OK

If an error occurs when using an EXCH instruction, bits %MSG.D and %MSG.E go to 1 and system word %SW69 contains the error code. See Section 6.2-2.

Reset input (R) : When the Reset input is set to 1, it immediately stops any messages that are being transmitted, resets the Fault (Error) output to 0 and sets the Done bit to 1. A new message can now be sent.

Fault (Error) output (%MSG.E) : **The error output is set to 1 either because of a communications programming error or a message transmission error. The error output is set to 1 if the number of bytes defined in the data block associated with the EXCH instruction (word 1, least significant byte) is greater than 128 (80 in hexadecimal).**

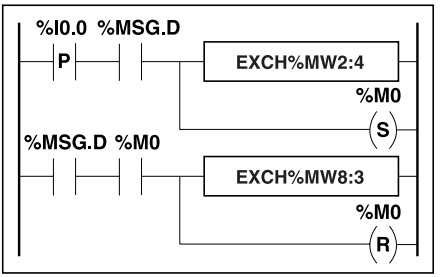
The error output is also set to 1 if a problem exists in sending a UNI-TELWAY message to a UNI-TELWAY device. In this case, the user should check wiring, and that the destination device supports UNI-TELWAY communication.

Communications Done output (%MSG.D) **: When the Done output is set to 1, the TSX Nano is ready to send another message. Use of the %MSG.D bit is recommended when multiple messages are sent. If it is not used, messages may be lost.**

Transmission of several successive messages

Execution of the EXCH instruction activates a message block in the application program. The message is transmitted if the message block is not already active (%MSG.D=1). If several messages are sent in the same cycle, only the first message is transmitted. The user is responsible for managing the transmission of several messages using the program.

Example : Transmission of 2 messages in succession.



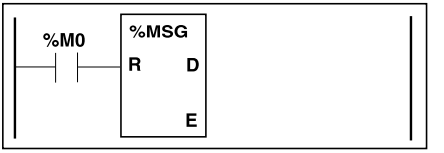

```
LDR   %I0.0
AND   %MSG.D
[EXCH %MW2:4]
S     %M0
LD    %MSG.D
AND   %M0
[EXCH %MW8:3]
R     %M0
```

Reinitializing exchanges

An exchange is cancelled by activating the input (or instruction) R. This input initializes communication and resets output %MSG.E to 0 and output %MSG.D to 1. It is possible to reinitialize an exchange if a fault is detected.

Example :

```
BLK %MSG
LD    %M0
R
END_BLK
```



Special cases

- **Effect of a cold restart :** (%S0=1) forces a reinitialization of the communication.

- **Effect of a warm restart :** (%S1=1) has no effect.
- **Effect of a PLC stop :** if a message transmission is in progress, the PLC stops its transfer and reinitializes the outputs %MSG.D and %MSG.E.

Characteristics of ASCII mode

The ASCII output communications mode is selected by jumpering pins on the terminal port (see Part F for specific information). Status bit %S100 is set to 1 when the TSX Nano is in ASCII mode. It also confirms that, with the necessary jumpers, the cable is connected.

The three possible uses of this instruction are :

- **Transmission**
- **Transmission / Reception**
- **Reception**

The maximum size of the frames transmitted and/or received is 128 bytes.
The word table associated with the EXCH instruction is composed of transmission and reception tables.

Most significant byte	Least significant byte	
Command	Length LgT / LgR	Control
Byte transmitted 1	Byte transmitted 2	Transmission table
...	...	
...	Byte transmitted n	
Byte transmitted n+1		
Byte received 1	Byte received 2	Reception table
...	...	
...	Byte received p	
Byte received p+1		

The Length byte (LgT) contains the length to be transmitted, which is overwritten by the number of characters received (LgR) at the end of reception.

The command byte must contain one of the following values :

- 0 : Transmission
- 1 : Transmission followed by reception
- 2 : Reception

The table can only be type %KWi in the case of transmission.
Reception ends when the end of frame byte is received (1). The value of this byte can be modified by the user (least significant byte of system word %SW68). The default value of this word is H'0D' (carriage return).

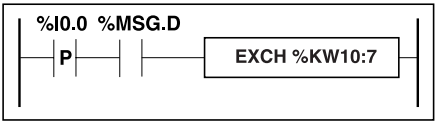
(1) Warning :
The system does not manage a reception time-out

Sending a message to an ASCII device : Pure transmission
The content of the data block associated with the SEND instruction needed to write data to an ASCII device (such as a display or printer) is as follows :

Most significant byte	Least significant byte
0 (transmission)	Length of message (bytes)
ASCII data	

The maximum length of a message is 128 bytes.

Example 1 : request to display message "FAULT 10" on a printer using ASCII protocol.




```
LDR   %I0.0
AND   %MSG.D
[EXCH %KW10:7]
```

Word table contents :

Word	Contents	Most significant byte	Least significant byte
%KW10	12	0	Length LgT in bytes
%KW11	'DE'	ASCII text	
%KW12	'FA'		
%KW13	'UT'		
%KW14	' 1'		
%KW15	'0 '		
%KW16	16#0A0D	Line return	Carriage return

ASCII Transmission / Reception

At the end of transmission, the TSX Nano switches to wait for reception, then after receiving a response, copies it into the %MWi zone contiguous with the transmission table if the status of the reception is OK and if the length of the question (LgT) and the response (LgR) is less than the %MWi reserved zone (length L). If this is not the case, bit %MSG.E changes to 1.
Reception is ended when the end code is detected (H'0D' by default but may be modified in %SW68) or when the reception table is full.
There is no reception time-out management.

Note

TSX Nano V1 or V2 cannot receive ASCII messages.

The contents of the word table associated with the EXCH instruction necessary for transmission / reception of ASCII data is described below :

Most significant byte	**Least significant byte**	
1 (transmission / reception)	Length LgT or LgR	Control
Byte to be transmitted 1	Byte to be transmitted 2	Transmission table
...	...	
...	Byte to be transmitted n	
Byte to be transmitted n+1		
Byte received 1	Byte received 2	Reception table
...	...	
...	Byte received p	
Byte received p+1	End code (H'0D')	

Notes

%KWi type words are prohibited.

When the exchange is finished (end of frame character received), the transmission length byte (LgT) contains the number of characters received (LgR).
The transmission length byte must therefore be updated before each exchange.

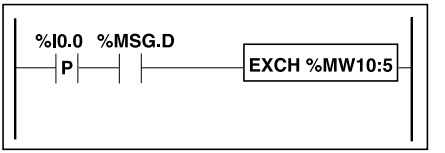
The message reception zone is aligned to the word following the transmission zone.

Example :

Words	Most signif.	Least signif.
%MW10	16#0001	16#0007
%MW11	'V'	'A'
%MW12	'L'	'U'
%MW13	'E'	' '
%MW14	':'	not used

Associated program :

LDR %I0.0
AND %MSG.D
[EXCH %MW10:9]



This program transmits the following frame: VALUE : 7 bytes and waits to receive a response (8 bytes maximum).
The characters received are accessible in the words %MW15 to %MW18.

Calculation of the length of the word table

LgT = 7
LgR = 8

$$L = 1 + \frac{LgT + LgT\%2}{2} + \frac{LgR + LgR\%2}{2} = 9$$

Note

TSX 07 2•••• Nano-PLCs cannot receive ASCII messages.

Receiving a message from an ASCII device

On execution of the EXCH block, which has been set for reception, the TSX Nano switches to wait for reception, then copies the response into the %MWi zone if the status of the reception is OK and if the length of the response (LgR) is less than the %MWi reserved zone (length L). If this is not the case, bit %MSG.E is set to 1.
Reception is ended when the end code is detected (16#0D by default but may be modified in %SW68) or when the reception table is full.
There is no reception time-out management.

Note

The TSX Nano V1 or V2 cannot receive an ASCII message.

The contents of the word table associated with the EXCH instruction necessary for receiving ASCII data is described below :

Most significant byte	**Least significant byte**	
2 (reception)	0 (1)	Control
Byte received 1	Byte received 2	Reception table
...	...	
...	Byte received p	
Byte received p+1	End code (H'0D')	

Note

%KWi type words are prohibited.

The communication type is managed in the first word of the table.

(1) In reception mode, this least significant byte is not taken into account.

Exchange control

Exchange control is performed using the %MSG function block and system word %SW69.
After each exchange, %SW69 (report of EXCH block) is updated and takes one of the following values :

- 0: Exchange OK.
- 1: Transmission table too large (LgT>128).
- 2: Transmission table too small (LgT=0).
- 3: Word table too small (1).
- 7: Bad ASCII command (command byte <> 0, 1 or 2).
- 8: Not used
- 9: Reception error (problem with communication format (speed, parity)).
- 10: %KWi table prohibited in reception or transmission / reception.

(1) $L < 1 + \frac{LgT + LgT\%2}{2} + \frac{LgR + LgR\%2}{2}$

where L in words
LgT and LgR in bytes

Dialog with a UNI-TELWAY device

The word table associated with the EXCH instruction used to send a request to a UNI-TELWAY device such as ATV speed controllers or man-machine interface devices (CCX 17 or XBT) is made up of transmission and reception tables.

Most significant byte	**Least significant byte**	
Target address	Length LgT / LgR (bytes)	Control
Category code	Request code	Transmission table
First word (MSB)	First word (LSB)	
...	...	
Word n-1 (MSB)	Word n-1 (LSB)	
Word n (MSB)	Word n (LSB)	
00 (forced)	Response code received	Reception table
Data received 2	Data received 1	
...	...	
Data received p-1	...	
	Data received p	

The maximum size of the messages transmitted and received is 128 bytes.
The message reception zone is always aligned on the word following the transmission zone.

The EXCH block reads the length to be transmitted (LgT) in the transmission length field. When the exchange has finished, it writes the length of the message received there (LgR). **The transmission length byte must therefore be updated before each transmission.**

The response code received is written in the least significant byte of the first word of the reception table. The most significant byte of this word is forced to 0. The following data, if there is any, is aligned on the next word.

Important

From TSX Nano V3, the order for transmitting data from a UNI-TE request becomes Most Significant byte then Least Significant byte.
Applications operating with TSX Nano V1 or V2 must be modified to take account of this inversion if they are loaded onto a TSX Nano V3.

Unitelway Master

In this mode, the TSX Nano normally manages two devices distributed over 5 slave addresses. It is possible to control 2 devices and a PL7 07 programming terminal, if the programming terminal is configured on a single address.

The TSX Nano does not manage slave to slave routing.

The TSX Nano Master can transmit a request to any slave address from 1 to 5, using the EXCH block. It uses the source address 0.254.16.

The target address coded in the word table associated with the EXCH block must be one of the following :

- 0: Transmission of a request to slave 4 (compatible with TSX07 2.).
- 1 to 5: Transmission and reception of a request to a slave with address 1 to 5.

If the target address is 0, the characteristics of the EXCH block are as follows :

- The reception buffer is not used,
- The table can be situated in zone %KWi,
- Bit %MSG.D changes to 1 when the response from the slave is received,
- The response from the slave is ignored.
- Only Write and non-solicited Data requests can be used.

If the target address is between 1 and 5 :

- The reception table is obligatory (1 word minimum),
- The word table must be situated in zone %MWi,
- Bit %MSG.D changes to 1 when the response from the slave is received,
- The response from the slave is copied into the reception table.

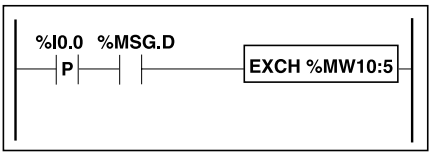
Example of use :

Transmission of the request "Read word" %MW513 (16#0201) to a device at address 2 :

Words	Most signif.	Least signif.
%MW10	02	04
%MW11	07	04
%MW12	02	01

Associated program :

```
LDR   %I0.0
AND   %MSG.D
[EXCH %MW10:5]
```



The EXCH block uses %MWi:L as parameters :

- i indicates the number of the first word in the table
- L indicates the number of words in the word table.

When bit %MSG.D = 1 and bit %MSG.E = 0, the table contains the following data :

Words	Most signif.	Least signif.
%MW10	02	**04**
%MW11	07	04
%MW12	02	01
%MW13	**00**	**34**
%MW14	'**AB**'	'**CD**'

The data in **bold** signifies :

- 4 bytes received,
- response code received = 16#0034,
- value of word %MW513 = 16#ABCD.

Unitelway Slave

Any device (local or remote) can question the system server from the TSX Nano Slave using Ad0 (server) as a target address.

A TSX Nano slave can transmit (client) a request to any Master or Slave device (address 0 to 98) using the EXCH block (when the master is a TSX 37/57 PLC).

The target address coded in the word table associated with the EXCH block must be between 100 and 198 (when the master is a TSX 47/67/87/107 PLC).
The characteristics of the EXCH block are as follows :

- The reception table is obligatory (1 word minimum),
- The word table must be situated in zone %MWi,
- Bit %MSG.D changes to 1 when the response from the slave is received,
- The response is copied into the reception table.

Example of use :

Transmission of the request "Read word" %MW513 to the Master :

Words	Most signif.	Least signif.
%MW10	00	04
%MW11	07	04
%MW12	02	01

Associated program :

LDR %I0.0
AND %MSG.D
[EXCH %MW10:5]

The EXCH block uses %MWi:L as parameters :
- i indicates the number of the first word in the table
- L indicates the number of words in the word table.

When bit %MSG.D = 1 and bit %MSG.E = 0, the table contains the following data :

Words	Most signif.	Least signif.
%MW10	00	**04**
%MW11	07	04
%MW12	02	01
%MW13	**00**	**34**
%MW14	'**AB**'	'**CD**'

The data in **bold** signifies :

- 4 bytes received,
- response code received = 16#0034,
- value of word %MW513 = 16#ABCD.

Other example :

Transmission of the request "Read word" %MW513 to slave 32 :

Words	Most signif.	Least signif.
%MW10	20	04
%MW11	07	04
%MW12	02	01

Associated program :

LDR %I0.0
AND %MSG.D
[EXCH %MW10:5]

The EXCH block uses %MWi:L as parameters :

- i indicates the number of the first word in the table
- L indicates the number of words in the word table.

When bit %MSG.D = 1 and bit %MSG.E = 0, the table contains the following data :

Words	Most signif.	Least signif.
%MW10	20	**04**
%MW11	07	04
%MW12	02	01
%MW13	**00**	**34**
%MW14	'**AB**'	'**CD**'

The data in **bold** signifies :

- 4 bytes received,
- response code received = 16#0034,
- value of word %MW513 = 16#ABCD.

Exchange control

Exchange control is performed using the %MSG function block and system word %SW69.

Bit %MSG.D is set to 1 in the following cases :

- When the response has been received.
- If there is a transmission error (negative acknowledgment)
- If the block is reinitialized.
- If the response does not arrive within 7 seconds (application time-out).

Bit %MSG.D changes to 1 for any other errors (detailed in word %SW69) :

After each exchange, word %SW69 (report of EXCH block) is updated and takes one of the following values :

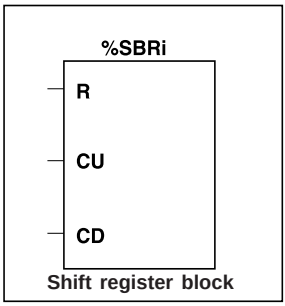
- **0** : Exchange OK.
- **1** : Transmission table too large (LgT>128).
- **2** : Transmission table too small (LgT=0).
- **3** : Word table too small (1).
- **4** : Incorrect Unitelway address (target address does not belong to [0...98] or [100...198] in UNI-TELWAY Slave mode or to [1...5] in UNI-TELWAY Master.
- **5** : Time-out expired.
- **6** : Transmission error (target responds with Nack).
- **7** : Bad ASCII command (command byte <> 0, 1 or 2).
- **8** : Not used
- **9** : Reception error (problem with communication format (speed, parity)).
- **10** : %KWi table prohibited in reception or transmission / reception.

(1) $L < 1 + \frac{LgT + LgT\%2}{2} + \frac{LgR + LgR\%2}{2}$

where L in words

LgT and LgR in bytes

3.4-7 Shift bit register function blocks %SBRi



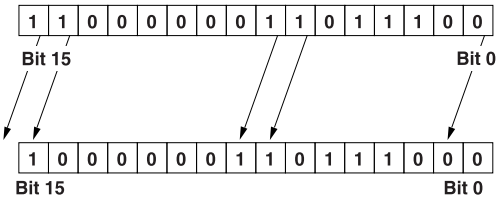
A shift bit register is used to enter binary data (0 or 1) and to move them in one direction or the other.

Characteristics

Register number	**%SBRi**	0 to 7
Register bit	**%SBRi.j**	Bits 0 to 15 (j=0 to 15) of the shift register can be tested by a test instruction and written by an assignment instruction.
Reset input (or instruction)	**R**	On a rising edge, set register bits 0 to 15 %SBRi.j to 0.
Shift to left input (or instruction)	**CU**	On a rising edge, shift a register bit to the left.
Shift to right input (or instruction)	**CD**	On a rising edge, shift a register bit to the right.

Operation

Initial state



CU %SBRi performs a shift to the left

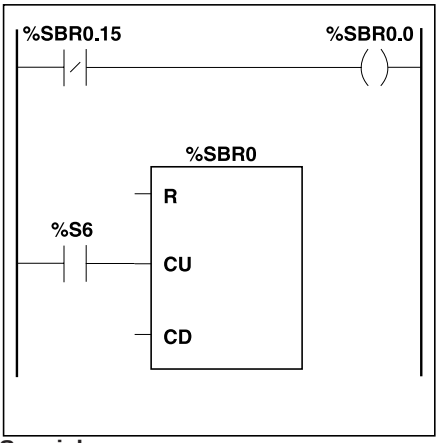
Bit 15 is lost

This is also true of a request to shift a bit to the right (bit 15 to bit 0) using instruction CD. Bit 0 is lost.

If a 16-bit register is not adequate, it is possible, using the program, to cascade several registers.

Programming

Example : Shift a bit to the left every second : bit 0 assumes the opposite state to bit 15.



Reversible programming

```
LDN   %SBR0.15
ST    %SBR0.0
BLK   %SBR0
LD    %S6
CU
END_BLK
```

Non-reversible programming

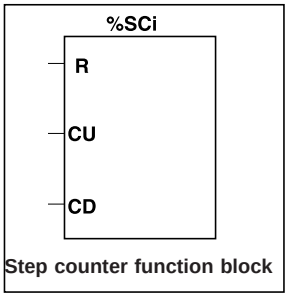
```
LDN   %SBR0.15
ST    %SBR0.0
LD    %S6
CU    %SBR0
```

Special cases

- **Effect of a cold restart :** (%S0=1)
  - sets all the bits of the register word to 0

- **Effect of a warm restart :** (%S1=1) has no effect on the bits of the register word.

3.4-8 Step counter function blocks %SCi



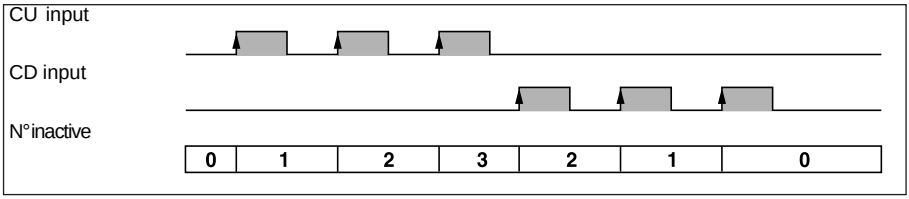
Step counter function block

A step counter is a series of steps to which actions can be assigned. Moving from one step to another depends on external or internal events. Each time that a step is active, the associated bit is set to 1. Only one step of a step counter can be active at a time.

Characteristics

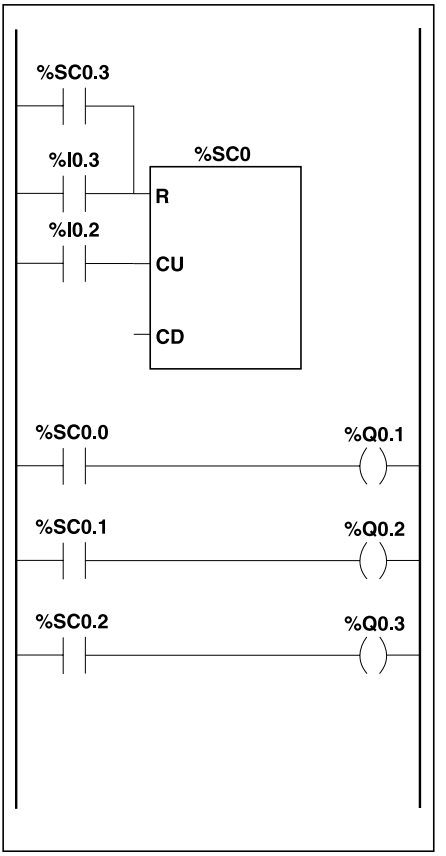
Step counter number	**%SCi**	0 to 7
Step counter bit	**%SCi.j**	Step counter bits 0 to 255 (j=0 to 255) can be tested by an LD logical operation and written by an assignment instruction.
Reset input (or instruction)	**R**	On a rising edge, resets the step counter.
Increment input (or instruction)	**CU**	On a rising edge, increments the step counter by one step.
Decrement input (or instruction)	**CD**	On a rising edge, decrements the step counter by one step.

Operation



Programming

Example : To program the step counter 0 incremented by input %I0.2. It is reset to 0 by input %I0.3 or when it arrives at step 3.
Step 0 controls output %Q0.1, step 1 controls output %Q0.2 and step 2 controls output %Q0.3.



Reversible programming

```
BLK    %SC0
LD     %SC0.3
OR     %I0.3
R
LD     %I0.2
CU
END_BLK
LD     %SC0.0
ST     %Q0.1
LD     %SC0.1
ST     %Q0.2
LD     %SC0.2
ST     %Q0.3
```

Non-reversible programming

```
LD     %SC0.3
OR     %I0.3
R      %SC0
LD     %I0.2
CU     %SC0
LD     %SC0.0
ST     %Q0.1
LD     %SC0.1
ST     %Q0.2
LD     %SC0.2
ST     %Q0.3
```

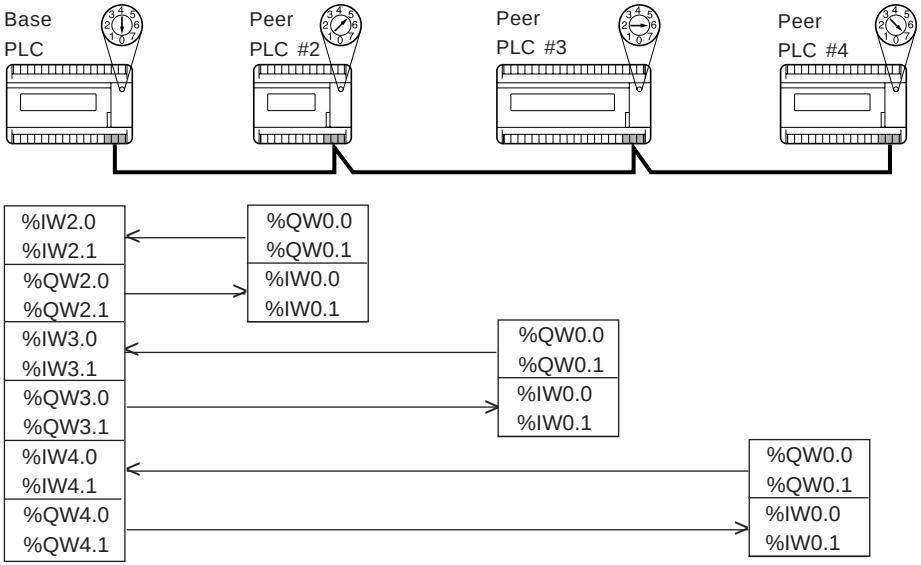
Special cases

- **Effect of a cold restart :** (%S0=1)
  - initializes the step counter

- **Effect of a warm restart** : (%S1=1) has no effect on the step counter.

3.5 Communication between PLCs

Words %IW and %QW are used to exchange data between PLCs. The diagram below illustrates the exchange words for each PLC.



These exchange words are updated automatically when the PLCs are in RUN mode. The user program for each PLC is limited to :

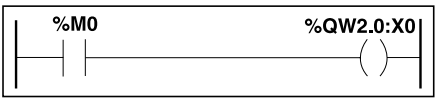
- Writing to output words %QWi.j
- Reading input words %IWi.j

The update cycle for %IW/%QW words is synchronized with the PLC scan. System bit %S70 is set to 1 when a complete update cycle has taken place. It is reset to 0 by the program or the terminal.

Bits %S71 / %S72 and word %SW71 are also used to control data exchanges (see Section 5).

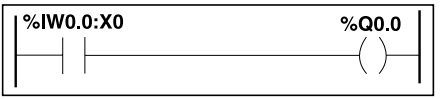
Note : Each PLC address is defined according to the position of the selector switch located on the PLC front panel, and this position is taken into account on every power-up.

Example 1 : The base PLC transmits data to peer PLC #2 indicating the end of production (bit %M0=1). Once it has received this data, the peer PLC starts a materials handling machine by activating output %Q0.0.



Base PLC programming

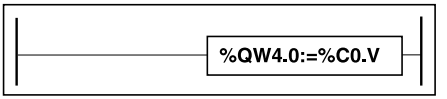
```
LD    %M0
ST    %QW2.0:X0
```



Peer PLC programming

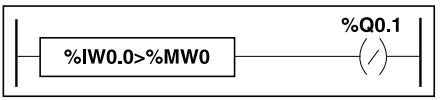
```
LD    %IW0.0:X0
ST    %Q0.0
```

Example 2 :
The base PLC transmits the current value of the counter 0 to peer PLC #4. When this current value is higher than the threshold contained in word %MW0, the peer PLC stops a machine by deactivating output %Q0.1.



Base PLC programming

```
LD    1
[%QW4.0:=%C0.V]
```



Peer PLC programming

```
LD    [%IW0.0>%MW0]
STN   %Q0.1
```

B

4.1 Presentation

TSX 07 30/31 •• •• "base PLC" PLCs with discrete I/O (starting from version V3) are able to manage analog I/O modules.

There are three types of analog module in the TSX Nano offer :

- **TSX AMN 4000/4001 analog I/O modules** :

  These modules communicate with the "base PLC" PLC via the I/O extension link. They are managed via exchange words %IW and %QW.

• Input module :
These modules perform a voltage/frequency conversion which requires the use of input %I0.0 in frequency meter mode on the PLC. A single input module can be connected by the PLC. The module is managed from the application program via system words %SW100 and %SW101.

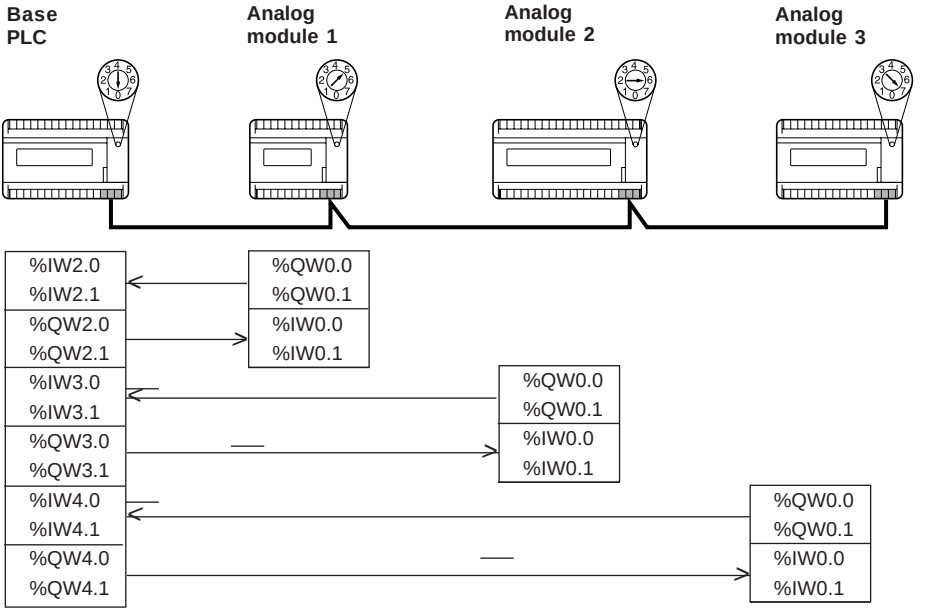
• Output module :
These modules perform a PWM/voltage conversion, by integrating the signal sent to output %Q0.0 of the PLC (pulse width modulation). A single module can be connected per PLC (base PLCs fitted with transistor outputs).
The module is managed from the application program via system words %SW102 and %SW103.

4.2 TSX AMN 4000/4001 analog modules

4.2-1 Operating principle of analog modules

B

Words %IW and %QW are used to exchange application data between a "base PLC" PLC and TSX AMN 400• analog modules.
This data, which is limited to four words (two in read mode, two in write mode) for both the "base PLC" PLC and analog modules, can be exchanged in both directions.
The figure below shows the words exchanged for each PLC.



These exchange words are automatically updated when the "base PLC" PLC is in RUN mode. For each PLC, the user program is limited to :

- writing to output words %QWi.j
- reading input words %IWi.j

The updating scan for %IW/%QW words is synchronous with the PLC scans. System bit %S70 is set to 1 when a complete scan has taken place. It is reset by the program or terminal.

Note

The address of each PLC is defined according to the position of the selector switch located on the PLC's front panel. Its position is taken into account whenever the PLC is powered up.

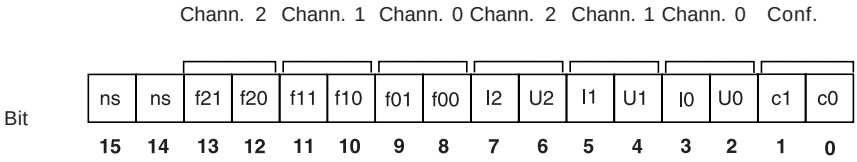
4.2-2 Programming analog modules

TSX AMN 400• analog I/O modules are programmed using 2 sets of exchange words : %QWi.0, %QWi.1 and %IWi.0, %IWi.1.

Exchange word %QWi.0

This word is used to configure the analog input channels of a module.
It contains :

- the number of input channels used,
- the type of input for each channel,
- the type of filtering for each channel.



C1	C0	Number of input channels used
0	0	None
0	1	Channel 0
1	0	Channel 0 and channel 1
1	1	Channel 0, channel 1 and channel 2

In	Un	Type of input for channel n
0	0	Voltage input +/-10 V
0	1	Voltage input 0..10 V
1	0	Current input 0..20 mA
1	1	Current input 4..20 mA

fn1	fn0	Type of filter for channel n (digital filter of the 1st order)
0	0	Hard filtering
0	1	75 ms
1	0	300 ms
1	1	1.5 s

ns : not significant

Exchange word %QWi.1

This 16-bit word contains the value of analog output 0 in a module.

Bit n°	Meaning
x0 to x14	Value of output 0 coded on 15 bits
x15	Sign bit

Exchange word %IWi.0

This 16-bit word contains the value of analog input 0 and the module status.

Bit n°	Meaning
x0 to x10	Value of input 0 coded on 11 bits
x11	Sign bit of channel 0
x12 = 1	Self-calibration fault
x13 = 1	Stop overshoot channel 0
x14 = 1	Stop overshoot channel 1
x15 = 1	Stop overshoot channel 2

Exchange word %IWi.1

This 16-bit word either contains the value of analog input 1 or the values of analog inputs 1 and 2 in a module.

Bit n°	Meaning when inputs 0 and 1 are configured	Meaning when inputs 0, 1 and 2 are configured
0 to 6		Value of input 1 coded on 7 bits
7	Value of input 1 coded on 15 bits	Sign bit of channel 1
8 to 14		Value of input 2 coded on 7 bits
15	Sign bit of channel 1	Sign bit of channel 2

Range of values of input modules

Type of input	Range of values 7 bits + sign	11 bits + sign	Overshoot detection threshold for low and high stops
-10 V / +10 V	-125 ..125	-2000 .. 2000	+/- 2.5% of the full scale
0 / 10 V	0 .. 125	0 .. 2000	+/- 2.5% of the full scale
0 / 20 mA	0 .. 125	0 .. 2000	+/- 2.5% of the full scale
4 / 20 mA	25 .. 125	0 .. 2000	+/- 2.5% of the full scale

All stop overshoot faults are signaled by setting one of the status bits in word %IWi.0 of the module to 1. For the 4/20 mA range, a fault is declared if the current is less than 3.5 mA, but the module is capable of feeding back values up to -0.5 mA.

Range of values of output modules

Type of output	Values written in word %QWi.1	Max. overshoot values of low and high stops
-10 V / +10 V	-2000 .. 2000	+/- 2.5% of the full scale
0 / 10 V	0 .. 2000	+/- 2.5% of the full scale
0 / 20 mA	0 .. 2000	+/- 2.5% of the full scale
4 / 20 mA	400 .. 2000	+/- 2.5% of the full scale

If an internal fault occurs, the output is set to fallback mode by the module (0 V for the voltage output, 0 mA for the current output, whatever the range).

Warning

When the PLC is set to STOP, the analog output is maintained at the value it had before the STOP. There is no automatic fallback in this case.

4.2-3 Using %IW words in the user program

In order for %IW exchange words to be manipulated by PL7-07 instructions, they must be converted to a 15-bit + sign format.

The table overleaf shows the conversion operations to be inserted in the user program :

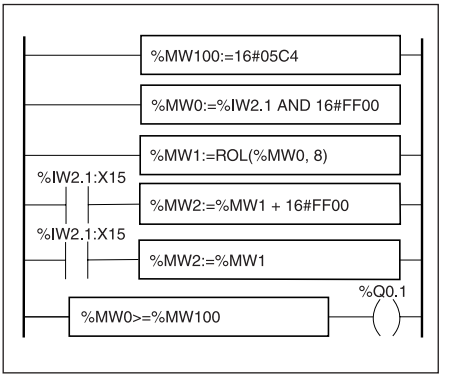
Value of the sign bit	7-bit word + sign	11-bit word Channel n° + sign	Conversion to 15-bit + sign format
0		%IWi.0	%MW0:= %IWi.0 AND 16#0FFF
1		%IWi.0	%MW0:= %IWi.0 OR 16#F000
0	%IWi.1	1	%MW0:= %IWi.1 AND 16#00FF
1	%IWi.1	1	%MW0:= %IWi.1 AND 16#00FF %MW1:= %MW0 OR 16#FF00
0	%IWi.1	2	%MW0:= %IWi.1 AND 16#FF00 %MW1:= ROR (%MW0,8)
1	%IWi.1	2	%MW0:= %IWi.1 AND 16#FF00 %MW1:= ROR (%MW0,8) %MW2:= %MW1 OR 16#FF00

Note :
Internal words %MW0, %MW1 and %MW2 are used as examples of logging variables in the conversion operations.

Example of a user program

Compares the analog value of channel 2 in module 1, all the inputs of which are configured, with a threshold contained in word %MW100 (hexadecimal value 16#05C4). Output Q0.1 in the "base PLC" PLC is set to 1 when the value is equal to or greater than the threshold.

```
LD   1
[%MW100 := 16#05C4]
LD   1
[%MW0 := %IW2.1 AND 16#FF00]
LD   1
[%MW1 := ROR ( %MW0, 8 )]
LD   %IW2.1:X15
[%MW2 := %MW1 OR 16#FF00]
LDN %IW2.1:X15
[%MW2 := %MW1]
LD   [%MW0 >= %MW100]
ST   %Q0.1
```



4.2-4 Diagnostics of communication status with analog modules

These diagnostics are identical to those carried out on a peer PLC and are indicated by word %SW71 (see Part B Section 6.2-2).

4.3 Analog input modules TSX ASN •••

4.3-1 Configuring analog inputs

Using input %I0.0 in frequency meter mode to connect the analog input module requires the following parameter settings :

- **Fast counter type** : to be set to Frequency to use the input in frequency meter mode
- **Max. frequency :** to be set to 10 KHz

4.3-2 Programming analog inputs

The input module is programmed using the two system words %SW100 and %SW101 and by enabling the fast counter %FC (see example in Section 4.4-4).

- **%SW100 :** analog input functions command word
- **%SW101 :** acquired value of analog input

The operating mode is selected by writing, via the program, word %SW100 and the acquired value of the analog input can be read in word %SW101.

These two words are reset to zero by the system on a cold restart.

Depending on the choice of operating mode, the system offers a scaling function. This scaling is within a range of 0 to +10 000 for unipolar modules (4/20mA and 0/10V input modules) and -10 000 to +10 000 for bipolar modules (-10/+10V).

%SW100	Operation	Value range of %SW101
0	Disable analog input function on %I0.0	0
1	Operate without scaling	0...1 000 Period of measurement 125 ms
2	Scale for unipolar range (4/20mA, 0/10V)	0...10 000 Period of measurement 125 ms
3	Scale for bipolar range (-10/+10V)	-10 000 ... +10 000 Period of measurement 125 ms
4	Scale for unipolar range (4/20mA, 0/10V)	0...10 000 Period of measurement 500 ms
5	Scale for bipolar range (-10/+10V)	-10 000 ... +10 000 Period of measurement 500 ms

The raw or scaled analog value is available in %SW101 if %SW100 is written with a value of 1 to 5. The validity of this measurement can be checked using system bit %SW111:X3 (set to 1 by the system if the measurement is valid).
If system bit %SW111:X3 is set to zero by the application, an analog acquisition function is launched, measurement acquisitions continue to be made when the PLC is in STOP.

The raw frequency measurement remains available in word %FC,V associated with input %I0.0 but it is a measurement which is a function of the measurement period (eg : the full scale 8KHz will give 1000 for 125ms and 4000 for 500ms). It is therefore recommended, to simplify the application program, that system word %SW101 is used by preference.

Note

The period of measurement can be modified during operation by rewriting word %SW100 but this mode of use is not recommended because the first measurement after the change in period may be wrong.

Using analog input modules on V2 PLCs

It is possible to use analog input modules on TSX 07 2••• V2 PLCs, according to the following rules :

- use of input %I0.0 as frequency meter (enabling of operation by instruction IN %FC)
- configuration of the measurement period by the application writing bit %SW111:X2.
  %SW111:X2=0 measurement every second (default)
  %SW111:X2=1 measurement every 100 ms
- the image of the analog input value is available in object %FC,V;
  the value is interpreted in the following way :

Range	Formula
0/10V	$U(V) = 1.25 \times (\%FC,V \times 10^{-3})$
4/20 mA	$I(mA) = 2 \times [(\%FC,V \times 10^{-3}) + 2]$
-10/+10V	$U(V) = 2.5 \times [(\%FC,V \times 10^{-3}) - 4]$

Note
In the case of a 4/20 mA module, the frequency is zero between 0 and 4 mA.

4.3-3 Response time of analog inputs

The response time IRT in acquiring an analog input, between the actual variation of the electrical value at the module terminals and the correspondence in word %SW101 of the measured value, depends essentially on the measurement period selected (125/500 ms) and less so on the PLC scan time. The variation of the electrical value has a negligible effect on the response time.

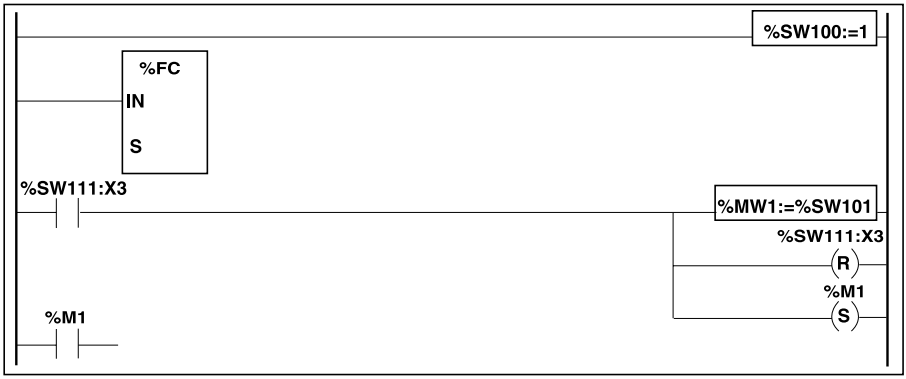
- For acquisition at a period of 125 ms : IRT is less than 500 ms.
- For acquisition at a period of 500 ms : IRT is less than 1.2 s.

Note

A change in the operating mode (changing %SW100) is taken into account by the system on each scan or immediately on a rising edge of "IN %FC". The measurements are linked in realtime one after another at the chosen period (125 ms or 500 ms). The result of the last measurement taken is transmitted at the beginning of the PLC scan to %SW101. The value of this word does not change during the PLC scan.

4.3-4 Example of analog input programming

```
(* ENABLE ANALOG INPUT FUNCTIONS *)
LD      1                   (* ANA INPUT MODE 0..1000 ON 125 MS *)
[%SW100 := 1]               (* SELECTED MODE TAKEN INTO ACCOUNT *)
BLK %FC
LD  1
IN
END_BLK
(* ACQUISITION OF MEASUREMENT*)
LD      %SW111:X3           (* MEASUREMENT VALID *)
[%MW1 := %SW101]            (* MEMO MEASUREMENT *)
R       %SW111:X3           (* MEASUREMENT ACKNOWLEDGED*)
S       %M1                 (* MEASUREMENT INDEX VALID *)
(* USE OF THE MEASUREMENT, DEPENDING ON THE APPLICATION *)
LD      %M1
...
```



4.3-5 Characteristics of analog inputs

Type	Input value	%SW101 value period 125ms	Resolution (1) /increment	%SW101 value period 500ms	Resolution (1) /increment
4/20mA	4mA 12mA 20mA	0 5000 10000	16µA/10 lsb	0 5000 10000	4µA/2.5 lsb
0/10V	0V 10V	0 10000	10mV/10 lsb	0 10000	2.5mV/ 2.5 lsb
-10/+10V	-10V +10V	-10000 10000	20mV/10 lsb	-10000 10000	5mV/ 2.5 lsb

The values of %SW101 are given for the operating mode with scaling.

(1) Resolution : Minimum value of the input variation to obtain a measurement variation. The measurement variation varies in "steps" called increments.

4.4 Analog output modules TSX AEN •••

4.4-1 Configuring analog outputs

Using output %Q0.0 to connect the analog output requires the following parameter settings :

- **Output %Q0.0 :** to be used for pulse width modulation %PWM.
- **Time base :** to be set to 0.1 ms.
- **Preset :** must be set to 249 so that operation remains enabled after a warm restart.
  Adjustment of this parameter is only appropriate for "base PLC" TSX Nano PLCs (V3.0).

4.4-2 Programming analog outputs

The output module is programmed using the two system words %SW102 and %SW103 and by enabling output %PWM (see example in Section 4.3-4).

- **%SW102 :** analog output functions command/status word
- **%SW103 :** value of analog output to be generated

The operating mode is selected by writing, by the application program, word %SW102 and the value of the analog output to be generated must be written in word %SW103.

These two words are reset to zero by the system on a cold restart.

Depending on the choice of operating mode, the system offers a scaling function. This scaling is within a range of 0 to +10 000 for unipolar modules (4/20mA and 0/10V input modules) and -10 000 to +10 000 for bipolar modules (-10/+10V).

%SW102	Operation	Value range of %SW103
0	Disable analog output function on %Q0.0	Not used
1	Operate without scaling	5...249
2	Scale for unipolar range (4/20mA, 0/10V)	0...10 000
3	Scale for bipolar range (-10/+10V)	-10 000 ... +10 000

The effective resolution of the analog outputs is 245 points.

If the value written in %SW103 is less than the minimum value (eg : less than 0 in unipolar mode), the minimum value in the range will be applied to the output module.
If the value written in %SW103 is greater than the maximum value (eg : greater than 10000 in unipolar mode), the maximum value in the range will be applied to the output module.
These two types of programming error are not indicated to the application.

Important

In the fallback conditions of discrete outputs, the PWM is no longer generated and a signal is no longer sent to the output modules.
From this, the bipolar modules take their lowest value (-10V).
This fallback mode must be taken into account by the user.

4.4-3 Response time of analog outputs

The response time ORT of an analog output, between writing the setpoint in word %SW103 and reaching the voltage (and/or current) corresponding to the module terminals, depends on the amplitude of the variation and the PLC scan time.

- For a full scale variation, ORT is less than 500 ms.

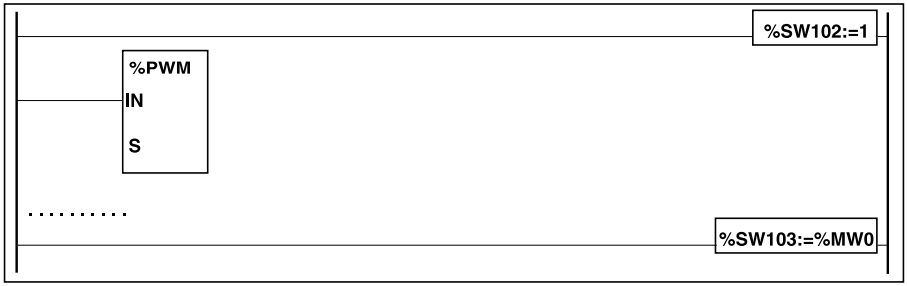
The shorter the PLC scan time and the lower the setpoint variation, the shorter the response time will be. For a scan time of 10 ms and a variation of1/10th of the full scale, this response time will go down to approximately 50 ms.

Note

A change in the operating mode (changing %SW102) is taken into account on a rising edge of input IN of the PWM (execution of the instruction "IN %PWM") or if setpoint %SW103 is changed. A change of setpoint (%SW103) is taken into account by the system on each scan and becomes effective from the next application program scan (Max. delay of 3ms).

4.4-4 Example of analog output programming

```
(* ENABLE ANALOG OUTPUT FUNCTION *)
LD      1
[%SW102 := 1]           (* ANA OUTPUT RAW MODE 5..249  *)
IN %PWM                 (* MODE TAKEN INTO ACCOUNT *)

(* GENERATION OF SETPOINT IN PWM *)
....                    (* SETP. CALC. DEPENDING ON APPLICATION *)
LD      1
[%SW103 := %MW0]        (* APPLICATION OUTPUT SETPOINT *)
```



4.4-5 Characteristics of analog outputs

Type	Value to be generated at output	%SW103 value to be written by application program	Resolution output module	Isb resolution (1) %SW103
4/20mA	4mA	0	65µA	40
	12mA	5000		
	20mA	10000		
0/10V	0V	0	40mV	40
	5V	5000		
	10V	10000		
-10/+10V	-10V	-10000	81mV	81
	0V	0		
	+10V	10000		

Values of %SW103 are given for the operating mode with scaling.

(1) LSB resolution : Minimum variation to be applied to %SW103 to obtain a variation of the output module equal to its resolution.

B

5.1 Introduction

TSX 07 •1 16/24 •• PLCs have a time-of-day clock which is used to perform three functions :

- **Schedule blocks** are used to control actions at predefined or calculated times.
- **Time/date stamping** is used to date events and **measure duration**.

The TSX Nano schedule blocks and the time-of-day clock can be accessed from the TSX PLC operations menu. Additionally, the time-of-day clock can be set by the program. They continue to operate for up to 30 days when the PLC is switched off if the battery has been charged for at least 6 consecutive hours before the PLC is switched off.

The time-of-day clock has a 24-hour format and takes leap years into account.

5.2 Schedule blocks

The schedule block is used to control actions at a particular predefined time.
A maximum of 16 schedule blocks can be used, each one performing this function. These blocks do not require any program entry. They can be configured (see operating modes, Part C).

5.2-1 Characteristics

Schedule block number	**RTC : n**	n=0 to 15
Output	**Q** :	Output assignment is activated by schedule block : %Mi or %Qj.k. This output is set to 1 when the current date and time are between the setting of the start of the active period and the setting of the end of the active period.
Start date	**DD:MMM**	Shows the day from 1 to 31 and the month Jan to Dec of the start of activation of schedule block.
End date	**DD:MMM**	Shows the day from 1 to 31 and the month Jan to Dec of the end of activation of schedule block.
Day	**MTWTFSS**	Shows the days of activation in the week (M : Monday to S : Sunday)
Start hour	**hh:mm**	Shows in hours (0 to 23) and minutes (0 to 59) the start of activation of schedule block.
End hour	**hh:mm**	Shows in hours (0 to 23) and minutes (0 to 59) the end of activation of schedule block.

The bits of system word %SW114 enable (bit set to 1) or disable (bit set to 0) the operation of each of the function blocks.

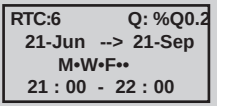
By default (or after a cold restart) all bits of this system word are set to 1. **Use of these bits by the program is optional**.

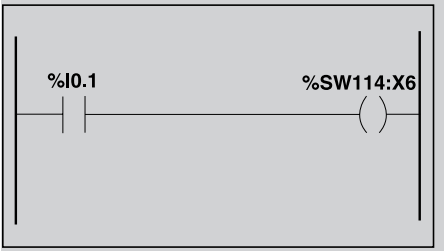
Comments :

- If the same output (%Mi or %Qj.k) is assigned by several blocks, it is the "OR" of the results of each of the blocks which is finally assigned to this object (which means it is possible to have several "operating ranges" for the same output).

Example : Configuration of a schedule block, for summer month spray program.

- RTC **6** : schedule block #6
- Q : **%Q0.2** : output activated by the schedule block
- **21 -Jun -> 21-Sep** : period of activation
- **M•W•F••** : activation days (Monday, Wednesday and Friday)
- **21 : 00 - 22 : 00** : scheduled activation period






```
LD %I0.1
ST %SW114:X6
```

In this example the user can disable the schedule block using a switch or a humidity detector wired to input %I0.1.

Note :
It is important to check the state of bit %S51 which shows any schedule block errors.

5.2-2 Time dating by program

Date and time are both available in system words %SW50 to %SW53 (see Section 6.2). It is therefore possible to perform time and date stamping by the PLC program by making arithmetic comparisons between the current date and time and the immediate values or words %MWi (or %KWi) which can contain setpoints.

5.3 Time/date stamping

System words %SW50 to %SW53 (see Section 6.2) contain the current date and time in BCD format, which is useful for display on or transmission to a peripheral device. These system words can be used to store the date and time of an event.

To date an event it is sufficient to use assignment operations, to transfer the contents of system words to internal words and then process these internal words (for example, transmission to display unit by EXCH instruction).

Example :


```
...
LDR   %I0.1
[%MW12:4 :=%SW50:4]
...
```

Once an event is detected, the word table contains :

Coding :	**Most significant byte**	**Least significant byte**
%MW12	Second	Day of the week(1)
%MW13	Hour	Minute
%MW14	Month	Day
%MW15	Century	Year

Example :	Monday 19 April 1994
In hex	13H, 40min, 30s
3000	30s, 0=Monday
1340	13H, 40min
0419	4=April, 19
1994	1994

(1) Where 0=Monday, 1=Tuesday, 2=Wednesday, 3=Thursday, 4=Friday, 5=Saturday, 6=Sunday

Reading the date and time of the last stop using system words
System words %SW54 to %SW57 (see Section 6.2) contain the date and time of the last stop, and word %SW58 contains the code showing the cause of the last stop, in BCD format.

5.4 Setting the date and time

5.4-1 Setting the date and time using the terminal

When the programming terminal is in TSX (online) mode, the date and time can be updated quickly and easily (see operating modes, Part C).

5.4-2 Updating date and time using system words

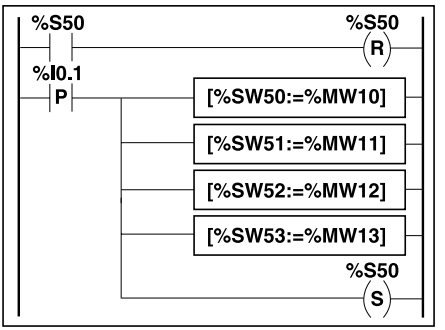
System words offer 2 other possibilities for updating the date and time :

Updating using system words %SW50 to %SW53

(see Section 6.2)

For this, bit %S50 must be set to 1. This bit :

- cancels the update of words %SW50 to %SW53 via the internal clock
- transmits the values written in words %SW50 to %SW53 to the internal clock.




```
LD    %S50
R     %S50
LDR   %I0.1
[%SW50:=%MW10]
[%SW51:=%MW11]
[%SW52:=%MW12]
[%SW53:=%MW13]
S     %S50
```

Words %MW10 to %MW13 will contain the new date and time in BCD format and will correspond to the coding of words %SW50 to 53.

The word table must contain the new date and time.

Coding :	Most significant byte	Least significant byte
%MW10	Second	Day of the week (1)
%MW11	Hour	Minute
%MW12	Month	Day
%MW13	Century	Year

Example : Monday 19 April 1994

Hex	13H 40min 30s
3000	30s, 0=Monday
1340	13H, 40min
0419	4=April, 19
1994	1994

(1) Where 0=Mon., 1=Tues., 2=Wednes., 3=Thurs., 4=Fri., 5=Satur., 6=Sun.

To ensure that the realtime clock is updated on a TSX Nano V1, V2 or V3.0 at the changeover to the year 2000, the PLC must be powered up during this time. It is however, possible to update the realtime clock via the program.

To do this, add the following to the application :

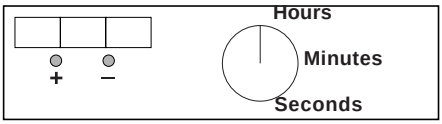
```
LD  [%SW53]=16#1900
ST %S50
[%SW53:=16#2000]
```

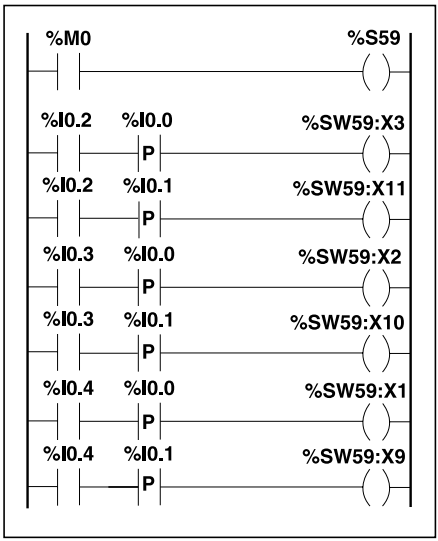
Updating using system word %SW59

Another method of updating is to use enable bit %S59 and adjustment word %SW59.

Setting bit %S59 to 1 enables adjustment of the current date and time by word %SW59. Word %SW59 is described in Section 6.2. It increments or decrements each of the date and time components on a rising edge.

Example : A front panel is created in order to modify the hour, minutes and seconds of the internal clock.






```
LD    %M0
ST    %S59
LD    %I0.2          (hour)
ANDR  %I0.0
ST    %SW59:X3
LD    %I0.2
ANDR  %I0.1
ST    %SW59:X11
LD    %I0.3          (minute)
ANDR  %I0.0
ST    %SW59:X2
LD    %I0.3
ANDR  %I0.1
ST    %SW59:X10
LD    %I0.4          (second)
ANDR  %I0.0
ST    %SW59:X1
LD    %I0.4
ANDR  %I0.1
ST    %SW59:X9
```

- Inputs %I0.2, %I0.3 and %I0.4 are controlled by the Hours/Minutes/Seconds switch.
- Incrementation is performed by input %I0.0, pushbutton +.
- Decrementation is performed by input %I0.1, pushbutton -.

B

6.1 System bits

6.1-1 List of system bits

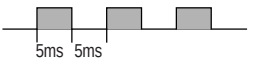
Bit	Function	Init state	Control
%S0	1 = cold restart (power return with loss of data)	0	S or U->S
%S1	1 = warm restart (power return without loss of data)	0	S or U->S
%S4, %S5	Time base 10ms, 100ms	-	S
%S6, %S7	Time base 1s, 1min	-	S
%S8	0 = outputs saved when the PLC stops	1	U
%S9	1 = PLC outputs reset to zero if the PLC is running	0	U
%S10	0 = I/O fault	1	S
%S11	1 = watchdog overflow	-	S
%S13	1 = first scan after setting to RUN	1	S
%S17	1 = overflow of an unsigned arithmetic operation or rotate shift	0	S->U
%S18	1 = arithmetic overflow or error	0	S->U
%S19	1 = scan period overshoot	0	S->U
%S20	1 = index overflow	0	S->U
%S21	1 = Grafcet initialization : sets the steps to 0 and the initial steps to 1	0	U->S
%S22	1 = reset Grafcet to zero	0	U->S
%S23	1 = enable Grafcet presetting. If held at 1, Grafcet freezes	0	U->S
%S49	1 = request to reset every 10s transistor outputs which have been tripped by overcurrent or short-circuit	0	U->S
%S50	1 = set schedule block (RTC)	0	U
%S51	1 = schedule block not initialized or error 0 = schedule block are set	0	S
%S59	1 = adjust current date	0	U
%S69	1 = display internal bits	0	U
%S70	1 = update exchange words %IW/OW on extension. Process Modbus request	0	S
%S71	1 = exchange via extension link	0	S
%S72	0 = scanning of peer PLCs	0	U
%S100	State of /DPT	-	S
%S118	1 = base PLC fault	0	S
%S119	1 = peer PLC fault	0	S

S = controlled by the system, U = controlled by the user, U->S = set to 1 by the user, reset to 0 by the system, S->U = set to 1 by the system, reset to 0 by the user.

6.1-2 Detailed description of system bits

TSX 07 PLCs have system bits %Si which indicate the status of the PLC or enable the user to intervene in its operation.

These bits can be tested in the user program in order to detect any operating event which requires special processing. Some of them must be reset to their initial or normal state by the program. However, the system bits which have been reset to their initial or normal state by the system must not be reset by the program or the terminal.

System bits	Function	Description
%S0	Cold restart	Normally at 0. It is set to 1 by : • A power return with loss of data (battery fault) • The user program • The terminal (in the Data editor). This bit is set to 1 during the first complete scan. It is reset to 0 before the next scan. For operating information : see Part A, Section 7.1.
%S1	Warm restart	Normally at 0. It is reset to 1 by : • A power return with saving of data • The user program • The terminal (in the Data editor). It is reset to 0 by the system at the end of the first complete scan and before updating of the outputs. For operating information : see Part A, Section 7.1.
%S4 **%S5** **%S6** **%S7**	Time bases 10ms 100ms 1s 1min	Changes in the status of these bits are controlled by an internal clock. They are not synchronized with the PLC scan. **Example** : %S4 

System bits	Function	Description
%S8	Output freeze	Initially at 1, it can be set to 0 by the program or by the terminal (in the Data editor) : • At state 1 PLC outputs are set to 0 if the program is not being executed normally or if the PLC changes to STOP • At state 0 PLC outputs are kept in their existing state if a program operation fault occurs or if the PLC changes to STOP.
%S9	Reset outputs	Normally at 0. It can be set to 1 by the program or by the terminal (in the Data editor) : • At state 1 outputs are forced to 0 when the PLC is in RUN mode • At state 0 outputs are updated normally.
%S10	I/O fault	Normally at 1. It is set to 0 when a I/O fault is detected on the base PLC or the peer PLC (configuration fault, exchange fault, hardware fault). Bits %S118 and %S119 indicate in which PLC the fault is and words %SW118 and %SW119 detail the nature of the fault (see Section 5.2). Bit %S10 is reset to 1 when the fault disappears.
%S11	Watch-dog overflow	Normally at 0, it is set to 1 by the system when the program execution time (scan time) exceeds the maximum scan time (software watchdog). Watchdog overflow causes the PLC to change to STOP.
%S13	First scan	Normally at 0, it is set to 1 by the system during the first scan after the PLC has been changed to RUN.
%S17	Carry overflow	Normally at 0, it is set to 1 by the system : • In the case of carry overflow during a non-signed arithmetic operation (remainder) • During a rotate or shift operation it indicates the output of a bit at 1. It must be tested by the user program after each operation where there is a risk of overflow, then reset to 0 by the user if an overflow occurs.

System bits	Function	Description
%S18	Arithmetic overflow or error	Normally at 0. It is set to 1 in the case of an overflow when a 16 bit operation is performed, that is : • A result greater than + 32767 or less than - 32768 • Division by 0 • The square root of a negative number • BTI or ITB conversion not significant : BCD value out of limits It must be tested by the user program after each operation where there is a risk of an overflow, then reset to 0 by the user if an overflow occurs.
%S19	Scan period overrun (periodic scan)	Normally at 0, this bit is set to 1 by the system in the event of a scan period overrun (scan time greater than than the period defined by the user at configuration or programmed in %SW0). This bit is reset to 0 by the user.
%S20	Index overflow	Normally at 0, it is set to 1 when the address of the indexed object becomes less than 0 or more than the maximum size of an object. It must be tested by the user program after each operation where there is a risk of overflow, then reset to 0 if an overflow occurs.
%S21	GRAFCET initialization	Normally at 0, it is set to 1 by : • A cold restart, %S0=1 • The user program, in the preprocessing program part only, using a Set Instruction (S %S21) or a set coil -(S)- %S21. • The terminal. At state 1, it causes GRAFCET initialization. Active steps are deactivated and initial steps are activated. It is reset to 0 by the system after GRAFCET initialization.

System bits	Function	Description
%S22	GRAFCET reset	Normally at 0, it can only be set to 1 by the program in pre-processing. At state 1 it causes the active steps of the entire GRAFCET to be deactivated. It is reset to 0 by the system at the start of the execution of the sequential processing.
%S23	Preset and freeze GRAFCET	Normally at 0, it can only be set to 1 by the user program in the pre-processing program module. At state 1, it validates the presetting of the GRAFCET chart. Maintaining this bit at 1 freezes the GRAFCET (freezes the chart). It is reset to 0 by the system at the start of the execution of the sequential processing to ensure that the GRAFCET chart moves on from the frozen situation.
%S49	Reactivate transistor outputs	Normally at 0, it is set to 1 by the user to request a reactivation every 10 s of the transistor outputs which have been tripped by an overcurrent or short-circuit, the first attempt occurring 10 s after the outputs have been tripped.
%S50	Updating the date and time using words %SW50 to 53	Normally at 0, this bit can be set to 1 or to 0 by the program or the terminal. • At 0, the date and time can be read. • At 1, the date and time can be updated.
%S51	Time-of-day clock status	• At 0, the date and time are set. • At 1, the date and time must be set by the user. When this bit is set to 1, the time of day clock data is not valid. The date and time may never have been configured, or the battery may be low.
%S59	Updating the date and time using word %SW59	Normally at 0, this bit can be set to 1 or to 0 by the program or the terminal. • At 0, the date and time remain unchanged. • At 1, the date and time are incremented or decremented according to the control bits set in %SW59.
%S69	Displaying internal bits on the front panel of the PLC	Normally at 0, this bit can be set to 1 or to 0 by the program or the terminal. • At 0 : the PLC LEDs display the status of the I/O • At 1 : the status of 8 internal bits (TSX 07 10, 14 and 16 I/O) or 16 internal bits (TSX 07 20 and 24 I/O) is displayed on the PLC LEDs (see Part A, Section 1.9). The LED farthest to the right flashes to indicate that displaying internal bits has been selected.

System bits	Function	Description
%S70	Update of exchange words Process Modbus request	For the base PLC, this bit is set to 1 as soon as the PLC has performed a complete cycle of transmitting exchange words %IW/%QW to the peer PLCs. For each peer PLC, this bit is set to 1 as soon as the peer PLC has received and transmitted the exchange words with the base PLC. This bit is reset to 0 by the program or by the terminal. This bit is set to 1 when a Modbus request is processed. It can be used by the operator. This bit is reset to zero by the program or terminal.
%S71	Exchange via extension link	Initially set to 0. Set to 1 until an exchange via the extension link is detected. This bit is set to 0 when no exchange is performed via the extension link. Word %SW71 of the base PLC shows the list and status of the available extensions.
%S72	Scanning of peer PLCs	Applies only to PLCs version 2.2 or lower. Normally at 0. Can be set to 1 by the program or the terminal. • At 0 the base PLC communicates with the peer PLCs • At 1 communication is disabled
%S100	State of /DPT signal	Shows the state of INL/DPT strap on the terminal port : • No strap : UNI-TELWAY master protocol (%S100 = 0) • Strap present : (/DPT at 0V) protocol defined by the application configuration (%S100 = 1)
%S118	Base PLC fault	Normally at 0. It is set to 1 when an I/O fault is detected on the base PLC. Word %SW118 determines the nature of the fault. Bit %S118 is reset to 0 when the fault disappears.
%S119	Peer PLC fault	Normally at 0. It is set to 1 when an I/O fault is detected on the I/O extension. Word %SW119 determines the nature of the fault. Bit %S119 is reset to 0 when the fault disappears.

6.2 System words

6.2-1 List of system words

Word	Function	Control
%SW0	Value of PLC scan period (periodic task)	U
%SW11	Software watchdog time	S
%SW14	UNITELWAY time-out	S and U
%SW15	PLC version and UI	S
%SW30	Time of last PLC scan cycle	S
%SW31	Maximum time of PLC scan cycle	S
%SW32	Minimum time of PLC scan cycle	S
%SW50 **%SW51** **%SW52** **%SW53**	Schedule block (RTC) function : words containing current date and time values (in BCD) %SW50 = seconds and day of the week %SW51 = hour and minute %SW52 = month and day %SW53 = century and year	S and U
%SW54 **%SW55** **%SW56** **%SW57**	Schedule block (RTC) function : words containing the data and time of the last power failure or PLC stop (in BCD) %SW54 = seconds and fault code %SW55 = hour and minute %SW56 = month and day %SW57 = century and year	S
%SW58	Identification code of last stop	S
%SW59	Adjust current date	U
%SW67	Value of Modbus end of frame character in ASCII mode	U
%SW68	End of frame character value (reception) in ASCII mode (TER port)	U
%SW69	EXCH block error code	S
%SW70	Function and type of TSX 07 PLC	S
%SW71	Devices on extension link	S
%SW76	Timer 1 ms	S
%SW77	Timer 1 ms	S
%SW78	Timer 1 ms	S
%SW79	Timer 1 ms	S
%SW100	Analog module input function command word	U
%SW101	Value of acquired analog module input	S
%SW102	Analog module output function command word	U
%SW103	Value of analog module output to be generated	U
%SW110	Read counter value	S
%SW111	Fast counting functions	S and U
%SW112	Value of potentiometer 0	S

Word	Function	Control
%SW113	Value of potentiometer 1	S
%SW114	Enable schedule blocks (RTC)	U
%SW118	Base PLC status word	S
%SW119	Peer PLC status word	S

S = controlled by the system,
U = controlled by the user.

6.2-2 Detailed description of system words

TSX Nano PLCs use the following system words :

System words	Function	Description
%SW0	PLC scan period	Modifies PLC scan period defined at configuration via the user program or the terminal (in the Data editor).
%SW11	Watchdog time	Reads watchdog time (150ms).
%SW14	Unitelway time-out	Is used to modify the value of the UNITELWAY time-out, via the user program (see part F Section 1.6)
%SW15	PLC version and UI	From versions V3.1 upwards, is used to show the version of the PLC (most significant byte) and its UI (least significant byte). Eg : 0x000 : version before V3.1 0x3119 : PLC is version V3.1 and UI : 25
%SW30	Last scan time (1)	Shows execution time of the last PLC scan cycle (in ms).
%SW31	Max scan time (1)	Shows execution time of the longest PLC scan cycle since the last cold start (in ms).
%SW32	Min scan time (1)	Shows execution time of shortest PLC scan cycle since the last cold start (in ms).
%SW50 **%SW51** **%SW52** **%SW53**	Schedule block function	System words containing current values of the date and time (in BCD) : %SW50 : SSXN Seconds and day of the week with (N= 0 for Monday to 6 for Sunday) %SW51 : HHMM Hour and Minute %SW52 : MMDD Month and Day %SW53 : CCYY Century and Year. These words are controlled by the system when bit %S50 is at 0. These words can be written by the user program or by the terminal when bit %S50 is set to 1.
%SW54 **%SW55** **%SW56** **%SW57**	Schedule block function	System words containing the date and time of the last power failure or PLC stop (in BCD) : %SW54 : Seconds and day of the week %SW55 : Hour and minute %SW56 : Month and day %SW57 : Century and year.

(1) This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle.

<table>
<tr><th>System words</th><th>Function</th><th>Description</th></tr>
<tr><td>%SW58</td><td>Code of last stop</td><td>Displays code giving cause of last stop :
1= Terminal switch changed from RUN to STOP
2= Stop at software fault (PLC scan overshoot)
4= Power outage
5= Stop at hardware fault</td></tr>
<tr><td>%SW59</td><td>Adjust current date</td><td>Contains two sets of 8 bits to adjust current date.
The operation is always performed on rising edge of the bit. This word is enabled by bit %S59.
<table>
<tr><th>Increment</th><th>Decrement</th><th>Parameter</th></tr>
<tr><td>bit 0</td><td>bit 8</td><td>Day of the week</td></tr>
<tr><td>bit 1</td><td>bit 9</td><td>Seconds</td></tr>
<tr><td>bit 2</td><td>bit 10</td><td>Minutes</td></tr>
<tr><td>bit 3</td><td>bit 11</td><td>Hours</td></tr>
<tr><td>bit 4</td><td>bit 12</td><td>Days</td></tr>
<tr><td>bit 5</td><td>bit 13</td><td>Months</td></tr>
<tr><td>bit 6</td><td>bit 14</td><td>Years</td></tr>
<tr><td>bit 7</td><td>bit 15</td><td>Centuries</td></tr>
</table></td></tr>
<tr><td>%SW67</td><td>End of Modbus frame</td><td>Is used to set 'LF' at the end of the Modbus frame in ASCII mode. This word is written at 16#000A by the system on a cold start. The user can modify this word using the program or Adjust mode when the Master uses an end of frame character other than 16#000A.</td></tr>
<tr><td>%SW68</td><td>End of Reception frame ASCII mode</td><td>Is used to set the parameters for the value of the end of frame byte in ASCII mode. Reception ends as soon as this is received.
The default value is 16#000D.</td></tr>
<tr><td>%SW69</td><td>EXCH block error code</td><td>If an error occurs when using the EXCH block, output bits %MSG.D and %MSG.E change to 1. This system word contains the error code. The possible values are as follows :
0 : No error, exchange correct
1 : Transmission buffer too large
2 : Transmission buffer too small
3 : Table too small
4 : Incorrect Unitelway address (Unitelway mode only)
5 : Time - out elapsed (Unitelway mode only)
6 : Transmission error (Unitelway mode only)
7 : Bad ASCII command (ASCII mode only)
8 : Not used
9 : Reception error (ASCII mode only)
10 : Table %KWi prohibited.

This word is set to 0 every time the EXCH block is used.</td></tr>
</table>

System words	Function	Description
%SW70	Address and type of PLC	Contains the following information : • bit 0 : 0=TSX 07 3L ••28 model 1= other models • bit 2 : 1=presence of schedule block (RTC) • bit 4 bit 3 PLC type TSX 07 0 0 TSX Nano with 6 inputs/ 4 outputs (10 I/O) 0 1 TSX Nano with 9 inputs/ 7 outputs (16 I/O) 1 0 TSX Nano with 14 inputs/ 10 outputs (24 I/O) 1 1 TSX Nano with AC inputs (16 I/O) • bits 7, 6 and 5 : address of PLC (copied from address code selector). If there is an I/O extension : • bits 12 and 11 : I/O extension type (same coding as bits 3 and 4) • bits 13 : 1 = there is an I/O extension The unused bits are at 0.
%SW71	Devices on extension link	Shows the communication status of each extension with the base PLC : bit 1 : I/O extension bit 2 : peer PLC or analog module #2 bit 3 : peer PLC or analog module #3 bit 4 : peer PLC or analog module #4 Bit at 0 if there is no extension or peer, no power, or a fault. Bit at 1 if there is an extension and exchange with the base PLC.
%SW76 to %SW78	Downcounting Words 1 ms	These 4 words serve as 1 ms timers. They are decremented individually by the system every millisecond if they have a positive value. This gives 4 downcounters downcounting in milliseconds, which is equal to an operating range of 1 ms to 32767 ms. Setting bit 15 to 1 can be used to stop decrementation.
%SW100	Analog input	Analog input functions command word. Value : 0 Analog input not valid Value : 1 Operation without scaling Value : 2 Unipolar scale range (period 125ms) Value : 3 Bipolar scale range (period 125ms) Value : 4 Unipolar scale range (period 500ms) Value : 5 Bipolar scale range (period 500ms) Writing this word is the responsibility of the application program.

System words	Function	Description
%SW101	Analog input	Word containing the value of the acquired analog input. The value range depends on the selection of operation made in %SW100. %SW100=0 %SW101=0 %SW100=1 %SW101 varies from 0 to 1000 %SW100=2 or 4 %SW101 varies from 0 to 1000 %SW100=3 or 5 %SW101 varies from -10000 to 10000
%SW102	Analog output	Analog output functions command word. Value : 0 Normal %PWM operation Value : 1 Operation without scaling Value : 2 Unipolar scale range Value : 3 Bipolar scale range analog %PWM Writing this word is the responsibility of the application program
%SW103	Analog output	Word containing the value to be applied to the analog output. The value range depends on the selection of operation made in %SW102. %SW102=0 %SW103=0 %SW102=1 %SW103 between 5 and 249 %SW102=2 %SW103 between 0 and 10000 %SW102=3 %SW103 between -10000 and 10000 Writing this word is the responsibility of the application program
%SW110	Up/down counter	Read value of the counter on a rising edge at input %I0.4
%SW111	Fast counter	Bit 0 : counting direction (1= upcounting, 0= downcounting) Bit 1 : 1= enable HSC (direct) outputs Bit 2 : 1= select frequency meter time base (1=100ms, 0=1s) Bit 3 : 1= update %FC frequency (also indicates the validity of the acquired value on the analog input module) This bit is reset to 0 by the user.
%SW112	Value of potentiometer 0	Contains the conversion on 8 bits (0 to 255) of the position of potentiometer number 0.
%SW113	Value of potentiometer 1	Contains the conversion on 8 bits (0 to 255) of the position of potentiometer number 1.
%SW114	Schedule block enable	Enables or disables operation of schedule blocks (RTC) by the user program or the terminal. bit 0 : 1 = enables schedule block #0 ..... bit 15 : 1 = enables schedule block #15 Initially all schedule blocks are enabled.

System words	Function	Description
%SW118	Base PLC status	Shows faults detected on base PLC. bit 0 : 0= tripping of one of the outputs (1) bit 3 : 0= sensor supply fault bit 8 : 0= TSX Nano internal fault or hardware fault bit 9 : 0= external fault or comm. fault bit 11 : 0= PLC performing self-tests bit 13 : 0= configuration fault (I/O extension configured but absent or faulty) All the other bits of this word are at 1 and are reserved. Thus, for a PLC which has no fault, the value of this word is : 16#FFFF.
%SW119	I/O peer PLC status	Shows faults detected on I/O peer PLC (this word is only used by the base PLC). The bit assignment of this word is identical to that of word %SW118, except for : • bit 13 : not significant • bit 14 : peer PLC now missing although it was present at initialization (2)

(1) after a shortcircuit or overload of the protected transistor outputs.

B

7.1 Operating modes

PL7 language is used to take into account the three main operating mode groups (1) :
- Checking.
- Running or production.
- Stopping.

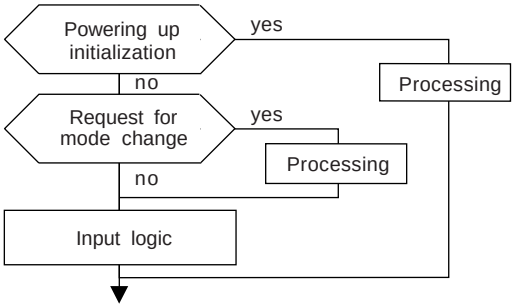
These different operating modes can be obtained around or starting from Grafcet using the following methods :
- Grafcet initialization.
- Prepositioning of steps.
- Maintaining a situation.
- Freezing charts.

Preliminary processing and use of system bits ensure effective operating mode management without complicating and overburdening the user program.

Structuring preliminary processing

The diagram below shows the structure which should be assigned to preliminary processing so that each processing operation is performed in order of importance for the following :

- Powering up
- Changing operating mode
- Input logic



Grafcet system bits

Use of bits %S21, %S22 and %S23 is reserved for preliminary processing only. These bits are automatically reset by the system. They must be written by operation code S only.

Initializing Grafcet, %S21

Causes :
- on cold restart
- setting %S21 to 1 via the program or terminal.

Consequences : deactivation of all active steps and activation of all initial steps.

(1) These operating modes are defined in the "Design Guide for Operating and Stoppping Modes" produced by the Applied Industrial Automation Development Agency.

Resetting Grafcet, %S22

Causes : %S22 set to 1 via the program or terminal.

Consequences :
- deactivation of all active steps
- scanning of sequential processing stopped.

Prepositioning Grafcet, %S22 and %S23

Procedure :
- reset Grafcet by setting %S22 to 1
- preposition the steps to be activated by a series of S Xi instructions
- enable prepositioning by setting %S23 to 1.

Freezing a situation :
- in initial situation : by maintaining %S21 at 1 by program
- in "empty" situation : by maintaining %S22 at 1 by program
- in situation determined by maintaining %S23 at 1.

7.2 Programming advice

Handling program jumps
Use program jumps with caution to avoid loops which are too long and can increase scan time. Avoid program jumps towards instructions which are located upstream.

Programming of outputs
An output bit or internal bit can only be controlled once in the program. In the case of output bits, only the last value scanned is taken into account when the outputs are updated.

Using directly-wired emergency stop sensors
Sensors used directly for emergency stops must not be processed by the PLC. They must be connected directly to the corresponding outputs.

Handling power returns

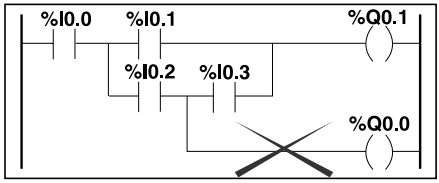
Make power returns conditional on a manual operation, as an automatic restart of the installation could cause unexpected operation of equipment (use system bits %S0, %S1 and %S9).

Time and schedule block management
The state of bit %S51, which indicates any schedule block faults, should be checked.

Note : When the program is entered, the terminal checks the syntax of the instructions, the operands and their association. The terminal diagnostics function checks programming errors (see Appendices, Part G).

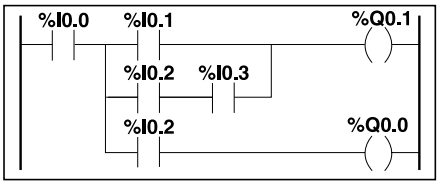
Additional notes on using parentheses

- Assignment operations should not be placed within parentheses.




```
LD    %I0.0
AND   %I0.1
OR(   %I0.2
ST    %Q0.0
AND   %I0.3
)
ST    %Q0.1
```

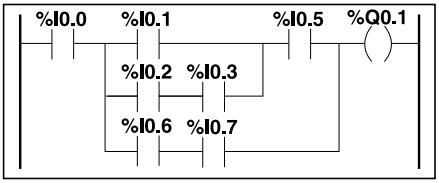
In order to perform the same function, the following equations must be programmed :



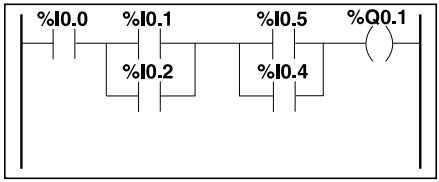

```
LD    %I0.0
MPS
AND(  %I0.1
OR(   %I0.2
AND   %I0.3
)
)
ST    %Q0.1
MPP
AND   %I0.2
ST    %Q0.0
```

- If several contacts are placed in parallel, they should either be nested within one another or totally dissociated from each other.

Example 1

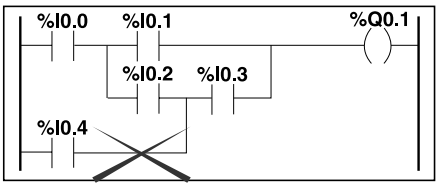


Example 2

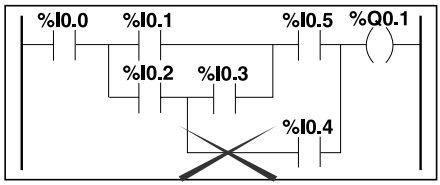


The following schematics cannot be programmed.

Example 3

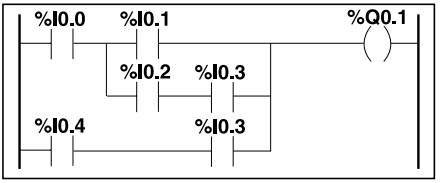


Example 4



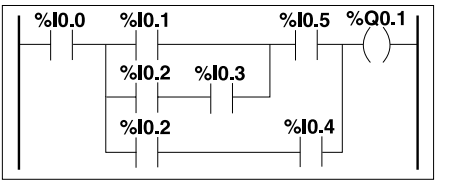
In order to execute schematics equivalent to those on the preceding page, they must be modified as follows :

Example 5 (see example 3)




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
OR(     %I0.4
AND     %I0.3
)
ST      %Q0.1
```

Example 6 (see example 4)




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
AND     %I0.5
OR(     %I0.2
AND     %I0.4
)
)
ST      %Q0.1
```

7.3 Reactivating protected transistor outputs on TSX 07•• ••12

When a channel fault has caused all the PLC outputs to trip (base PLC or I/O extension), they need to be reactivated. The reset may be :

- requested by an operator command. This is the recommended type of reactivation (see Part A Section 1.7-2)
- automatic. When using this procedure, it is essential to know in advance how automatic reactivation of the outputs will effect the process or machine being controlled.

This selection is made by programming system bit %S49.

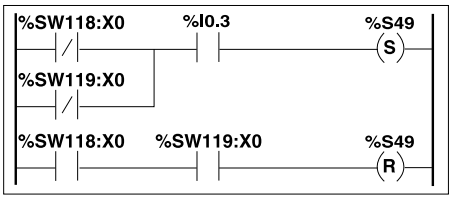
Output tripping

The appearance of an overload or short-circuit on an output causes :

- current limitation of the affected output
- tripping of all the outputs in the block (base PLC or I/O extension)
- activation with a steady light of the I/O lamp on the base PLC (if base PLC outputs tripped) or lamps on the base PLC and I/O extension (if I/O extension outputs tripped)
- setting to 0 of I/O fault system bit %S10
- setting to 1 of the system bit %S118 (tripping of base PLC outputs) or system bit %S119 (tripping of I/O extension outputs),
- setting to 0 of system word bit %SW118:X0 (tripping of base PLC outputs) or system word bit %SW119:X0 (tripping of I/O extension outputs).

Manual reactivation of outputs (dependent on an operator command)
System bit %S49 is set to 1 by an external action. System bit %S49 must be reset to 0 after the outputs have been reset :

```
LDN   %SW118:X0
ORN   %SW119:X0
AND   %I0.3 (input I0.3 for example)
S     %S49
LD    %SW118:X0
AND   %SW119:X0
R     %S49
```

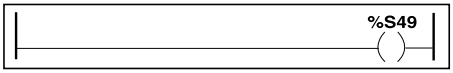


The minimum time between two reactivate commands allowed by the system is 10s. If the fault which caused tripping has disappeared :

- the outputs are once again active in the state defined by the program
- the I/O lamps are extinguished
- the system bits and system word bits are set to their default state *(normal state)* : %S10, %SW118:X0, %SW119:X0 at state 1, %S118 and %S119 at state 0 (see Part B Section 6.2).

Automatic reactivation of outputs
System bit %S49 is permanently set to 1 by the following program :

```
LD    1
ST    %S49
```

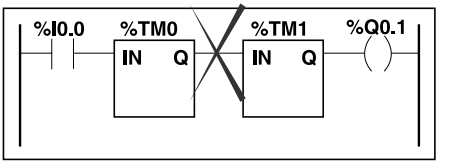


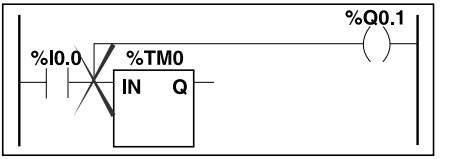
Reactivation is automatically requested every 10 seconds. When reactivation occurs, the behavior of the outputs, the indicator lamps and system words and bits is identical to that described for manual reactivation.

7.4 Reversibility conditions

The following conditions must be checked so that a program can be fully reversible (1) :

- The following instructions should not be used for reversibility : XOR, XORN, XORF, XORR, JMPCN, ENDCN, or N.
- Function blocks must be programmed in such a way as to be reversible (see Section 2.2-2).
- Function blocks cannot be cascaded or nested.
- Assignment instructions are prohibited between BLK and OUT_BLK instructions, and BLK and END_BLK instructions (if OUT_BLK is not programmed).





Prohibited programming

```
BLK  %TM0
LD   %I0.0
ST   %Q0.1
IN
END_BLK
```

Permitted programming

```
BLK  %TM0
LD   %I0.0
IN
END_BLK
LD   %I0.0
ST   %Q0.1
```

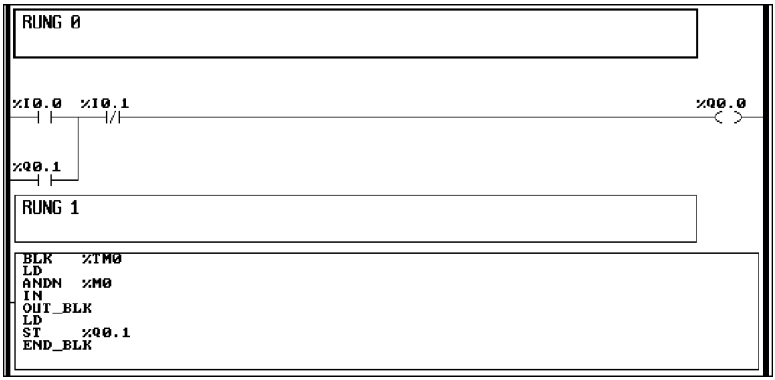
(1) **Note :** When a sequence of instructions is not reversible, it remains in List Language, while the remainder of the reversible program is converted into Ladder diagrams.

7.5 Reversibility rules

- A canonical rung cannot exceed a height of 7 cells, and a width of 11 cells (7x11 grid).
- A sentence beginning with LD must end with a conditional action instruction.
- JMPCN, ENDCN, NOP, and N instructions are not reversible.
- Action instructions within parentheses are not reversible.
- Stack instructions within parentheses are not reversible.
- An OR instruction after an action instruction is not reversible.
- RET, JMP, and END instructions are unconditional. No other instructions can be in the rung.
- Function block inputs and outputs can only be accessed through reversible standard function block instructions. Direct access instructions for function blocks are not reversible.

- Instructions after an END_BLK in a sequence cause the sequence to be non-reversible.
- Function block outputs used with AND and OR instructions are not reversible.
- An OR instruction inside a function block output rung must be inside parentheses.
- An unconditional action instruction between a BLK and an END_BLK is not reversible.
- An OUT_BLK must be followed by a LD of a valid function block output or an END_BLK.
- An instruction preceding an MRD or MPP instruction must be either a conditional action instruction or a stack operation.
- A non-nested OR instruction between an MPS and MPP pair is not reversible.
- An OR instruction after an MPS, MRD, or MPP instruction is not reversible.
- An MCS instruction cannot be used in the same rung with any other action instruction.
- A subroutine call or JMPC instruction must be the last action instruction of a rung.
- When putting a title and comments before a List instruction, there can only be 1 title line and up to 4 comment lines. If a blank line is placed anywhere between the title and the List instruction, part of the header may not show up in the corresponding Ladder rung header.
- If more than 4 lines of comments are placed ahead of the List instruction, the 5th comment line will be interpreted as a title line and the comment lines above will not appear in the corresponding Ladder rung header.

When a sequence of instructions is not reversible, it remains in List Language, while the remainder of the reversible program is converted into Ladder diagrams as shown below.



B

C

C

C

C

C

C

C

C

C

C

1.1 Introduction

Part C describes how to use the PL7-07 PC programming software to develop and manage applications for TSX Nano PLCs.

1.2 Features of the PL7-07 PC programming software

The PL7-07 PC programming software is a graphical development environment for writing and maintaining applications for the TSX Nano PLCs.

The PL7-07 PC programming software is a Windows-like development tool that is used to develop applications on IBM-AT compatible PCs with MS-DOS 3.3 or later. Among the PL7-07 PC programming software features are :

- multi-windows capability (up to 3)
- easy use with keyboard or mouse
- reversible Ladder and List programming
- two-step, point and click Ladder programming
- offline and online programming
- program and/or data animation
- easy configuration by simply selecting desired functions in dialog boxes,
- cut, copy, and paste program editing
- symbolic programming
- independent data files
- import/export features
- cross referencing
- print-out
  - program in List and/or Ladder
  - configuration

1.3 Other information

Part G contains appendices that are referenced in part C. The following is a list of the appendices :

A.1 Boolean List and control instructions with Ladder equivalent

A.2 Tool bar and instruction bar menu options

C

1.4 Conventions

The following typographic conventions are used in part C :

Format	Represents
bold	A word or phrase shown in **bold** text indicates that you should enter the word or phrase exactly as it appears. The names of menu options and dialog box fields are also shown in **bold** type.
italics	A word in *italics* indicates a place holder for information that you must provide. A word in *italics* also indicates a new term, which is then followed by a definition.
<carats>	A word in <carats> indicates the name of a key on the keyboard, such as <delete>.

2.1 System requirements

2.1-1 Machine compatibility

The following class and/or types of computers can run the PL7-07 PC programming software :

- IBM PC-AT compatible machines
- Programming terminals
  - FTX 417 20 / 417 40 (1) / FT 2000 (1)
  - FTX 507 or FTX 517 (1)

(1) recommended terminals for optimal performance

2.1-2 System requirements

The computer system should meet these minimum requirements :

- 20MHz 286 microprocessor
- 640 KB of RAM and 2 megabytes of available extended memory
- 3 megabytes of available hard disk space
- EGA, VGA, or SVGA monitor
- One available COM serial port - COM1 through COM4
- One parallel port for printing - LPT1 through LPT4
- MS-DOS version 3.3 or higher

Typical configuration :

- 386 microprocessor or higher

Although the PL7-07 PC programming software will run on a computer with the minimum requirements above, you may experience slow performance if you have multiple windows open simultaneously, a large amount of animated data, or a very large application.

2.2 Connections

The connections specific to the PC or FTX 417/FTX 507 terminals (display, keyboard, mouse, and printer) are described in their documentation. This section describes the following :

- connection of power supply
- connection of communications cable.

WARNING

A communication cable should never be connected to the PLC port before being connected to the peripheral device (programming terminal, operating terminal etc).

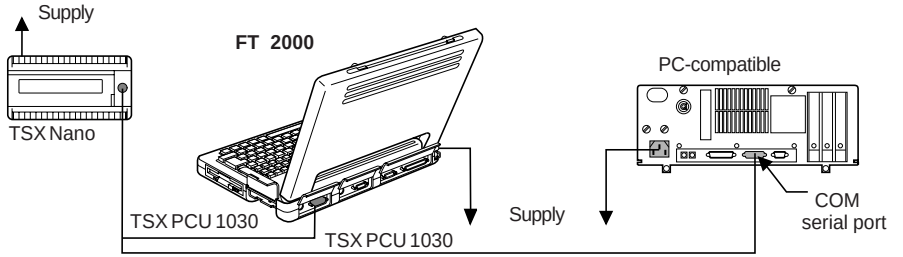
C

2.2-1 Connection of power supply

See part A, section 3.2 for connection of the power supply.

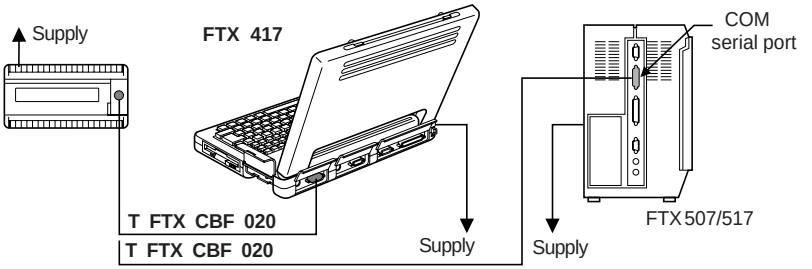
2.2-2 Connection of communication cable from PC to PLC

This connection requires communication cable TSX PCU 1030 (2.5 meters long), supplied with TLX L PL7 07P 30E software. For a 25-pin serial connector, use the TSX CTC 07 adaptor with communication cable TSX PCU 1030.



2.2-3 Connection of communication cable from FTX 417 / FTX 507/517 / FT 2000 terminals to PLC

This connection requires communication cable T FTX CBF 020 (2 meters long), supplied with software TLX L PL7 07F 30E.



2.3 Installing the PL7-07 PC programming software

The PL7-07 PC programming software is contained on two, high-density 3.5" disks. To install the PL7-07 PC programming software :

1. With the DOS prompt at **c:**, insert **installation Disk #1** in the 3.5" floppy disk drive.
2. Select the disk drive by typing **a:** or **b:** at the DOS prompt and press **<enter>**.
3. Type **install** and press **<enter>** to launch the PL7-07 PC programming software.
4. Select the language for installing the program. The PL7-07 PC programming software can be installed in one of the following five standard languages : English, French, German, Italian, and Spanish. Other languages can be installed using a 3.5" disk (supplied separately) containing the files translated into the required language.

Note :
Contact your nearest SCHNEIDER distributor to obtain information on how to install PL7-07 in a language other than E/F/G/I/S.

The PL7-07 PC programming software can operate in only one language at a time. If support for multiple languages is required, create a new directory with a different name and reinstall the software in the new directory. If you reinstall the software in the same directory as the original installation, the newly-installed version will overwrite the old one.

5. When prompted, select a target drive for installing the PL7-07 software.
6. When prompted, select a name for the PL7-07 PC programming software directory. The default subdirectory is <drive>:\PL707.
7. If the operating system used is DOS or Windows 3.1/95/98, when prompted select a communication port for the PLC.
   If the operating system used is Windows NT, the communication port for the PLC will be selected using the XWAY drivers manager (see Section 2.4).
8. The program decompresses the installation files and writes the PL7-07 PC programming software files to the selected drive.
9. When prompted, insert **installation Disk #2** in the floppy drive.
10. Respond **Yes (or No, see note)** to the prompt : "May I create/modify your CONFIG.SYS file if needed (Y/N)?
11. Respond **Yes (or No, see note)** to the prompt : "May I create/modify your AUTOEXEC.BAT file if needed (Y/N)?
12. When prompted, insert **installation Disk #1** in the floppy disk drive.

13. When prompted, press any key to terminate the installation program and reboot the computer.

Note : If you replied **No** in steps 10 and 11, write down the data displayed at the end of the installation program to manually update your configuration files. Before terminating the installation program, modify your configuration files as follows :

1. Display your AUTOEXEC.BAT file in a text editor. Insert the statement, C:\PL707, to the path statement :

   PATH=C:\WINDOWS;C:\DOS;C:\PL707

   where C: is the drive selected and \PL707 is the name selected for the PL7-07 PC programming software directory.

2. Display your CONFIG.SYS file in a text editor. Insert these statements ***:***

   FILES=30 or more

   DEVICE=C:\PL707\DUNTLW.EXE PROFILE=C:\PL707\DUNTLW.001

   COM1 is the default communications port. To change the COM port setting, edit DUNTLW.001 in directory C:\PL707. In the Basic Parameters section, change the COM setting in the line PORT=COM1:O,8,1 to the desired COM port.

3. Reboot the computer.

Note :
The PL7-07 installation software, INSTALL.EXE, requires at least 370 Kb of available memory in the 640 Kb of RAM memory to run correctly.

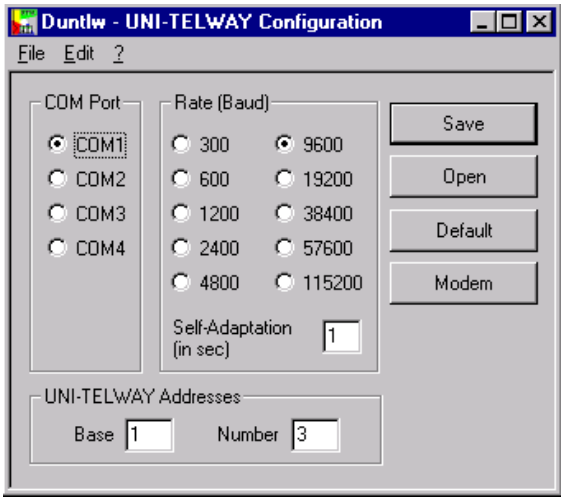
2.4 Installing the UNI-TELWAY driver in Windows NT

The NT version of the UNI-TELWAY communication driver, supplied on a 3.5" high-density floppy disk, has its own installation procedure which is different from that of PL7.07 V3.0.
The UNI-TELWAY driver can be installed before or after installation of PL7.07 software.

To install the UNI-TELWAY driver :

1. Insert the UNI-TELWAY driver floppy disk in the disk drive.
2. In the **Start/Run** menu**,** select **A:\Setup.exe** and click **OK**.
3. Confirm the welcome screen by clicking **Next**.
4. Select a target drive for installation of the UNI-TELWAY software.
5. Define the program group (the software suggests the **Modicon Telemecanique** group by default).
6. Select the language.
7. Configure the communication port by selecting :
   - the communication port for the PLC
   - the transmission speed
   - the UNI-TELWAY address

   then save the configuration.



8. Close the dialog box by clicking on  ,
9. Reboot the computer.

The configuration parameters of the UNI-TELWAY driver can be displayed and/or modified using the XWAY drivers manager located in the program group defined during installation (Modicon Telemecanique by default).

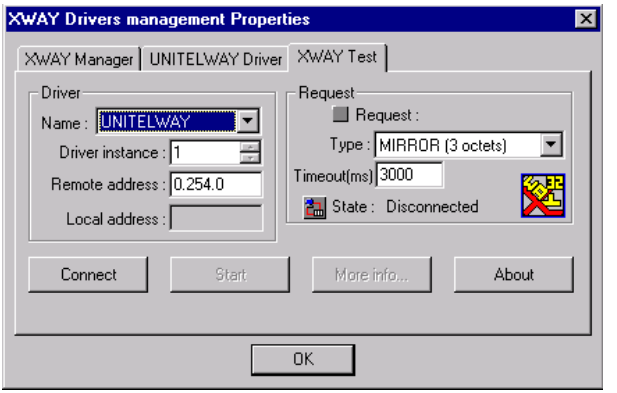
This tool is made up of XWAY Manager, UNI-TELWAY Driver and XWAY Test tabs.

The XWAY Manager tab :

- provides information on the driver(s) installed
- can be used to install, update or uninstall a driver.

The Unitelway Driver tab is used to modify the configuration parameters of the UNI-TELWAY driver.

The XWAY Test tab is used to test the link of the driver selected :



Driver group

- **Name** : Name of the driver to be used for the UNI-TELWAY test.
- **Driver instance** : Instance number of the driver to be used for the test (usually 1).
- **Remote address** : XWAY address of the remote station in “network.station.port” format. The address “0.254.0” is the default address (terminal port for example).

Request group

- **Type** : Type of request. Different sizes of mirror requests are provided.
- **Time-out** : Time-out in ms of the response to the request sent.
- **State** : Connection status : “disconnected”, “connecting…” or “connected”.

(3) Control buttons

- **Connect** : Opens an internal communication channel on the driver selected.
- **Start** : Starts sending requests to the station defined in the “Remote address” field of the “Driver” group. This button is active in online mode only.
- **More info…** : Displays system information on the driver. This button is active in online mode only.
- **About** : Displays the version and copyright of the XWAY drivers manager.

2.5 File management

2.5-1 Types of files

Seven different types of file are used in the PL7-07 PC programming software. The following describes the file types and how they are used :

Application file (.PL7 extension)	This file type (.PL7 extension) is used for storing applications developed in the PL7-07 PC programming software. Application files are accessed from the File menu.
Binary file (.APP extension)	This file type (.APP extension) is used for storing applications developed in the PL7-07 PC programming software to a format that allows the file to be transferred to the FTX 117 terminal using a memory card or vice versa.
Data files (.DAT extension)	This file type (.DAT extension) is used for storing data pages developed in the Data Editor. Data files are accessed from the Tools menu when the Data Editor window is displayed. The .dat extension allows for these files to be transferred to the FTX 117 terminal via the memory card without saving them as a binary file type.
Symbol files (.SYM extension)	This file type (.SYM extension) is used when exporting symbols from PL7-07 application to another
ASCII text files (.TXT extension)	This file type (.TXT extension) is used when importing and exporting List program files.
Export files (.IL extension)	This file type (.IL extension) is used when exporting PL7-07 List source programs to PL7 Micro.
Export files (.LD extension)	This file type (.LD extension) is used when exporting PL7-07 Ladder source programs to PL7 Micro.

2.5-2 Backup files

When changes are made to an application file (.pl7 extension) and the file is saved, a backup file is automatically generated with a .bak extension. The .bak file is the latest version of the application file before the changes were made. The .bak file is stored in the same directory as the application file. If you desire to use a different backup extension, change the setting in the "BackupExtension=" line of the "AppDefaults" section of the PL707.INI file using the DOS editor. If you do not want to automatically generate a backup file, leave the space blank after "BackupExtension=".

If you want to use the .bak file, select **All (*.*)** from the List Files of Type field in the File Selection dialog box. All the files in the selected directory will be displayed. Select the .BAK file for the desired application.

2.5-3 Making directories for the application and data files

To assist in managing the application and data files created during the development of an application, it is recommended that separate directories be created for the application (.PL7) and data (.DAT) files.

Note : The following access paths assume the PC programming software has been installed in the default directory C:\PL707.

1. At the prompt **C:\PL707,** type MKDIR APPS to create a subdirectory for the application files. The full path designation is C:\PL707\APPS\filename.
2. At the prompt **C:\PL707,** type MKDIR DATA to create a subdirectory for the data files. The full path designation is C:\PL707\DATA\filename.

C

2.5-4 Import/Export information

The PL7-07 PC programming software allows you to import and export ASCII Program files and Symbols files. It also allows source programs (List or Ladder) to be exported to PL7-Micro. The Import and Export options are accessed from the File menu. The following describes the files that can be imported and exported :

1. List programs can be developed by external ASCII text editors. ASCII Program Import and Export allows you to import and export these files between the PL7-07 software and the ASCII text editor.
2. Symbols Import and Export allows you to copy symbols from one PL7-07 software application to another. This type of import/export uses an internal file format with the file extension .SYM. This file type is not usable by any external tools such as text editors and spreadsheets. Symbols Import and Export also allow you to create default Symbols files for common applications.
3. A current PL7-07 application source program can be exported to PL7-Micro in reversible List or Ladder. The extension used, IL (List) or LD (Ladder), determines in which programming language it is exported. The default extension is that of the current editor selected.
   When the LD extension is used, the export file contains the program, the rung headers (only the titles are exported in Ladder), the symbols and configuration data. Objects which cannot be interpreted by PL7-Micro (fast counter, Grafcet instructions, etc) are replaced by an empty line.
   When the IL extension is used, the export file contains the program, the comment lines, the symbols and the configuration data. Instructions comments are not exported.

2.5-5 Importing symbols and comments

The PL7-07 PC programming software enables you to import program comments and symbols from a file in another application. This option is used to import object symbols and comments from a standard program without having to re-enter them.
This option is a sub menu of the "Import" function which can be accessed from the "File" menu. An application must be open to use this option. When the option is launched, a screen appears, enabling the source application file to be selected.

The symbols loaded from the source application file are placed in the symbol table of the current application. The comments (placed at the end of the program line or a whole line) are merged with the current application.

Rules for importing symbols and comments :

- The current application should not include any symbols or comments for the import operation to be performed successfully.
- End of line comments are placed in a rung and line with the same number and row as the source application.
- If the rung line row does not exist in the current application, the end of line comment is inserted as a whole line comment at the end of this rung.
- If the rung number does not exist in the current application, the end of line comment is inserted as a whole line comment at the end of the program.
- Whole line comments are inserted as a new line in a rung and line with a number and row equivalent to the source application.
- If the rung line row does not exist in the current application, the whole line comment is inserted at the end of this rung.
- If the rung number does not exist in the current application, the whole line comment is inserted at the end of the program.

Example

- Application 1 containing symbols and comments :

```
(* FIRST LINE COMMENT AT START OF APPLICATION*)
(* SECOND LINE COMMENT AT START OF APPLICATION*)
LD      INPUT_1
ST      OUTPUT_1        (* PROGRAM COMMENT*)
```

where %I0.1=INPUT_1 and %Q0.1=OUTPUT_1

- Current application without symbols or comments :

```
LD      %I0.0
AND     %I0.1
ST      %Q0.1
```

- After importing symbols and comments from application 1, the current application becomes :

```
(* FIRST LINE COMMENT AT START OF APPLICATION*)
(* SECOND LINE COMMENT AT START OF APPLICATION*)
LD      %I0.0
AND     INPUT_1         (* PROGRAM COMMENT*)
ST      OUTPUT_1
```

The comments are inserted in the program and the objects are linked to the imported symbols.

2.6 Running the PL7-07 PC programming software

2.6-1 Starting the PL7-07 PC programming software

1. At the prompt **C:\PL707** (or other specified subdirectory), type **PL707** to run the PL7-07 programming software.
2. The first time the PL7-07 programming software is run, a PL7-07 Registration dialog box is displayed. After entering your name and company, select **Ok** to open the program in the initial state.

When the program is subsequently run, the PL7-07 Registration dialog box is displayed briefly and the program is automatically opened at the initial state.

C

2.6-2 Opening an application

To open a new or existing application, see sections 4.2 and 4.3.

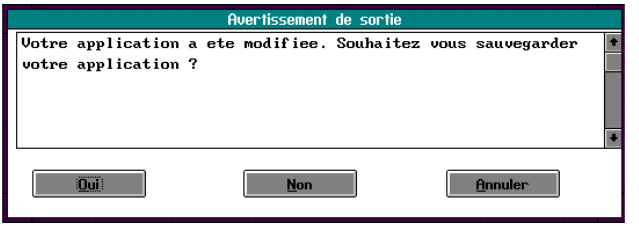
2.6-3 Closing an application

Only one application can be open at a time. If you try to open a second application with one already open, an Information dialog box is displayed with the following message : "Please close your application before opening another."

To close an application :

1. Select **Close** from the File menu.

   If the application file is not saved before it is closed, a Warning Exit dialog box appears.
2. Select **Yes** to save changes and exit the application file or **No** to close the file without saving the changes. Select **Cancel** to return to the application file.



2.6-4 Exiting the PL7-07 PC programming software

To exit the PL7-07 PC programming software :

1. Select **Exit** (CTRL Q) from the File menu.
2. If Exit is selected from the initial state, the program is immediately closed and the system returns to DOS.
3. If an application with unsaved changes is open, in either the offline or online state, a Warning Exit dialog box is displayed.
4. Select **Yes** to save changes and exit the application file or **No** to close the file without saving the changes. Select **Cancel** to return to the application file.

For information on executing PL7-07 in a Windows environment, see Appendix A.9.

C

3.1 Introduction

This section explains the concepts needed to develop and manage applications with the PL7-07 PC programming software.

3.1-1 Glossary of terms

Active window The window being currently used or currently selected. Only one window can be active at a time. Keystrokes and commands affect the active window. If a window is active, its title bar changes color to differentiate it from other windows.

Application file The PL7-07 application file is the main storage file for TSX Nano programs within the PL7-07 software. In the file is the source code, comments, and symbols for the program. Application files are distinguished by the .pl7 file extension.

Binary file A file containing information that is in machine-readable form—it can only be read by the TSX Nano PLC, the FTX 117 handheld programmer, and the PL7-07 software. The binary file contains the program that was compiled (validated) from an application file. It contains the PLC instructions in machine language form. No comments or symbols (program documentation) are contained in this file. Binary files are distinguished by the .app file extension.

Check box A small, square box that appears in a dialog box that can be selected or cleared. When the check box is selected, an X appears in the box. A check box represents an option that can be turned on or off.

Choose To use the mouse or keyboard to pick an item that begins an action in the PL7-07 software. Menu options are chosen to perform tasks.

Clear To turn off an option by removing the X from a check box. A check box is cleared by clicking it, or selecting it and pressing the <spacebar>.

Click To press and release the mouse button quickly.

Clipboard A temporary storage location used to transfer program segments between areas of the program and between applications. Program segments are transferred to the clipboard by using the Copy or Cut option from the Edit menu. Program segments are transferred from the clipboard by using the Paste option from the Edit menu.

Close To remove a window or dialog box , or to remove an application from memory.

Command button	In a dialog box, a button that carries out an action. A command button has a label in it to describe the action carried out (for example, Cancel, Help, or Ok).
Copy	To duplicate a selected program segment into the clipboard to be transferred to another location. The program segment selected is not removed from its original location.
Cursor	The place where text or programming instructions are inserted. The PL7-07 software uses a vertical, flashing cursor for most text insertions. The Ladder editor uses a square cursor the size of one cell of the programming grid.
Cursor keys	On the computer keyboard, the keys used to navigate around the screen. These include the <home>, <end>, <page up>, <page down>, up arrow, down arrow, right arrow, and left arrow keys.
Cut	To remove a selected program segment and place it in the clipboard.
Device	A component of the computer hardware configuration, such as a modem, printer, mouse, or PLC.
Device driver	The software that controls how a computer communicates with a device, such as a PLC. For example, the PL7-07 device driver translates information from the computer into information the PLC can understand.
Dialog box	A small window that appears temporarily to request information. Many dialog boxes have multiple options. An option must be chosen before the PL7-07 software can continue with an action or command.
Double-click	To rapidly press and release a mouse button twice without moving the mouse.
File	A collection of information that has been named and stored on a disk.
File name	The name of a file. The PL7-07 software uses the MS-DOS file naming conventions.
Grayed-out	A dimmed button, option, or menu that is displayed in light gray instead of black. The button, option, or menu cannot be chosen.
Inactive window	Any open window not currently being used.
List box	Within an application window or dialog box, a type of box that lists available choices—for example, a list of files in a directory. If all the choices do not fit in the list box, there is a scroll bar present to the right of the box.

Menu	A list of available options, summarized by the items in the menu bar at the top of the PL7-07 software main window. A menu is opened by selecting the menu name.
Menu bar	The horizontal bar near the top of the PL7-07 software main window containing the names of the PL7-07 software menus. The menu bar appears below the title bar.
Mouse pointer	The arrow-shaped cursor on the screen that follows the movement of the mouse. The mouse pointer indicates which area of the screen will be affected when you press the mouse button.
Option	A choice in a menu. For example, for File/Open, Open is an option in the File menu.
Paste	To copy the contents of the clipboard at the location of the cursor.
Point	To position the mouse pointer to the desired location on the screen.
Radio button	A small, round, option button that appears in a dialog box. Within a group of related radio buttons, only one can be selected.
Scroll	To move through lists and program displays (up and down) in order to view the entire contents of the list or program.
Scroll arrow	An arrow on either end of a scroll bar that is used to scroll through the contents of the window or list.
Scroll bar	A bar that appears at the right edge of a window or list box when the contents exceed the capacity of the display. Each scroll bar contains a scroll box and two scroll arrows.
Scroll box	In a scroll bar, the small box that shows the position of the information in the window or list box relative to the contents of the entire amount of information to be displayed.
Select	To highlight and pick the desired option from a menu, window, or dialog box. This can be done with either the mouse or keyboard.
Shortcut key	A key or key combination that is pressed to carry out a command or menu option. If an option has a shortcut key, the key combination is listed to the right of the option name on the menu.
Status bar	A bar that appears at the bottom of the PL7-07 software screen. It displays information and error messages, available memory in the PLC, whether the PLC is running or stopped, and the status of the application with respect to the PLC (initial, offline, online, or monitor).
System button	A button located at the left side of the title bar that is used to close a window by clicking on it with the mouse pointer.

Text box	A box located in a dialog box in which you type information needed to carry out an option or action. The text box may be blank or may contain text when the dialog box is opened.
Title bar	The horizontal bar (at the top of a window) that contains the title of the window or dialog box.
Window	A rectangular area on the screen in which you view programs, data, and documentation. A window can be opened, closed, and have its vertical size changed. Up to 3 windows can be opened at a time. For EGA monitors, only 2 windows can be opened at a time.

3.2 Using the PL7-07 PC programming software

You can use the PL7-07 PC programming software with either a mouse and keyboard, or keyboard only. This section describes how to perform some of the most common actions for both methods.

3.2-1 Selecting a menu option

1. To select a menu option with a **mouse**, position the cursor on the desired menu name and select it with the left mouse button. A menu appears. Position the cursor on the desired option and select it with the left mouse button.
2. To select a menu option using **keystrokes**, hold down the **<alt>** key and enter the letter that is underlined on the menu name. A menu appears. To select the option from the menu, enter the letter that is underlined on the menu option name.
3. To select a menu option using a **"quick key"**, hold down the combination of keys for the desired option. See the tables in appendix A.2 of part G for the options that have quick keys.

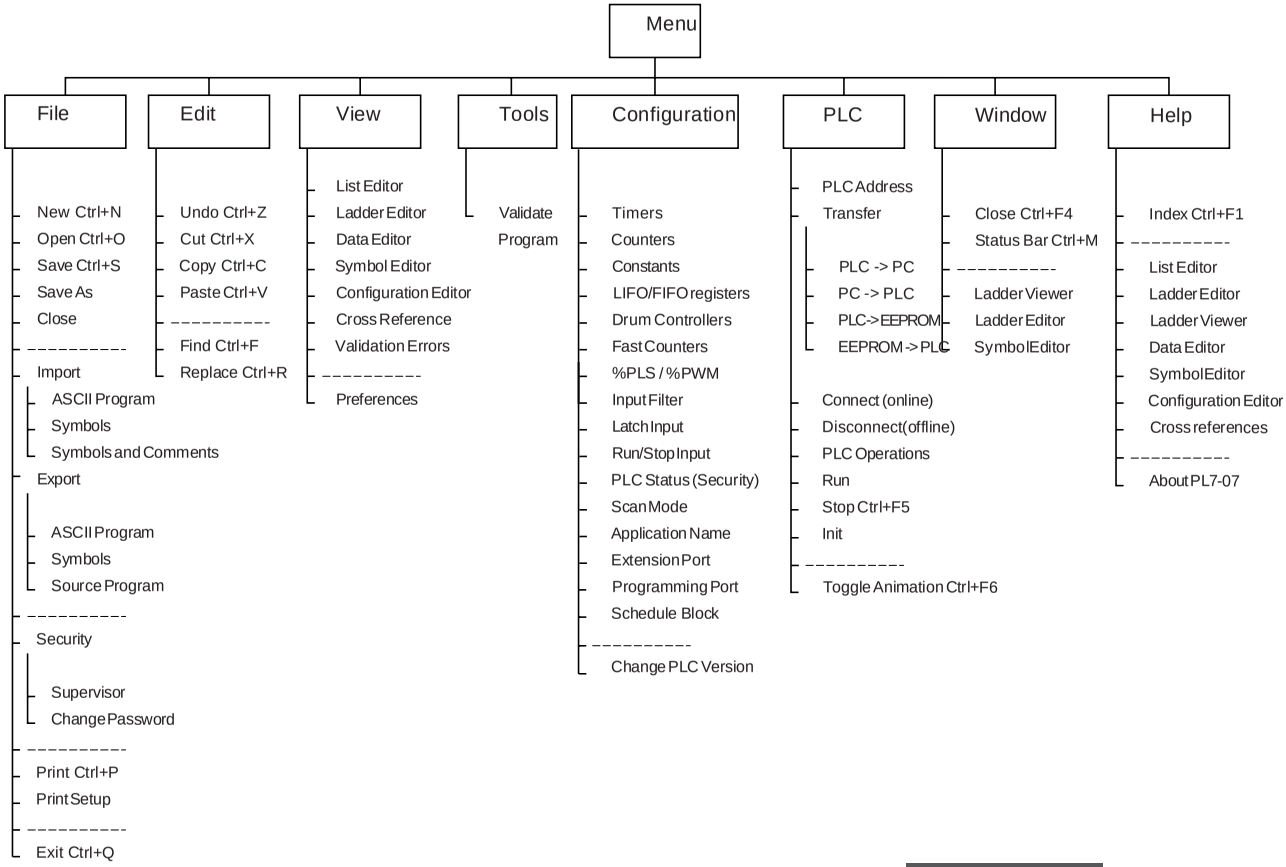
3.2-2 Selecting a toolbar item

To select a tool bar option, position the cursor on the desired "button" and select it with the left mouse button. See the tables in appendix A.2 of part G for the description of the tool bar buttons.

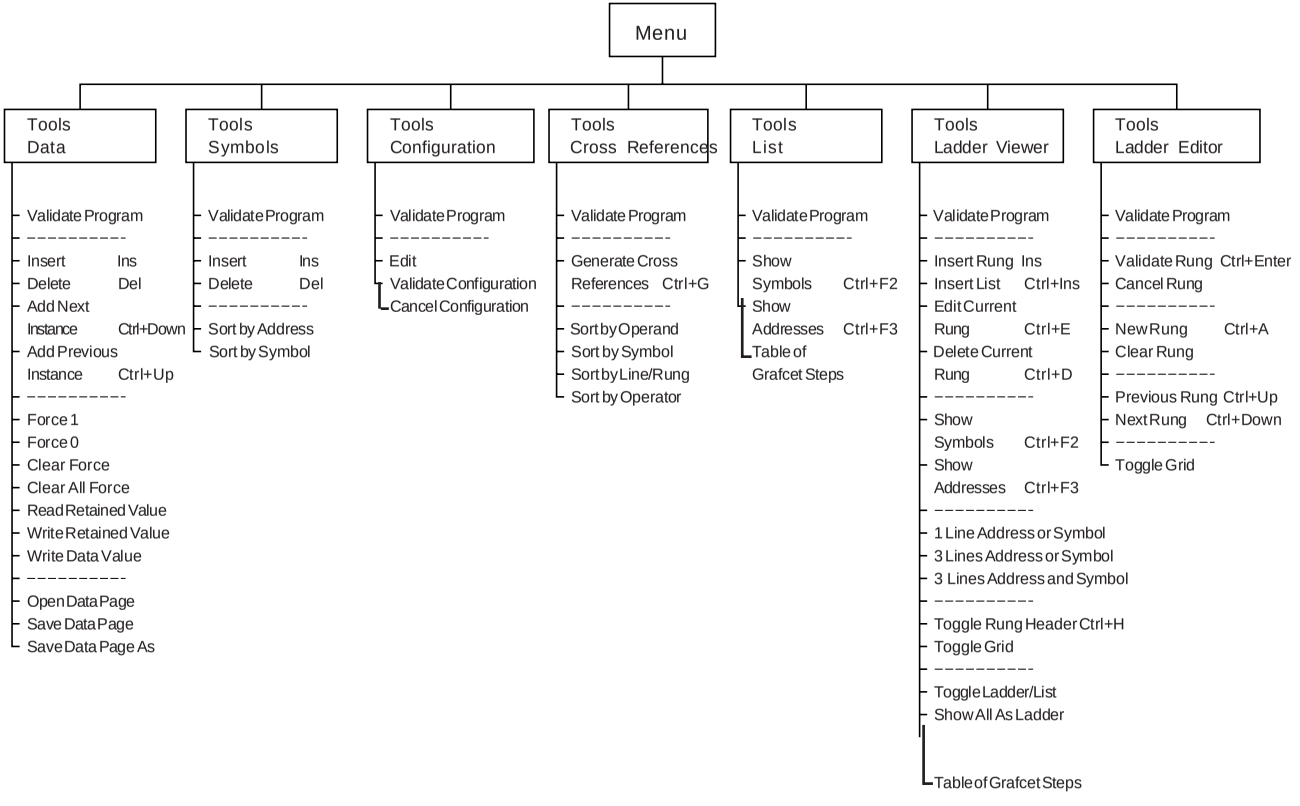
3.2-3 Menu charts

The following two illustrations show the PL7-07 software Main menu options and the Tools menu options. The Tools menu options vary depending on the window that is open.

MENU OPTIONS



MENU OPTIONS



3.3 Using the editors

The PL7-07 PC programming software is used to develop applications for the TSX Nano PLC. The term *application* includes a program, configuration data, symbols, and comments.

The PL7-07 software makes use of four editors in constructing an application—List/Ladder editor, Data editor, Configuration editor, and Symbol editor. For example, you write or edit a software program in the List/Ladder editor, construct and save data pages in the Data editor, enter or modify symbols in the Symbol editor, and enter or modify configuration data in the Configuration editor. Developing each application part in a separate editor makes the application development process more systematic, resulting in more clearly defined applications.

You can construct the application in any order. For example, you can choose to define configuration data first, then define data variables, and then write the program instructions—or complete these tasks in the reverse order. There is no prescribed order. The sections that follow describe each application part in detail.

3.3-1 Using the List and Ladder editors

The List and Ladder editors are used to develop the main program—the core of the application. The development of the main program includes:

- assigning addresses to the inputs and outputs being used for the program..
- selecting the pre-defined data variables to be used for the program.
- constructing the order in which the program is to be executed.

You can write a program for the PLC in either the List or Ladder format. Selecting one method over the other is a matter of preference and does not affect the application.

Ladder is a diagrammatic language that uses both graphic elements and text. Each segment of a Ladder diagram is called a rung. For more information on developing a Ladder program, see section 7.

List is a Boolean text-based instruction set. For more information on developing a List program, see section 8.

In either Ladder or List, you write the program in the logical order required to control a machine or process. You can automatically convert or *reverse* Ladder instructions to List instructions. You can also reverse List to Ladder instructions, if the structure of the program and the instructions comply with the reversibility rules. (See Section 10 of Part C.)

You can document your program using comments in both the List and Ladder Editors. Comments are notes that you put into your program instructions in free-format text. Comments are used primarily to document the meaning and purpose of program instructions.

3.3-2 Using the Data editor

The Data editor is used to construct and save data pages. A data page is used for listing all or part of the data variables used in the main program. The data page consists of variable addresses, current values, retained values, and associated symbols. This is useful for adjusting and debugging an application. For more information on using the Data editor, see section 14.

3.3-3 Using the Configuration editor

The Configuration editor is used to assign specific values to the hardware and software resources of the PLC—timers, counters, latched inputs, external run/stop switches, etc.—to control the behavior of these resources. The assigned values are called configuration data. For more information on configuring PLC resources, see section 5.

3.3-4 Using the Symbol editor

The Symbol editor is used to document a program by assigning tag words (symbols) to the data variables used in the program. Symbols are an optional part of an application. Symbols can be used in writing the program instructions, rather than the associated memory variable address. Symbols and their associations are defined and edited in the Symbol Editor. For more information on the Symbol editor, see section 6.

Note that only certain parts of an application reside in the PLC. For example, while you have an application online in the PLC, the symbols and comments remain in the PC memory, while the program and configuration data are maintained in the PLC memory. PL7-07 PC programming software maintains the relationship between the parts of the application.

3.4 PL7-07 PC programming software operating states

The PL7-07 PC programming software has four operating states:

- Initial
- Offline
- Online
- Monitor

The following describes the significance and attributes of each operating state.

3.4-1 Initial State

The PL7-07 software is in the initial state when the program is first started or when an application is closed. When in the initial state, the word “Initial” appears on the right side of the status bar at the bottom of the screen.

3.4-2 Offline State

When you open a new or existing application, the PL7-07 software changes from the initial to the offline state. An application in the offline state is not connected to the PLC and exists completely within the PC memory. In the offline state, you can create and edit program instructions and configuration data, and modify data variables, symbols, and comments. When you open an application in the offline state, the word “Offline” appears on the right side of the status bar at the bottom of the screen.

From the offline state, select **Connect** from the PLC menu to make an online connection. The flow chart shows the decisions made by the software when the Connect option is chosen.

3.4-3 Online State

An application in the online state is directly connected to the PLC memory. In the online state, you have unrestricted access to an application, unless the application is protected. Changes that you make to the application program, configuration data, and data variables are written directly to the PLC memory. Program documentation, such as symbols and comments, remain in the PC memory.

Online, you have complete access to all PL7-07 software features. When the PL7-07 software has an application in the online state, the word “Online” appears on the right side of the status bar at the bottom of the screen.

If you want to return to the offline state, select **Disconnect** from the PLC menu. You can also select the **Close** option from the File menu to return to the initial state, without passing through the offline state. If you select Close, you will be prompted to save your changes.

3.4-4 Monitor State

The monitor state provides access to the operating states and to the adjustment of the PLC. For instance, you may start, stop or reboot the PLC and view, modify, or transfer data in the Data Editor. However, the List/Ladder Editor, Symbol Editor, and Configuration Editor are not available in the monitor state. If the application in the PLC is protected (see section 12.4), the monitor state is the only online state available.

If you are in the offline state, and the application in the PLC is not protected—but is different than the application in the PC's offline memory you may choose to enter the monitor state. This allows you to monitor the PLC while maintaining the application in the PC's offline memory.

When the PL7-07 software is in the monitor state, the word “Monitor” appears on the right side of the status bar at the bottom of the screen. In the monitor state, select **Disconnect** from the PLC menu to change from the monitor state to the offline state.

3.5 Status bar

The status bar appears at the bottom of the PL7-07 software screen. The following shows the different parts of the status bar:

(1) Information and error messages. For a complete description of the message, click on this area of the status bar.

(2) Available memory in the PLC (in bytes).

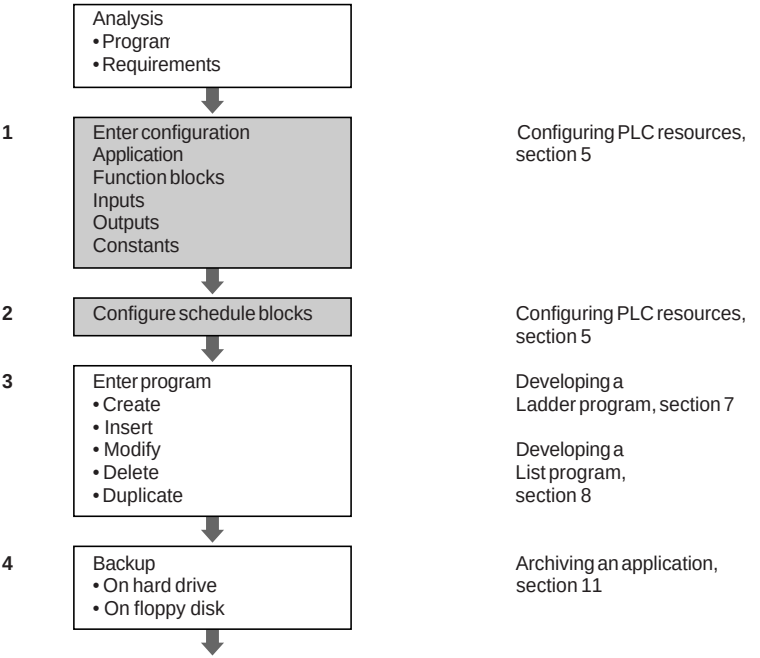
(3) Whether the PLC is running or stopped.

(4) The state of the PL7-07 software (initial, offline, online, or monitor).

3.6 Developing an application

3.6-1 Design phase

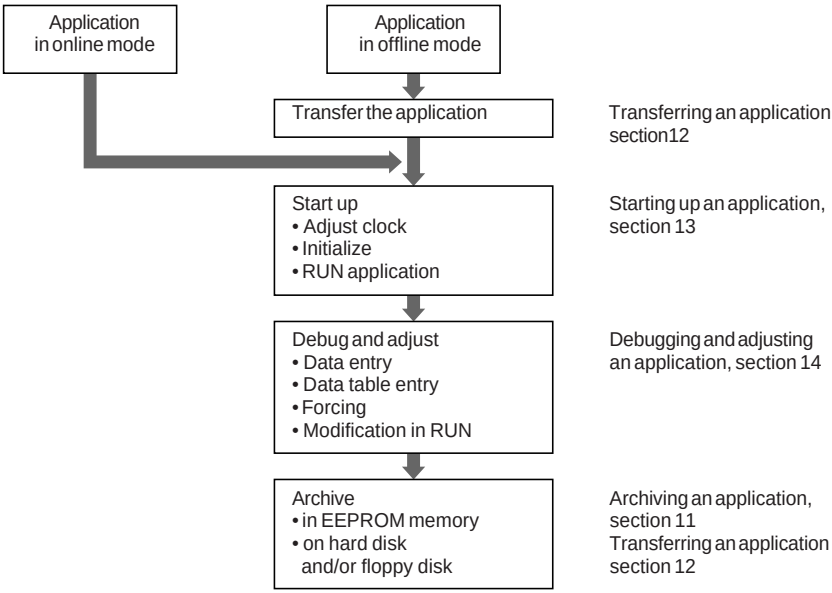
The following flowchart shows the recommended steps and associated sections for creating an application in List or Ladder using the PL7-07 PC programming software.



Optional, enabling the application being created to be customized. This operation can also be performed in parallel with operation 3.

3.6-2 Adjusting and debugging phase

The adjusting and debugging phase is done in the online or monitor state. The following flow chart shows the recommended steps and associated sections for adjusting and debugging an application.



C

C

4.1 Introduction

From the initial state, you can:

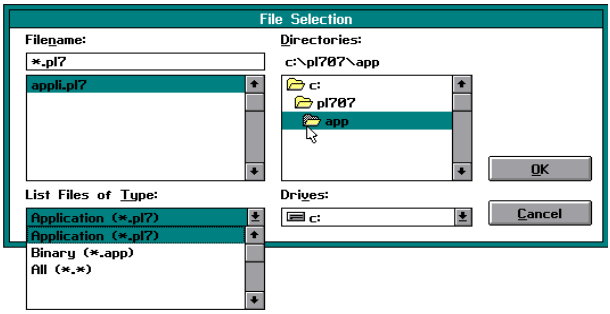
- create a new application file.
- open an existing application file.
- transfer an application to or from the PC, the PLC, or the EEPROM.
- connect the PC to the PLC.
- open a binary file.

4.2 Creating a new application file

Select **New** from the File menu to create a new application file. Selecting **New** changes the menu bar display. Menu bar items that were previously grayed out are active. In addition, the application state displayed in the status bar is changed from *initial* to *offline* state. PL707 V3.0 or later automatically opens the List editor or the Ladder viewer according to the choice made in the Preferences dialog box. The Ladder viewer is open by default.

4.3 Opening an existing application file

From the File menu, select **Open** to access an existing application file. A File Selection dialog box is displayed.



To select a file:

1. If the drive that stores the file you want to open is not the current drive, open the **Drives** field selection box by selecting the down arrow, and select a drive.

2. In the **Directories** field, select the directory containing the file that you want to open.
3. In the **List Files of Type** field, open the list box by selecting the down arrow. Select:

• Application (*.pl7) file type to display application files

• Binary (*.app) file type to display binary files

• All (*.*) to display all files in the directory.

4. In the **Filename** list box, select a file to display in the **Filename** field or type the name of the file you want to open.
5. Select **Ok** to open the file. When the file is opened, the application state changes from *initial* to *offline*. Select **Cancel** to close the File Selection dialog box without selecting a file.

4.4 Transferring an application

In the offline state, select **Transfer** from the PLC menu to copy an application between the PC and the PLC. You can also select **Transfer** to copy an application between the PLC and the EEPROM. The EEPROM is a permanent memory area in the PLC where an application can be stored.

For detailed information about the Transfer option, see section 12, "Transferring an application."

4.5 Connecting the PC to a PLC

In the offline state, select **Connect** from the PLC menu to initiate communication between the PC and the PLC. For detailed information about the Connect option, see section 13.1.

4.6 Opening a binary file

A binary file has an *.app* file extension and contains the program and the configuration data in binary machine code.

Generally, a binary file is opened from a PC card (memory card). You can use the PC card to transfer an application from the FTX 117 terminal to the PC (TFTX REM/RSM cards only). For more information, see appendix A.7 in part G.

5.1 Introduction

The Configuration options are the hardware and software resources of the PLC. You can configure these resources before you write a program, while you write a program, or after you have completed a program.

The Configuration option can be selected by one of the following :

1. A resource can be configured from the Configuration menu on the main menu bar.
2. A resource can be configured from the Configuration Editor, accessed from the View menu.
3. A function block can be configured directly from the Ladder Editor or Ladder Viewer window.
4. A resource can be configured from the Object Browser dialog box in the Symbol Editor.

5.1-1 Configuration menu on the main menu bar

You configure resources from the Configuration menu on the main menu bar. Use the Configuration menu when you want to configure single objects as you construct other parts of the application.

5.1-2 Configuration Editor, accessed from the View menu

A resource can be configured from the Configuration Editor, which you access from the View menu. When you use the Configuration Editor, the Configuration Editor tool bar is displayed.

The buttons on the tool bar correspond to resources in the Configuration menu and Tools menu. Appendix A.2-1 in part G lists the options on the Configuration and Tools menu with the corresponding buttons on the tool bar.

When a resource is selected with the Configuration Editor window open, the current configuration of the selected resource appears. To configure Timers, Counters, Constants, and LIFO/FIFO Registers, select the resource directly from the list displayed. For Change PLC Version, the PLC Version Management dialog box appears instead of a list. To configure the other resources, select configuration editor from the View menu to display the appropriate dialog box.

5.1-3 Configuring a resource from the Ladder Editor or Ladder Viewer window

A resource can be configured directly from the Ladder Editor or Ladder Viewer window. Select the desired resource by double-clicking with the mouse or pressing **<enter>** on the resource in the Ladder program. The dialog box that appears allows you to configure the resource.

5.1-4 Configuring a resource from the Symbol Editor window

A resource can be configured from the Symbol Editor window via the Object Browser dialog box. The Symbol Editor window is selected from the View menu. Select the desired resource from the Symbol Editor list. The Object Browser dialog box appears. Select the Configure radio button to configure the resource.

C

5.2 Validate Program

Validate Program is available from the Tools menu when the Configuration Editor is displayed. See Section 10.1 for a full description of Validate Program.

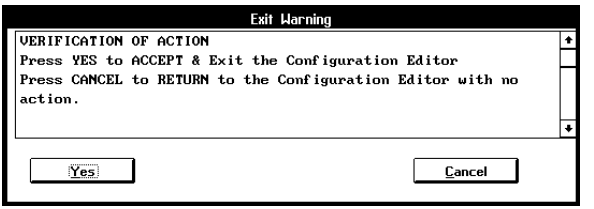
5.3 Validate Configuration

Validate Configuration is available from the Tools menu when the Configuration Editor is active.

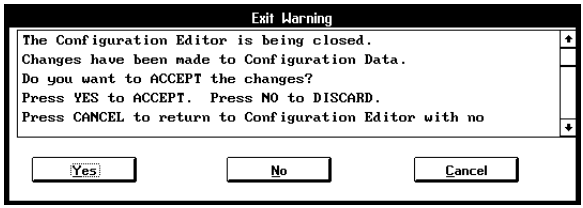
After changes or additions have been made to the configuration information :

1. Select **Validate Configuration** from the Tools menu. An Exit Warning dialog box appears.

Note : After configuration information is changed or added, Validate Configuration must be run to commit the new configuration information to the application. If you try to print, transfer, or save the application without running Validate Configuration, the previous information will be used.



2. Select **Yes** to accept the changes and exit the Configuration Editor. Select **Cancel** to return to the Configuration Editor.
3. If Validate Configuration is not run after changes have been made, and you attempt to exit the Configuration Editor window, an Exit Warning dialog box appears.



4. Select **Yes** to accept the changes and exit the Configuration Editor. Select **No** to discard the changes and exit the Configuration Editor. Select **Cancel** to return to the Configuration Editor.

5.4 Cancel Configuration

Select **Cancel Configuration** to exit the Configuration Editor window without making any configuration changes to the application file.

5.5 Application Name

The Application Name is printed on the first line of an application printout. The application name and the file name can be the same name or a different name.

Using a method described in Sections 5.1-1 through 5.1-4, select **Application Name** to display the current name of the application. Select Edit to display the Application Name dialog box.

1. Enter or change the name for the current application.

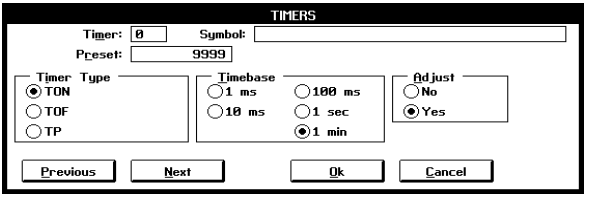
   Range : 1-8 characters

   Default : No default value.
2. Select **Ok** to commit the name entered. Select **Cancel** to close the dialog box without entering or changing the application name.

5.6 Timers

For general information about Timers, see Section 2.2-3 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Timers**. All of the Timers are listed. Select the Timer to be configured. The Timers dialog box appears.



1. Enter the **Timer** number, if it is different from the one selected. For example, to identify timer %TM0, enter 0 in the Timer field.
   Range : 0-31
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. Select a **Timer Type**.

   Range : TON (Timer On-delay), TOF (Timer Off-delay), TP (Timer Pulse).

   Default : TON.
4. Select the **Timebase** value, which is the unit of time of the Timer.

   Range : 1 millisecond (%TM0 and %TM1 only), 10 milliseconds, 100 milliseconds, 1 second, 1 minute.

   Default : 1 minute
5. Select the **Adjust** value. Select Yes or No to indicate whether a user can adjust the Preset value in the Data Editor.

   Default : Yes
6. Select **Previous** or **Next** to scroll to the Timer listed just before or after the current Timer without closing the Timers dialog box. These buttons are only shown when the Timers resource is accessed from the Configuration Editor window.
7. Select the **Preset** value.

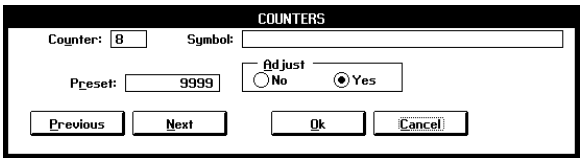
   Range : 0 to 9999

   Default : 9999
8. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.7 Counters

For general information about Counters, see Section 2.2-4 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Counters**. All of the Counters are listed. Select the Counter to be configured. The Counters dialog box appears.



1. Enter the **Counter** number, if it is different from the one selected. For example, to identify Counter %C8, enter 8 in the Counter field.

   Range : 0 to 15

2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.

3. Select the **Adjust** value. Select Yes or No to indicate whether a user can adjust the Preset value in the Data Editor.

   Default : Yes

4. Select a **Preset** value.

   Range : 0 to 9999

   Default : 9999

5. Select **Previous** or **Next** to scroll to the Counter listed just before or just after the current Counter without closing the Counters dialog box. These buttons are only shown when the Counters resource is accessed from the Configuration Editor window.

6. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

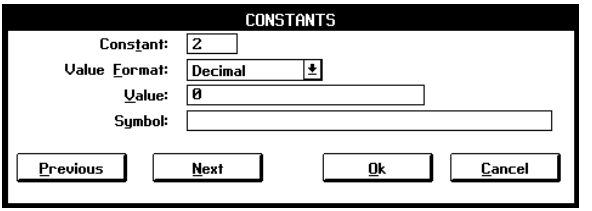
5.8 Constants

For general information about Constants, see Section 3.1-1 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Constants**. All of the Constants are listed. Select the Constant to be configured. The Constants dialog box will display.

1. Enter the **Constant** number, if it is different from the one selected. For example, to identify Constant %KW2, enter 2 in the Constant field.

   Range : 0-63



2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. Select a **Value Format**.

   Range : Decimal, Hexadecimal, Binary, or ASCII.

   Default : Decimal
4. Enter a value for the **Value** field, which is the value of the Constant.

   Decimal Range : -32768 to 32767

   Hexadecimal Range : 0000 - FFFF

   Binary Range : 0000000000000000 to 1111111111111111

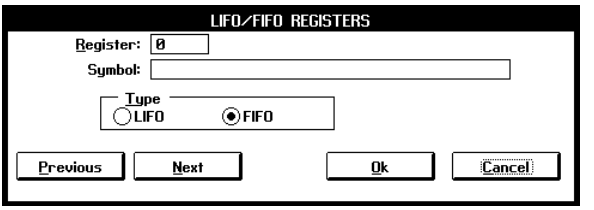
   ASCII Range : Any ASCII character

   Default : 0, decimal
5. Select **Previous** or **Next** to scroll to the Constant listed just before or after the current Constant without closing the Constants dialog box. These buttons are only shown when the Constants resource is accessed from the Configuration Editor window.
6. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.9 LIFO/FIFO Register

For general information about LIFO/FIFO Registers, see Section 2.2-5 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **LIFO/FIFO Registers**. All of the registers are listed. Select the register to be configured. The LIFO/FIFO Registers dialog box will display.



1. Enter the **LIFO/FIFO Register** number, if it is different from the one selected. For example, to identify LIFO/FIFO Register %R0, enter 0 in the Register field.

   Range : 0-4

2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. Select the Register **Type** :

   FIFO - The first data item entered in the register is the first to be retrieved, also known as a queue.

   LIFO - The last data item entered in the register is the first to be retrieved, also known as a stack.

   Default : FIFO

4. Select **Previous** or **Next** to scroll to the register listed just before or after the current register without closing the LIFO/FIFO Registers dialog box. These buttons are only shown when the LIFO/FIFO Register resource is accessed from the Configuration Editor window.
5. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.10 Drum Controllers

For general information about Drum Controllers, see Section 2.2-6 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Drum Controllers**.

1. Enter the **Drum** number. For example, to identify Drum Controller %DR2, enter 2 in the **Drum** field.

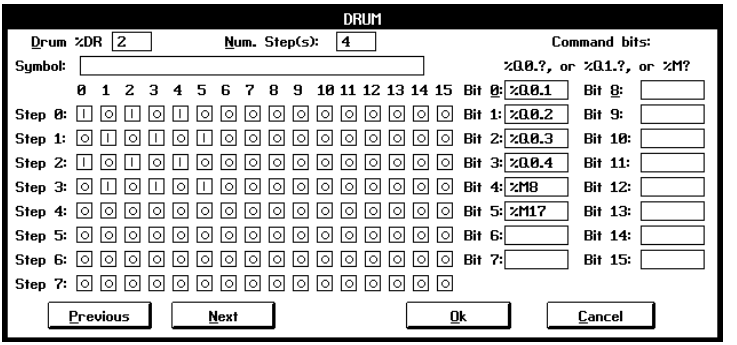
   Range : 0 to 3

2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. In the **Num. Steps** field, enter the number of steps in the drum.

   The check boxes for the number of steps you define remain active. The other check boxes are grayed out. For example, if four steps are defined in the **Num. Steps** field, then the first four rows of check boxes are enabled.

   Range : 1 to 8

4. In the **Command bits** field, assign either a physical bit or a memory bit to each of the 16 (0-15) logical bits that you want to use.



C

Range :

%Q0.0-%Q0.9 : Output bit on the base PLC.

%Q1.0-%Q1.9 : Output bit on an I/O PLC extension.

%M0-%M127 : Internal memory bit.

Default : No default values

5. For each step that you defined, click on the check boxes for each logical bit that you want to activate in each step. For example, if you run Step 0, three logical bits are activated : %Q0.1, %Q0.3, and %M8, as indicated by the "1" or ON indicator in each check box.
6. Select **Previous** or **Next** to scroll to the Drum Controller listed just before or just after the current Drum Controller without closing the Drum dialog box. These buttons are only shown when the Constants resource is accessed from the Configuration Editor window.
7. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

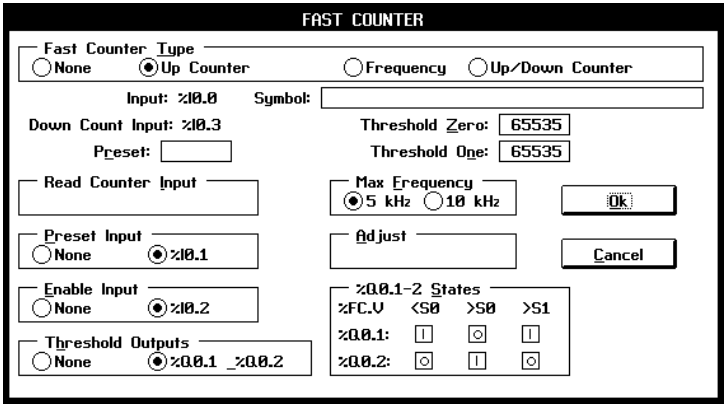
5.11 Fast Counter

The Fast Counter is a single resource, which you can configure as one of three counter types : an Up Counter, a Frequency Meter, or an Up/Down Counter. For more information about the Fast Counter, see Section 4.4 in Part A and Section 3.4-5 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Fast Counter**.

5.11-1 Up Counter

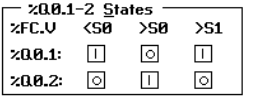
For the Up Counter, the counting input is always from input %I0.0. The counting input is not configurable. When the state of input %I0.0 changes from 0 to 1, then the current value of the Fast Counter (%FC.V) is incremented by 1.



1. Select **Up Counter** in the Fast Counter Type field. The following fields become active.
   - Preset Input
   - Enable Input
   - Threshold Outputs
   - Threshold Zero
   - Threshold One
   - Max Frequency
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. In the **Preset Input** field, select **%I0.1** to dedicate input %I0.1 for the Up Counter function. When the state of input %I0.1 changes from 0 to 1, the current value of the Up Counter (%FC.V) is set to 0. Select **None** if you do not want to use input %I0.1 as a dedicated input.
4. In the **Enable Input** field, select **%I0.2** to dedicate input %I0.2 to the Up Counter function. If the state of %I0.2 is 1, the Up Counter is turned on; the counter counts pulses from input %I0.0 and updates the current value of the count. If the state of %I0.2 is 0, then the Up Counter is disabled and the pulses at %I0.0 are ignored. Select **None** if you do not want to use input %I0.2 as a dedicated input.

5. In the **Threshold Outputs** field, select **%Q0.1 _%Q0.2** to dedicate these outputs for the Up Counter function. The state of these two outputs is a function of the Threshold Zero and Threshold One values' (%FC.S0 and %FC.S1, respectively) relationship to the Fast Counter current value (%FC.V) as defined in the %Q0.1-2 States field.

   For example, in the %Q0.1-2 States field shown :



   When %FC.V is less than %FC.S0, output %Q0.1 is 1 and output %Q0.2 is 0.

   When %FC.V is greater than %FC.S0 and less than %FC.S1, output %Q0.1 is 0 and output %Q0.2 is 1.

   When %FC.V is greater than %FC.S1, output %Q0.1 is 1 and output %Q0.2 is 0.

   Select **None** if you do not want to use outputs %Q0.1 and %Q0.2 as dedicated outputs.

6. In the **Threshold Zero** and **Threshold One** fields, enter the desired values.

   Range : 0 to 65535

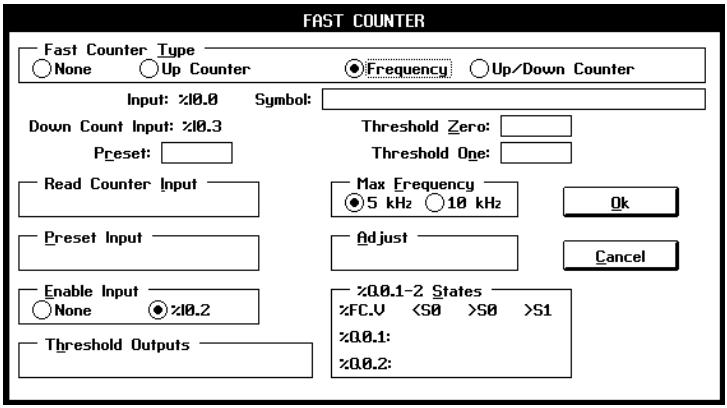
   Default : 65535

7. In the **Max Frequency** field, select **5 kHz** for normal mode or **10 kHz** for fast mode.
8. Configure the relationship between %FC.S0-S1 and %Q0.1-2 in the **%Q0.1-2 States** field. %Q0.1 _%Q0.2 must be selected in the Threshold Outputs field for this field to be active. In each box, enter 0 or 1, as desired.
9. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.11-2 Frequency Meter

For the Frequency Meter, the frequency counting input is from input %I0.0 and is not configurable.

1. Select **Frequency** in the Fast Counter Type field. The following fields become active.
   - Enable Input
   - Max Frequency
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. For the **Enable Input** field, select %I0.2 to dedicate input %I0.2 to the frequency meter function. If %I0.2 is at 1, the frequency meter is turned on; that is, the input counts pulses and updates the current value of the count. If %I0.2 is 0, then the frequency meter is disabled and the pulses at %I0.0 are ignored. Select **None** if you do not want to use input %I0.2 as a dedicated input.

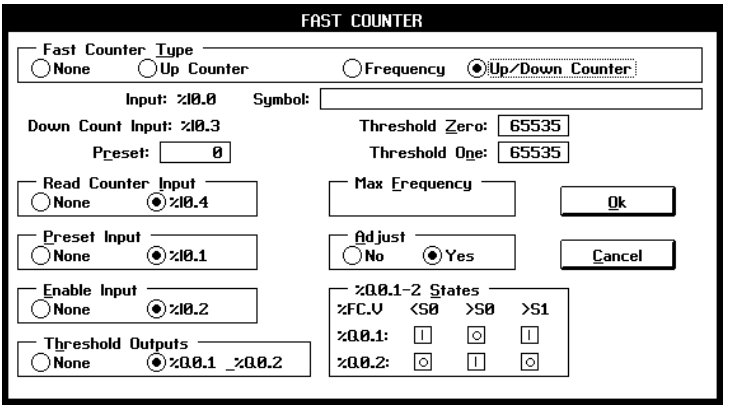


4. In the **Max Frequency** field, select **5 kHz** for normal mode or **10 kHz** for fast mode.
5. Select **Ok** to commit the values you selected or **Cancel** to close the dialog box without changing the configuration.

5.11-3 Up/Down Counter

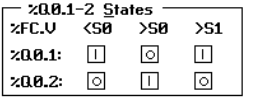
For the Up/Down Counter, the up counting input is from input %I0.0. The down counting input is from %I0.3. Neither input is configurable.

1. Select **Up/Down Counter** in the Fast Counter Type field. The following fields become active.
   - Preset
   - Read Counter Input
   - Preset Input
   - Enable Input
   - Threshold Outputs
   - Threshold Zero
   - Threshold One
   - Adjust
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. In the **Preset** field, select the number of events to be counted by the Up/Down Counter.
4. In the **Read Counter Input** field, select **%I0.4** to dedicate input %I0.4 for the Up/Down Counter function. When input %I0.4 is 1 or ON, the current value of the fast counter is written to a system memory word in the program. Select **None** if you do not want to use input %I0.4 as a dedicated input.



5. In the **Preset Input** field, select **%I0.1** to dedicate input %I0.1 for the Up/Down Counter function. When the state of input %I0.1 changes from 0 to 1, the current value of the Up/Down Counter (%FC.V) is set to 0. Select **None** if you do not want to use input %I0.1 as a dedicated input.
6. In the **Enable Input** field, select **%I0.2** to dedicate input %I0.2 to the Up/Down Counter function. If the state of %I0.2 is 1, the Up/Down Counter is turned on; the counter counts pulses from inputs %I0.0 and %I0.3 and updates the current value of the count. If the state of %I0.2 is 0, then the Up/Down Counter is disabled and the pulses at %I0.0 and %i0.3 are ignored. Select **None** if you do not want to use input %I0.2 as a dedicated input.
7. In the **Threshold Outputs** field, select **%Q0.1 _%Q0.2** to dedicate these outputs for the Up/Down Counter function. The state of these two outputs is a function of the Threshold Zero and Threshold One values' (%FC.S0 and %FC.S1, respectively) relationship to the Fast Counter current value (%FC.V) as defined in the %Q0.1-2 States field.

   For example, in the %Q0.1-2 States field shown :



   When %FC.V is less than %FC.S0, output %Q0.1 is 1 and output %Q0.2 is 0.

   When %FC.V is greater than %FC.S0 and less than %FC.S1, output %Q0.1 is 0 and output %Q0.2 is 1.

   When %FC.V is greater than %FC.S1, output %Q0.1 is 1 and output %Q0.2 is 0.

   Select **None** if you do not want to use outputs %Q0.1 and %Q0.2 as dedicated outputs.
8. In the **Threshold Zero** and **Threshold One** fields, enter the desired values.

   Range : 0 to 65535

   Default : 65535

9. In the **Adjust** field, select **Yes** or **No** to indicate whether a user can adjust the Preset value in the Data Editor.
10. Configure the desired relationship between %FC.S0-S1 and %Q0.1-2 in the **%Q0.1-2 States** field. %Q0.1 _%Q0.2 must be selected in the Threshold Outputs field for this field to be active. In each box, enter 0 or 1, as desired.
11. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.12 %PLS

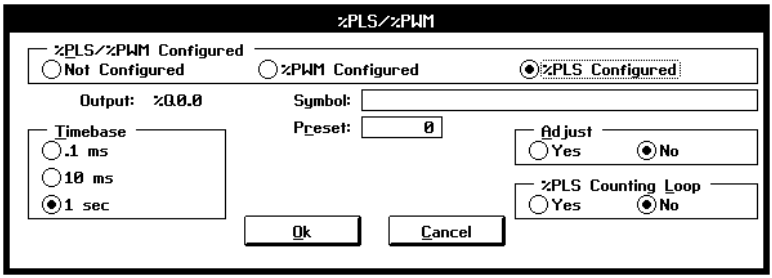
The %PLS/%PWM Pulse Generator function block is used as a square wave generator. For %PLS, the on time equals the off time for a period (50% duty cycle). For %PWM, the signal width (duty cycle) can be varied. For more information about the Pulse Generator, see Sections 4.5 and 4.6 in Part A and Sections 3.4-3 and 3.4-4 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **%PLS/%PWM**.

5.12-1 %PLS Configured

For the %PLS pulse generator, the output is from %Q0.0 and is not configurable.

1. In the **%PLS/%PWM Configured** field, select **%PLS Configured**. The following fields become active.
   - Timebase
   - Preset
   - Adjust
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.



Note : If you configure either the %PLS or the %PWM, output %Q0.0 is dedicated to %PLS or %PWM processing and should not be assigned to any other function in the program.

3. For the **Timebase** field, select the unit of time. When .1 ms is selected the %PLS Counting Loop field becomes active.

   Range : .1 ms, 10 ms, or 1 sec

   Default : 1 second

4. In the **Preset** field, specify the length of the period in timebase units.

   Range : If Timebase = 10 ms or 1 second, the value can be between 0 and 32767. If Timebase = 0.1 ms, the value can be between 0 and 255.

   Default : 0

5. Select the **Adjust** value and then select **Yes** or **No** to indicate whether a user can adjust the Preset value in the Data Editor.
6. If .1 ms was selected in the Timebase field, the **%PLS Counting Loop** field becomes active.

   Select **Yes** to implement the Fast Counter. The period of the pulse is less than the total scan time, requiring auxiliary processing from the Fast Counter. For the %PLS to work properly, physically loop output %Q0.0 to input %I0.0.

   Select **No** if you choose not to loop to the Fast Counter to perform auxiliary processing.

7. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

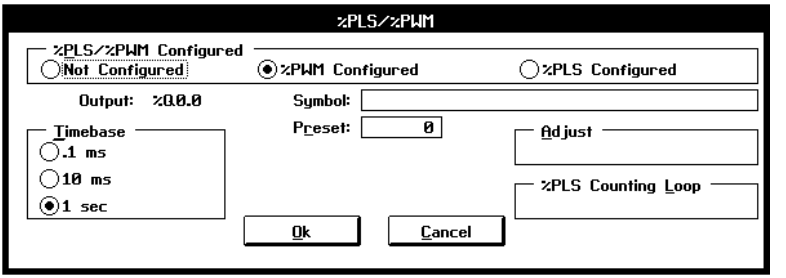
5.12-2 %PWM Configured

For the %PWM pulse generator, the output is from %Q0.0. This is not configurable.

1. In the **%PLS/%PWM Configured** field, select **%PWM Configured**. The following fields become active.
   - Timebase
   - Preset
2. The **Symbol** field is not active in this dialog box. This field is for information only. You can only assign a symbol to a data variable in the Symbol Editor. For more information, see Section 6.
3. For the **Timebase** field, select the unit of time.

   Range : .1 ms, 10 ms, or 1 sec

   Default : 1 second

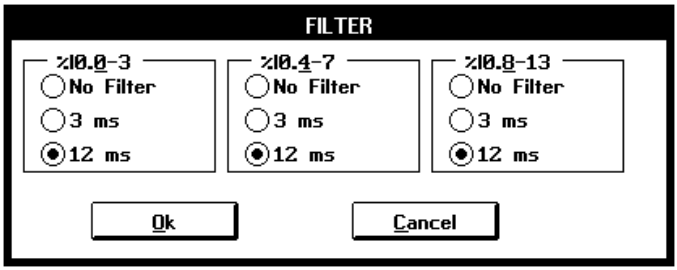


4. In the **Preset** field, specify the length of the period in timebase units.
   Range : 0 to 9999
   Default : 0
5. Select **Ok** to commit the values selected or **Cancel** to close the dialog box without changing the configuration.

5.13 Input Filters

Input Filters is used to reduce the effect of noise on the input. For more information, see Section 1.7-1 in Part A.

Using a method described in Sections 5.1-1 through 5.1-4, select **Input Filters**.

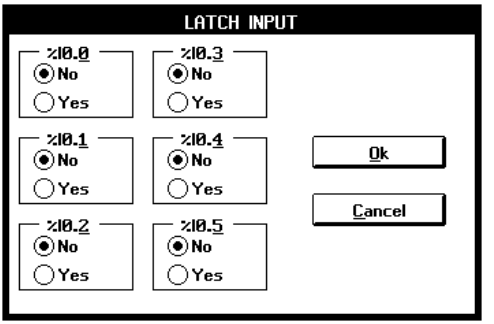


1. In each input field, **%I0.0-3**, **%I0.4-7**, and **%I0.8-13**, select **No Filter**, **3 ms**, or **12 ms**.
   If No Filter is selected, the input filtering device is not activated on the inputs.
   If 3 ms is selected, any pulses which are 3 ms or less in duration are filtered.
   If 12 ms is selected, any pulses which are 12 ms or less in duration are filtered.
   Default : 12 ms
2. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

5.14 Latch Input

Latch Input captures and records an incoming pulse on an input. For more information, see Section 4.3 in Part A.

Using a method described in Sections 5.1-1 through 5.1-4, select **Latch Input**. The current Latch Input configuration is displayed. Select **Edit** to display the Latch Input dialog box.



C

1. For each input, %I0.0 through %I0.5 :

   Select **Yes** to enable Latch Input processing for the selected input.

   Select **No** to disable Latch Input processing for the selected input.

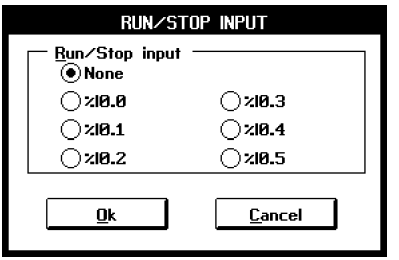
   Default : No

2. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

5.15 Run/Stop Input

Run/Stop is used to start or stop the PLC from an external input. A stop command from the PL7-07 PC programming software or the FTX 117 overrides a run command from the Run/Stop input. For more information, see Section 4.1 in Part A.

Using a method described in Sections 5.1-1 through 5.1-4, select **Run/Stop Input**.



1. Select the desired input to the PLC for the Run/Stop input.

   Range : Inputs %I0.0 through %I0.5

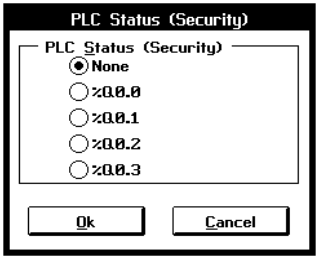
   Default : None

2. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

5.16 PLC Status (Security)

Use PLC Status (Security) to configure an output to indicate whether the PLC is running. For more information, see Section 4.2 in Part A.

Using a method described in Sections 5.1-1 through 5.1-4, select **PLC Status (Security)**.



1. In the **PLC Status (Security)** field, select a designated output to provide the status of the PLC.

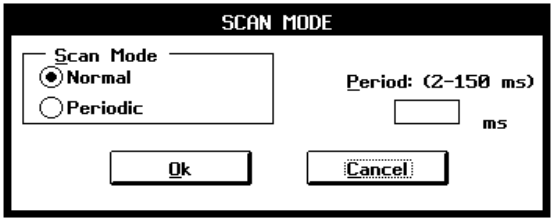
   Range : Outputs %Q0.0 through %Q0.3

   Default : None
2. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

5.17 Scan Mode

Use Scan Mode to define whether you want to scan your program in normal or periodic mode. For more information, see Section 1.3 in Part A.

Using a method described in Sections 5.1-1 through 5.1-4, select **Scan Mode**.



1. In the **Scan Mode** field :

   Select **Normal** to scan a program one operational cycle after another, regardless of the length of the scan.

Select **Periodic** to define the minimum rate at which the CPU scans a program in the PLC. You define the minimum rate in the Period field.

Default : Normal

2. In the **Period** field, type a value from 2 to 150 milliseconds.

   Default : None

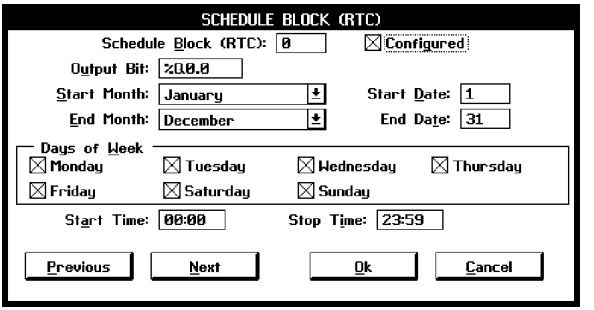
3. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

C

5.18 Schedule Blocks

For general information about Schedule Blocks, see Section 5.2 in Part B.

Using a method described in Sections 5.1-1 through 5.1-4, select **Schedule Blocks**.



1. Select the **Configured** box to activate all fields in the dialog box.
2. Enter the **Schedule Block Number**. You can define up to 16 schedule blocks in a program.

   Range : 0-15.

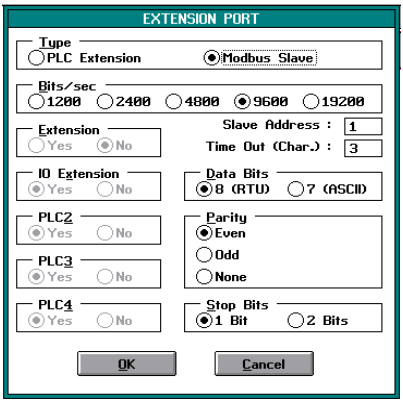
3. Enter an **Output Bit** address. For example, if you are configuring a sprinkler to turn on, enter the address of the output terminal connected to the sprinkler system.
4. Select the **Start Month** (the month that you want to activate the schedule block output).
5. Select the **End Month** (the month that you want to de-activate the schedule block output).
6. Select the **Start Date** (the day of the month that you want to activate the schedule block output).
7. Select the **End Date** (the day of the month that you want to de-activate the schedule block output).

8. Select the **Days of Week** that you want to activate the schedule block output. Checked boxes indicate the days that the output device is activated.
9. Enter the **Start Time** (the hour that you want to activate the schedule block output).
   Default : 00:00
10. Enter the **End Time** (the hour that you want to de-activate the schedule block output).
    Default : 23:59
11. Select **Previous** or **Next** to scroll to the Schedule Block listed just before or just after the current Schedule Block without closing the Schedule Block dialog box. These buttons are only shown when the Schedule Block resource is accessed from the Configuration Editor window.
12. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

5.19 Extension Port

Use Extension Port to define the PLC communications configuration.

Using a method described in Sections 5.1-1 through 5.1-4, select **Extension Port**.



Select **PLC Extension** to configure the I/O extension link or **Modbus Slave** for a Modbus Slave type link.

PLC Extension :

1. The **Extension** field is used to detect communication errors with the I/O extension. Select **Yes** if you want the base PLC to generate an error when communication is not possible with the I/O extension.

Select **No** if you do not want the base PLC to generate an error when no communication is received from one of the PLCs defined in the dialog box.

Default : No

2. In the **Bits/sec** field, indicate the speed at which you want connected PLC units to communicate. Communication quality decreases, or becomes less reliable, as distance increases or as electrical noise in the environment increases. Therefore, when communicating across large distances or in a noisy environment, decrease the communication rate to increase reliability.

Note

The same speed must be configured for all PLCs in the same Nanet network.

Select **9600** bits per second if the distance between PLC units is relatively large or if there is a substantial amount of electrical noise in the environment.
Select **19200** bits per second if the distance between PLC units is relatively short or if there is relatively little electrical noise in the environment.

Default : 19200

3. In the **I/O Extension** field, indicate whether you have a PLC configured and connected as an I/O extension. The I/O Extension supplies additional inputs and outputs to the base PLC. For more information, see Section 3.5 in Part A.

Default : No

5. In the **PLC2**, **PLC3**, and **PLC4** fields, select **Yes** or **No** to indicate whether you have a corresponding PLC configured as a peer PLC connected to the base PLC.

Defaults : No

6. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the configuration.

Modbus Slave : (see Section F-2.1)

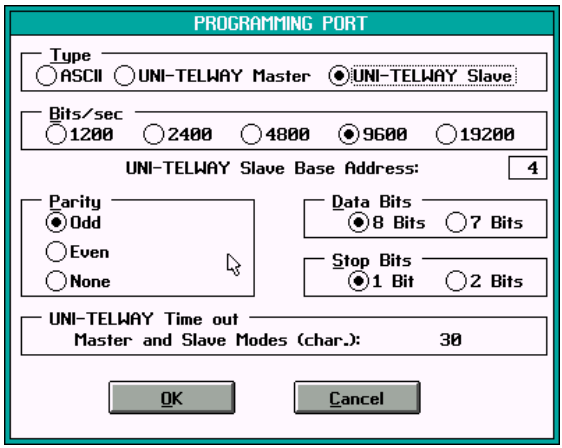
1. The **Bits/sec** field indicates the data rate of the Modbus link.
2. The **Slave Address** field configures the address of the TSX Nano Modbus slave (1 to 98).
3. The **Time-out (Char)** field sets the parameters for the intercharacter time (1 to 127 characters).
4. The **Data Bits** field selects ASCII mode (7 bits) or RTU mode (8 bits).
   It specifies the size of the data on the line.
5. The **Parity** field defines the addition or not of a parity bit as well as its type.
6. The **Stop Bits** field specifies the number of stop bits used.

Note

These parameters should be the same as those for the Modbus link Master.

5.20 Programming Port

Use the programming port box to select the type of protocol for the TSX Nano PLC programming port.



ASCII mode

Select **ASCII** to configure the TSX Nano terminal port in ASCII mode.

1: The **Bits/sec** field specifies the transmission speed of data on the line. Available speeds are : 1200, 2400, 4800, 9600 and 19200 bits/second.
2: The **Data Bits** field selects the size of the data exchanged on the line.
3: The **Parity** field defines the addition or not of a parity bit as well as its type.
4: The **Stop Bits** specifies the number of stop bits used.
5: The **UNI-TELWAY Time-out (Char.)** field is used to set the value of the Uni-telway time-out.

UNI-TELWAY Slave mode

Select **UNI-TELWAY Slave** to configure the TSX Nano terminal port in UNI-TELWAY Slave mode :

1: The **UNI-TELWAY slave address** field is used to select the AD0 link address of the TSX Nano.
The PLC uses 2 consecutive logic addresses :
AD0 : base address (configuration address) is the server address, which all the UNI-TELWAY bus devices use to send requests to the TSX Nano.
AD1: AD0+1 is the client address used by the EXCH block to send requests to other UNI-TELWAY bus devices.
2: The other fields are the same as for ASCII mode and should be identical to those of the UNI-TELWAY bus Master.

UNI-TELWAY Master mode

The format of this protocol is fixed. Only the Time-out can be configured (30 to 255).

To configure Time-out in **UNI-TELWAY Master** mode, modify its value in the **UNI-TELWAY Time-out (Char.)** field.

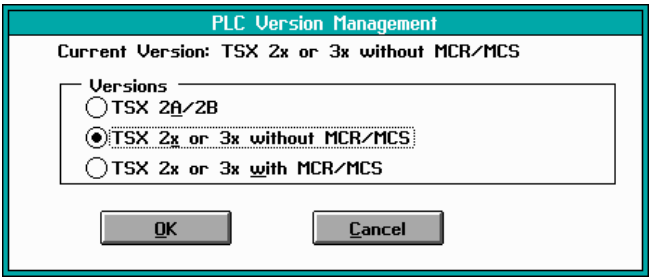
C

5.21 Change PLC Version

Use the Change PLC Version options to specify the model number or version of the TSX Nano Programmable Logic Controller that you are using as the base PLC.

Using a method described in Sections 5.1-1 through 5.1-4, select **Change PLC Version**. The PLC Version Management dialog box appears.

1. Locate the model number or version of your TSX Nano PLC on the left side panel :



   TSX 2A/2B : version 1 of the TSX Nano PLC.

   TSX 2x or 3x without MCR/MCS : Provides all functionality, except the Master Control Relay functions.

   TSX 2x or 3x with MCR/MCS : Provides all functionality, including the Master Control Relay functions. Use of this function will slightly improve performance.

   Default : TSX 2x or 3x without MCR/MCS

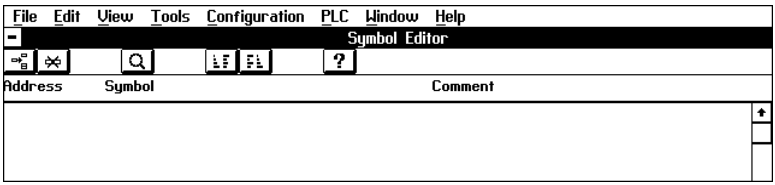
2. Select **Ok** to commit the values selected or **Cancel** to exit the dialog box without changing the PLC version.

6.1 Introduction

In the Symbol Editor, you assign easily recognizable alphanumeric names, called *symbols*, to data variables in your program. Symbols can help you quickly examine and analyze your program logic, greatly simplifying the development and testing process. You can display the Symbol Editor offline and online, but it is restricted in the monitor state.

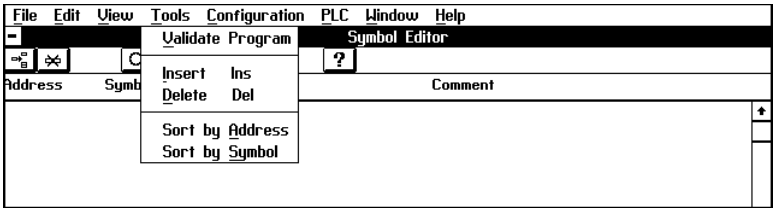
6.2 Selecting the Symbol Editor

To display the Symbol Editor window, select **Symbol Editor** from the View menu.



6.3 Using the Symbol Editor Tools menu

The Symbol Editor window is used for defining symbols for the program addresses. The table in appendix A.2-2 in part G lists the options on the Symbol Editor Tools menu and the corresponding buttons on the tool bar.



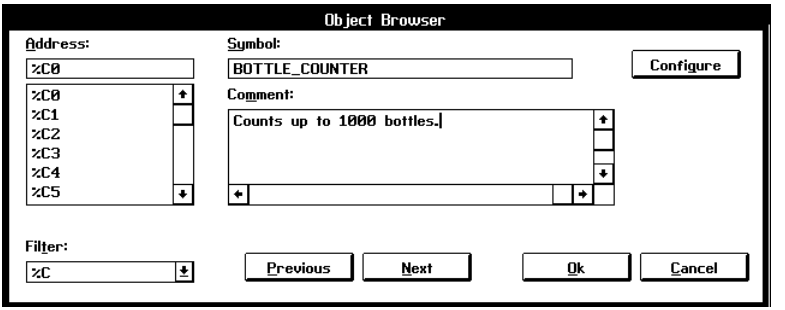
6.3-1 Validate Program

Validate Program is used to compile a program and check for errors. For more information, see section 10.1, "Validating a program."

6.3-2 Insert

Insert is used to add an object to the symbol table.

1. From the Symbol Editor window, select **Insert**. The Object Browser dialog box is displayed.



2. In the **Address** field, enter the variable address that you want the symbol to represent. You can enter an address in two ways.

   • If you know the address, you can type it directly in the **Address** field selection box.

   • To assist you in selecting the correct address, open the **Filter** list box and select a variable type.

   The Address list box automatically lists all instances of the type selected.

3. In the **Symbol** field, type a symbol that describes the variable, such as BOTTLE_COUNTER.

   Guidelines:

   • A symbol cannot be longer than 32 characters.

   • A symbol can only contain letters (A-Z), numbers (0-9), underscores (_) and some accented characters. When an unauthorized character is entered, a dialog box appears on the screen to help the user. The authorized characters correspond to page 850 of the code.

   'A-ZÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖØÙÚÛÜÝÞŸß', '0-9', et '_' **(underlining)**

   • A symbol must start (first character) with an alphabetical or accented character.

   • A symbol cannot have any spaces or special characters.

   • A symbol is not case sensitive. For example, the symbol names *Pump1* and *PUMP1* are the same symbol and can only occur once in an application.

4. In the **Comment** field, type a description of the variable up to 128 characters.
5. If you want to configure the defined variable, select the **Configure** button. The appropriate dialog box appears. For more information, see section 5.
6. Select **Previous** or **Next** to select the object just before or after the current object in the symbol table.
7. Select **Ok** to commit the values or **Cancel** to exit the dialog box and return to the Symbol Editor window.

6.3-3 Delete

Delete is used to remove an object from the symbol table.

1. Select the object to be removed from the symbol table. Select **Delete**. A Warning dialog box appears to prompt you to confirm the deletion.
2. Select **Ok** to delete the object or **Cancel** to exit the dialog box and return to the Symbol Editor window.

6.3-4 Sort by Address

Sort by Address is used to list the symbol table, in alphabetic and numeric order, by address.

1. From the Symbol Editor window, select **Sort by Address**.
2. The symbol table is listed, in alphabetic and numeric order, by address.

6.3-5 Sort by Symbol

Sort by Symbol is used to list the symbol table, in alphabetic order, by symbol.

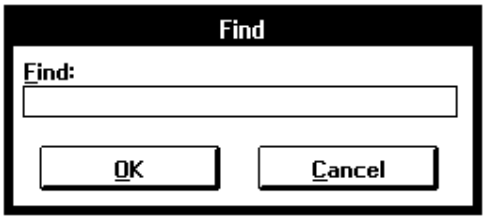
1. From the Symbol Editor window, select **Sort by Symbol**.
2. The symbol table is listed, in alphabetic order, by symbol.

6.4 Using the Symbol Editor Edit menu

With the Symbol Editor window open, Find is the only function that is available from the Edit menu.

Find is used to locate a symbol or address in the symbol table.

1. From the Symbol Editor window, select **Find**. The Find dialog box is displayed.



2. In the Find dialog box, enter the symbol or address to be located.
3. Select **Ok** to initiate the search or **Cancel** to return to the symbol table.

   If the symbol or address is found, the symbol table is displayed with the line containing the symbol or address highlighted.

4. For the case where you are trying to locate by symbol, if the symbol is not found, a Warning dialog box is displayed with the message: "Symbol not found. Would you like to create it?"

   Select **Ok** to display the Object Browser dialog box to create the symbol. Select **Cancel** to return to the symbol table.

5. For the case where you are trying to locate by address, if the address is not found, the Object Browser dialog box is displayed with the desired address shown in the Address field.

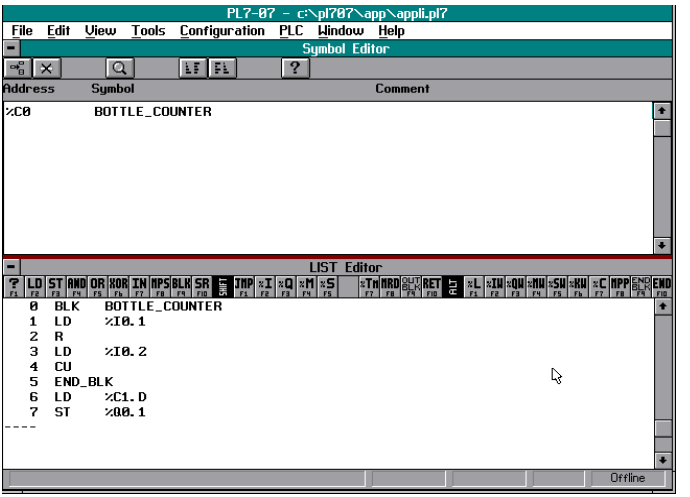
C

6.5 Editing a symbol

A symbol that has no variable address is an *unresolved* symbol. You can write a program with unresolved symbols, while you design your program logic, and then complete the symbol table after you complete the program. You can also choose to define symbols in the Symbol Editor before you write a program or while you write a program.

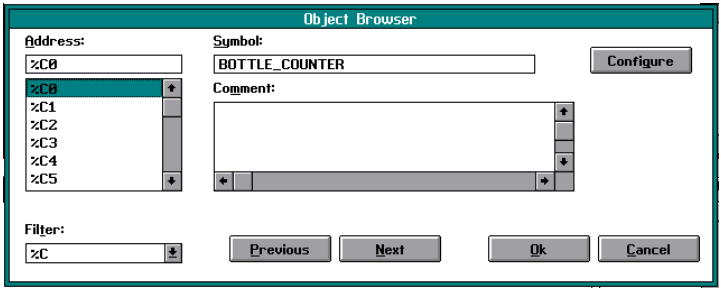
For example, in the program function block displayed in the List Editor, the symbol BOTTLE_COUNTER has no assigned address and is an unresolved symbol.

Select **Symbol Editor** from the View menu to display the symbol, BOTTLE_COUNTER, in the symbol field.



To define (or resolve) the symbol with an address:

1. Select the symbol in the Symbol Editor window to display the Object Browser dialog box.



2. In the **Address** field, assign an address to the symbol. If the address is already assigned, an Error dialog box is displayed indicating a duplicate entry exists. Select **Ok** to return to the Object Browser dialog box to assign another address.
3. After assigning the address to the symbol and making any other entries in the Object Browser dialog box, select **Ok** to commit the values or **Cancel** to exit the dialog box without making any changes.
4. After assigning addresses to unresolved symbols, Validate Program must be run to allow you to toggle between Show Addresses and Show Symbols in the List Editor window.

Note: When editing an existing symbol, if you re-name the symbol to a name not previously used as a symbol and use an address that is already assigned to an existing symbol, the new symbol name will be assigned to the existing address.

C

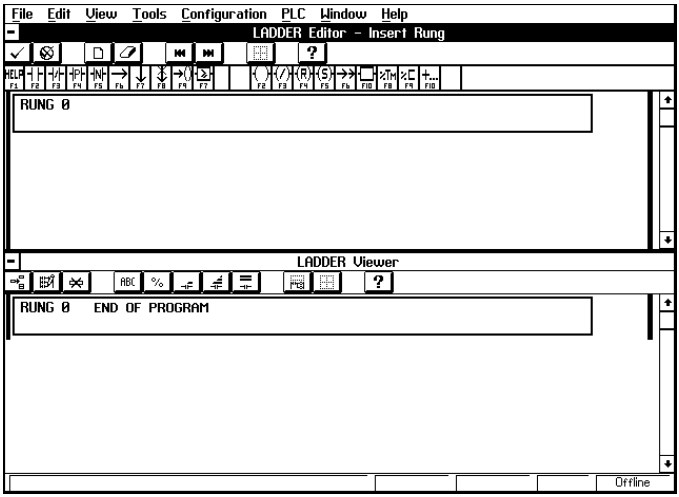
7.1 Introduction

The PL7-07 PC programming software provides a user-friendly method for programming in the Ladder format.

A ladder program is framed vertically by the power and return rails. Ladder rungs start on the left with the first connection of the rung to the power rail and end on the right with the outputs connected to the return rail.

Easily distinguishable icons located on tool bars and instruction bars allow the programmer to quickly and accurately build and edit the Ladder program. The selection and placement of the Ladder elements can be done with either the mouse or the keyboard.

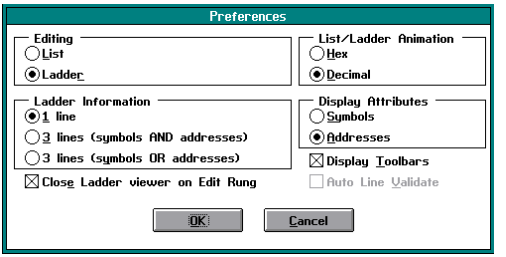
Editing of the Ladder diagram is done using a two-window format. The top window is the Ladder Editor and the bottom window is the Ladder Viewer. You use the Ladder Editor to insert and edit rungs. The Ladder Viewer window is used to scroll through and view a program. Before you begin to write a program, the Ladder Viewer displays the first rung number, Rung 0, and the END OF PROGRAM delimiter.



7.2 Configuring the Ladder Editor

To configure the Ladder editor :

1. Select **Preferences** from the View menu to display the Preferences dialog box.



2. In the **Editing** field, select **Ladder**.
3. After selecting Ladder in the Editing field, the **Ladder Information** field becomes active.

   Select **1 line** to display one line of either the symbol *or* address, depending on which one is selected in the Display Attributes field, defined in step 5.

   Select **3 lines (symbols AND addresses)** to display three lines with the symbol *and* the address at the same time. This option affects coils and contacts only. List rungs, compare blocks, and operate blocks will show only one line; either Symbols or Addresses depending on which one is selected in the Display Attributes field.

   Select **3 lines (symbols OR addresses)** to display three lines of either the symbol *or* the address, depending on which one is selected in the Display Attributes field, defined in step 5. This option affects coils and contacts only. List rungs, compare blocks, and operate blocks will show only one line; either Symbols or Addresses depending on which one is selected in the Display Attributes field.
4. You animate a program to display the current value of a variable in the PLC.

   In the **List/Ladder Animation** box, select Hexadecimal or Decimal format for displaying the current values when the program is animated.
5. In the **Display Attributes** field, select the desired attribute, either **Symbols** or **Addresses**, to be displayed. When **3 lines (symbols AND addresses)** is selected in the Ladder Information field, both attributes are displayed. Only coils and contacts are affected. List rungs, compare blocks, and operate blocks will show only one line.
6. Select **Display Toolbars** to display the tool bars in all editor windows.
7. Select **Close Ladder viewer on Edit rung** to display the full-screen Ladder editor when program modifications are being entered. The Ladder viewer will automatically reopen when the Ladder editor window is closed.
8. Select **Ok** to commit the values you selected. Select **Cancel** to exit the dialog box without changing the preferences selected.

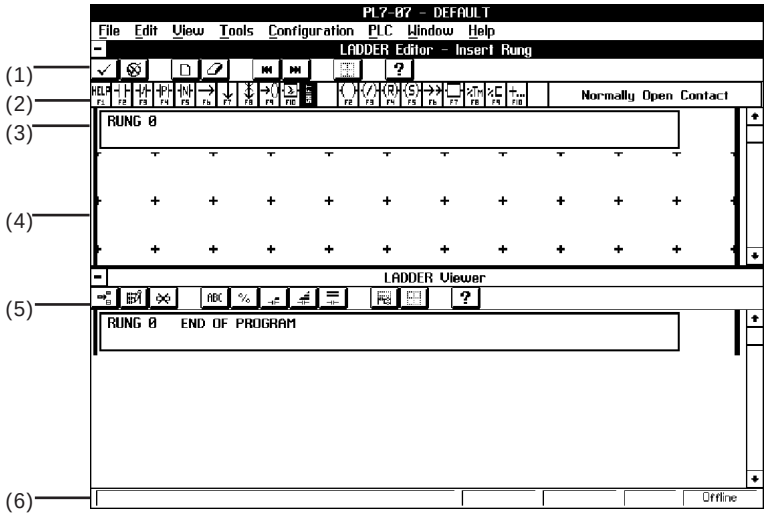
7.3 Using the Ladder Editor

1. Select **Ladder Editor** from the View menu. The Ladder Viewer window is displayed.
2. To display the Ladder Editor, select **Insert Rung** from the Tools menu.

Above each rung is a *rung header*. The rung header displays the rung number and can, optionally, display a title, a label or subroutine declaration, and comments. In the Ladder Viewer window the rung header can be toggled on or off using **Toggle Rung Header**.

The status bar is located at the bottom of the screen. The left side displays any error or informational messages that may occur from the entry of a program rung.

The two-window Ladder programming environment has two tool bars and one instruction bar.

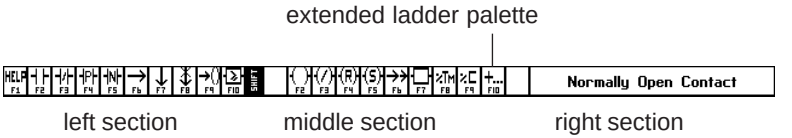


(1) Ladder Editor tool bar
(2) Ladder Editor instruction bar
(3) Rung header
(4) Programming grid
(5) Ladder Viewer tool bar
(6) Error message status bar

The icons on the tool bars correspond to options in the Tools menu. The Tools menu changes depending on which window is active. The Ladder Editor and Ladder Viewer each have a separate Tools menu.

The tables shown in appendix A.2-3 and A.2-3.1 in part G list the instructions that are available from the Ladder Editor instruction bar. The right side of the Ladder Editor instruction bar gives the name of the instruction that is selected.

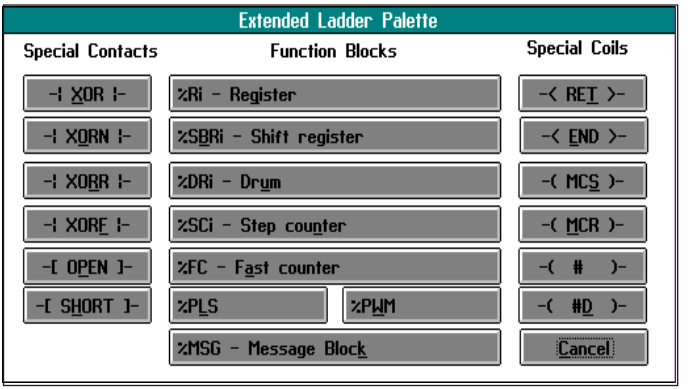
7.4 Inserting a graphic instruction



7.4-1 Rules for inserting graphic instructions

The following are rules for inserting graphic instructions in a Ladder rung :

1. From left to right, there are eleven columns in the grid. The graphic instructions located in the left section of the Ladder Editor instruction bar cannot be inserted in the last column of the grid. Additionally, the compare block instruction, which takes up two cells, cannot be inserted in the last two columns of the grid.
2. The coil, inverse coil, reset coil, set coil, and jump/subroutine call instruction can only be inserted in the last column of the grid. If you try to insert these instructions anywhere else, a horizontal connector line is automatically inserted from that point to the last column, where the instruction will be inserted.
3. The operate block, which takes up four cells, can only be inserted in the last four columns of the grid. If you try to insert this instruction anywhere else, a horizontal connector line is automatically inserted from that point to the last four columns, where the block will be inserted.
4. The timer and counter blocks, each which take up two horizontal cells, cannot be inserted in the first column of the grid or the last two columns of the grid.
5. The special contacts, located on the left side of the Extended Ladder Palette, cannot be inserted in the first and last column of the grid. The exceptions to this are the Open and Short special contacts, which can be inserted in the first column of the grid.
6. The function blocks, located on the Extended Ladder Palette, each of which take up two horizontal cells, cannot be inserted in the first column or the last two columns of the grid. There can only be one function block per rung.



7. The special coils, located on the right side of the Extended Ladder Palette, can only be inserted in the last column of the grid. If you try to insert these instructions anywhere else, a horizontal connector line is automatically inserted from that point to the last column, where the instruction will be inserted.

The Ladder Editor instruction bar is used for inserting graphic instructions.

To insert graphic instructions from the Ladder Viewer window :

1. Select **Insert Rung** from the Tools menu to display the Ladder Editor above the Ladder Viewer window.
2. If you want the programming grid to show, toggle the grid *on* from the Tools menu or from the icon in the Ladder Editor tool bar.

The graphic instructions can be entered either by using the mouse or the keyboard.

7.4-2 Inserting graphic instructions using the mouse

1. Select an instruction from the instruction bar by pointing to the instruction and clicking the **left mouse button**. The instruction name appears in the right section of the Ladder Editor instruction bar.

   For instructions on the Extended Ladder Palette, select the graphic instruction for the palette from the instruction bar. The Extended Ladder Palette display appears. Select the desired instruction from the palette using the **left mouse button**. The Ladder Editor window appears with the selected instruction's name listed in the right section of the instruction bar.
2. Place the instruction by pointing to the cell and clicking the **right mouse button**. The instruction selected stays active until another instruction is selected. To insert the same instruction in another cell, point to the cell and click the right mouse button. If you insert an instruction in a cell that already has an instruction, the previous instruction is overwritten.
3. Press **<delete>** to remove an instruction from a selected cell.

7.4-3 Inserting graphic instructions using the keyboard

1. Select an instruction from the instruction bar using the listed *function* key. For example, press **F2** to select a normally open contact. The name of the instruction will appear in the right section of the instruction bar.

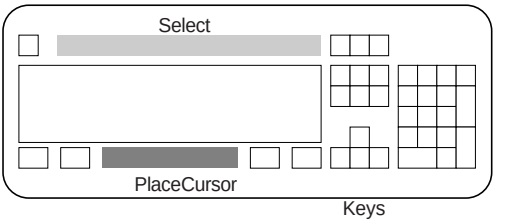
   For instructions on the Extended Ladder Palette, hold down the **<shift>** key and press **F10**. The Extended Ladder Palette appears. Select the desired instruction. The Ladder Editor window appears with the selected instruction's name listed in the right section of the instruction bar.
2. Select a cell in the Ladder Editor window using the arrow keys. Insert the instruction in the cell by pressing the **<spacebar>**.

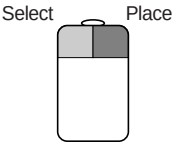
   The instruction selected stays active until another instruction is selected. To insert the instruction in another cell, select the cell and press the **<spacebar>**.

If you insert an instruction in a cell that already has an instruction, the previous instruction is overwritten.

3. Press **<delete>** to remove an instruction from a selected cell.

The following summarizes how to select and insert a graphic instruction, using either the mouse or the keyboard.



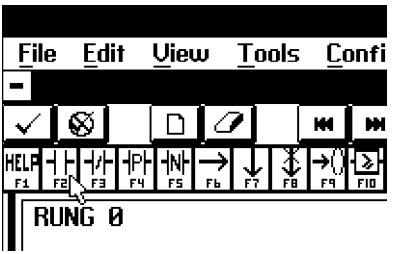


7.5 Inserting specific contacts, coils, and function blocks

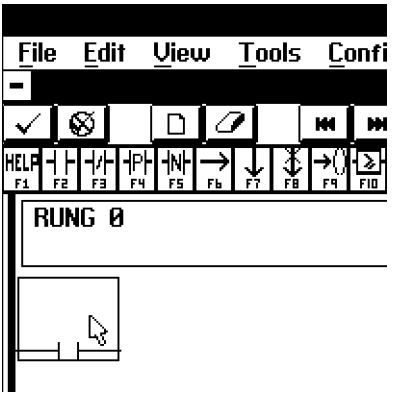
7.5-1 Inserting a contact

A contact can be inserted in any column except the last column of the grid. To insert a normally open contact, normally closed contact, rising edge contact, or falling edge contact :

1. Select the contact from the instruction bar or by pressing the corresponding function key.



2. Place the contact by clicking the right mouse button in a cell. Or, select a cell with the cursor keys and press the **<spacebar>**.

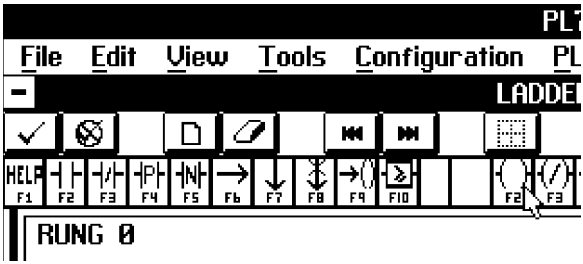


3. To complete the programming of the contact, insert an operand or symbol as described in Section 7.6.

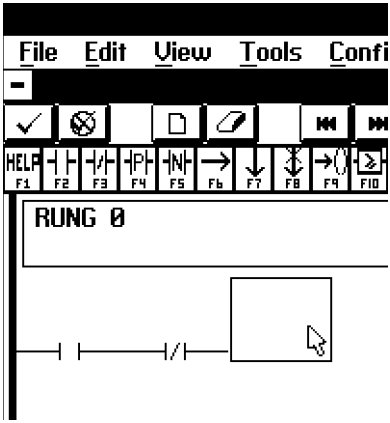
7.5-2 Inserting a coil or jump/subroutine call

A coil or jump/subroutine call can only be inserted in the last column of the grid. To insert a coil, inverse coil, reset coil, set coil, or jump/subroutine call :

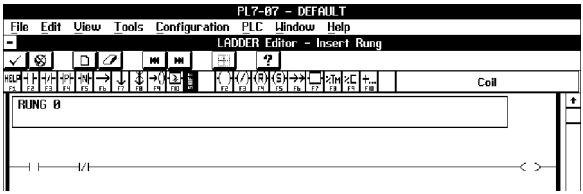
1. Select the coil or jump/ subroutine call from the instruction bar or by pressing the **<shift>** key plus the corresponding function key.



2. Select the cell after the last graphic instruction in the rung.



3. Place the coil or jump/ subroutine call by clicking the right mouse button in the cell, or select the cell with the cursor keys and press the **<spacebar>**.



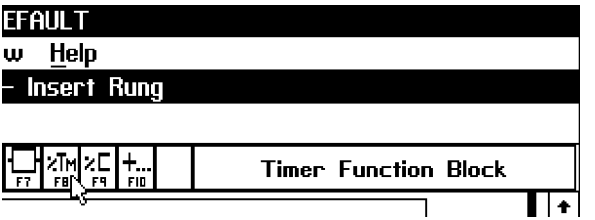
The horizontal connector is inserted automatically and the coil is inserted in the last cell.

4. To complete the programming of the coil, insert an operand or symbol as described in Section 7.6.

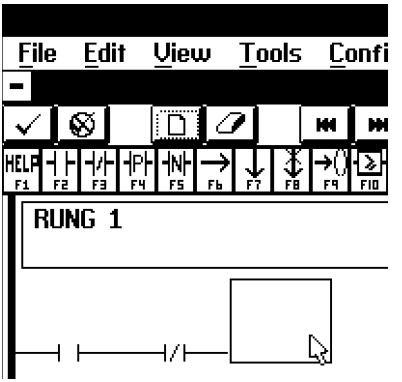
7.5-3 Inserting a timer or counter block

A timer or counter block cannot be inserted in the first column or the last two columns of the grid.

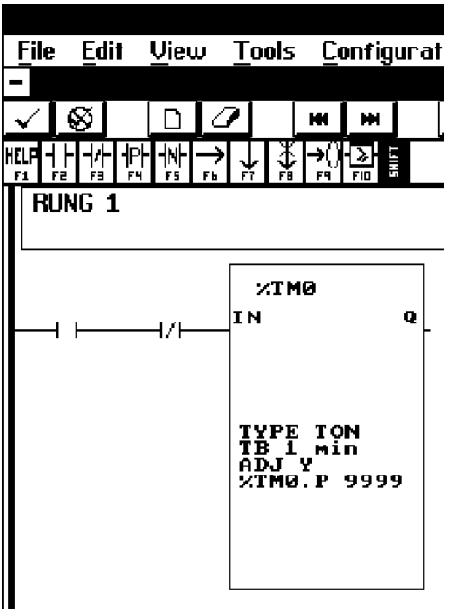
1. Select the timer or counter block from the instruction bar or by pressing the **<shift>** key plus the corresponding function key.



2. Place the contact by clicking the right mouse button in a cell. Or, select a cell with the cursor keys and press the **<spacebar>**.



The timer or counter block is inserted

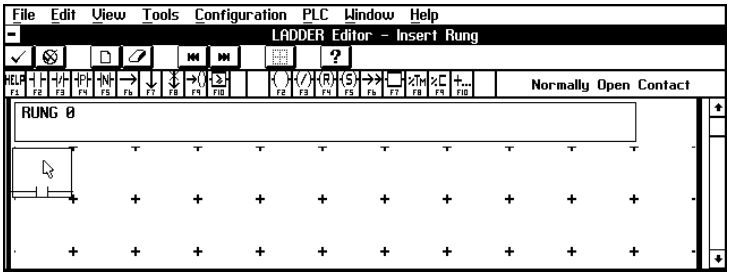


3. To complete the programming of the timer or counter block, configure the block as described in Section 5.

7.5-4 Inserting and removing a down connector

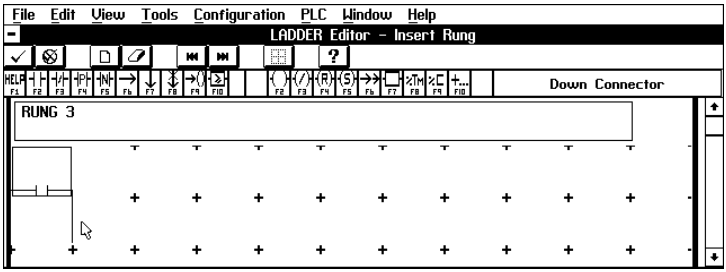
To insert a down connector :

1. Insert the desired instruction, such as a normally open contact, in the rung.



2. Select the down connector from the instruction bar or by pressing **F7**.

3. The down connector appears on the right side of the cell just below the cell selected. Point to the highlighted cell and click the right mouse button. Click the right mouse button again if you want to remove the down connector.

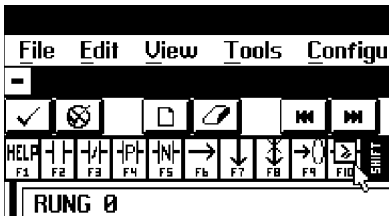


4. To remove a down connector after it has already been placed, select erase down connector from the instruction bar or by pressing **F8**.
5. Point to the cell above and to the left of the down connector and click the right mouse button.

7.5-5 Inserting a comparison block

A comparison block cannot be inserted in the last two columns of the grid.

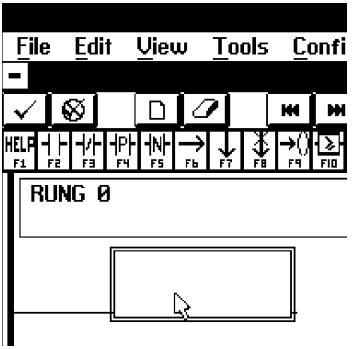
1. Select the comparison block from the instruction bar or by pressing **F10**.



2. Place the comparison block by clicking the right mouse button in the cell. Or, select the cell with the cursor keys and press the **<spacebar>**.

   The comparison block is inserted.

3. To complete the programming of the comparison block, insert an operation string as described in Section 7.6.

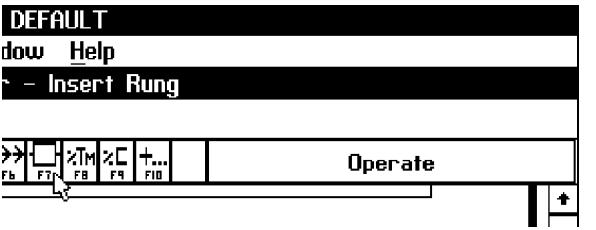


C

7.5-6 Inserting an operate block

An operate block is used for numerical instructions. An operate block can only be inserted in the last four columns of the grid. If you try to insert this instruction anywhere else, a horizontal connector line is automatically inserted from that point to the last four columns, where the block is inserted.

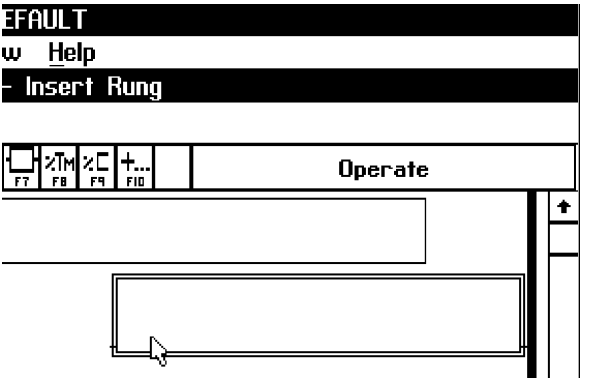
1. Select the operate block from the instruction bar or by pressing **<shift>** + **F7**.



2. Place the operate block by clicking the right mouse button in the cell. Or, select the cell with the cursor keys and press the **<spacebar>**.

   The operate block is inserted.

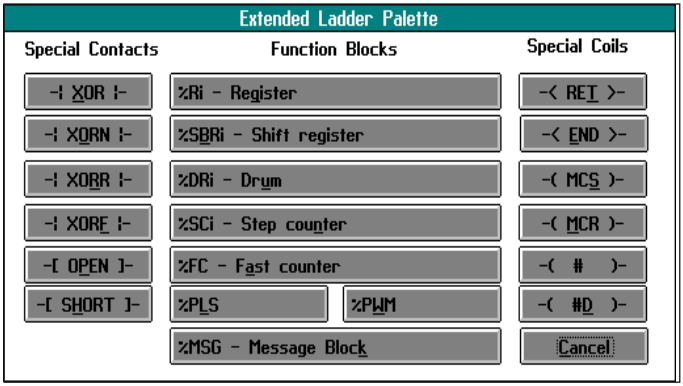
3. To complete the programming of the operate block, insert an operation string as described in Section 7.6.



7.5-7 Inserting special instructions from the Extended Ladder Palette

The Extended Ladder Palette contains special contacts, function blocks, and special coils. To insert an instruction from the Extended Ladder Palette :

1. Select the Extended Ladder Palette (+... instruction) from the instruction bar or by pressing **<shift>** + **F10**. The Extended Ladder Palette appears.



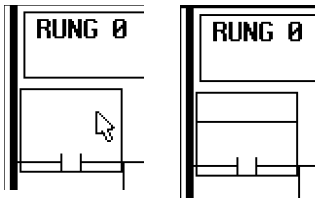
2. Select the desired instruction. The Extended Ladder Palette closes and the selected instruction's name appears on the right side of the instruction bar.
3. Place the instruction in the desired location.

7.6 Inserting an operand or symbol

Simple Ladder instructions, such as contacts and coils, use a single operand. However, some instructions, such as compare and operate blocks, require multiple operands with operators or option calls, called operation strings. For example, "%MW50 := %MW3 + %KW5" in an operate block or "%MW15<0" in a compare block are operation strings. You enter operation strings directly from the keyboard, just as you do when you specify a single operand.

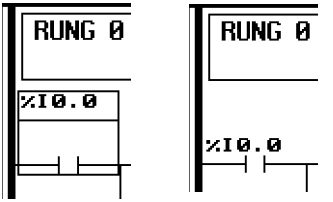
To insert an operand or a symbol above a contact or coil instruction :

1. Select the cell and double-click with the mouse pointer or use the arrow keys to select the cell and press **<enter>**. A rectangular box appears above the instruction.



Note : The % character appears automatically when Show Addresses is selected in the Ladder Viewer Tools menu

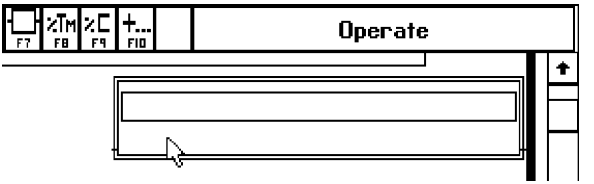
2. Type the operand or symbol in the rectangular box. Press **<enter>**. The rectangular box closes and the operand or symbol appears above the instruction.



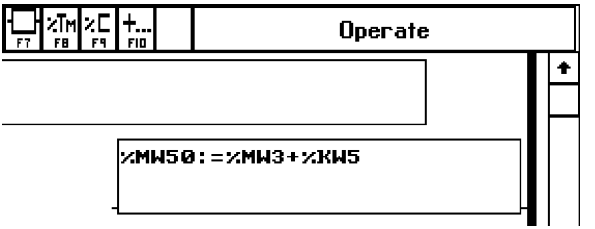
C

To insert an operation string in a comparison or operate block :

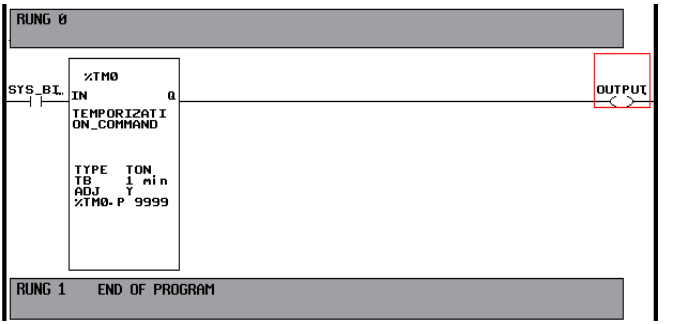
1. Select the comparison or operate block and double-click with the mouse pointer or use the arrow keys to select the cell and press **<enter>**. A rectangular box appears in the block.



2. Type the operation string in the rectangular box. Press **<enter>**. The rectangular box closes and the operand appears in the comparison or operate block.

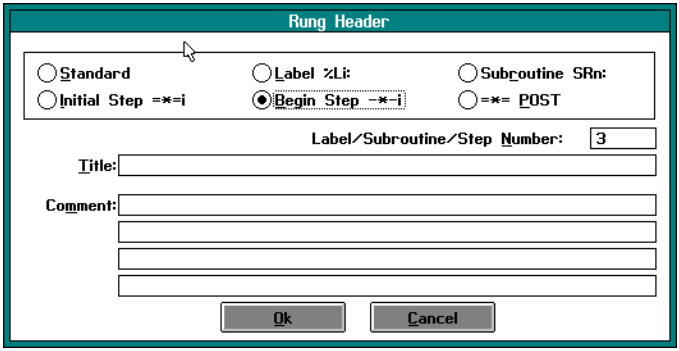


Note :
When it is not possible to see the online instruction or symbol in the rung, the whole instruction or symbol is displayed in the status bar at the bottom of the editor (Ladder Viewer).



7.7 Inserting a rung title, label, or comments

1. Select a rung header by double-clicking with the mouse pointer or use the arrow keys to select the rung header and press **<enter>**. The Rung Header dialog box appears.



2. In the Rung Header Type field, select one of three options :

   Select **Standard** to display only the rung number, the title, and comments.

   Select **Label** to designate the rung as a jump destination. A label (%Li:) is used with a jump instruction as a jump destination. The label is displayed beneath the rung number in the header. See Section 2.4-3 in part B for more information about jump instructions.

   Select **Subroutine** to designate the rung as a call destination. The subroutine label (SRn:) is used in a program as a call destination. See Section 2.4-3 in part B for more information about subroutine instructions.

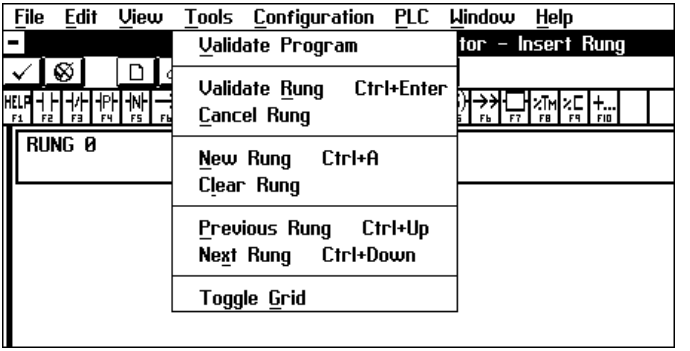
3. If either the Label or Subroutine header type is selected, the **Label or Subroutine Number** field becomes active.

   In the **Label or Subroutine Number** field, declare the identifying number for the label or subroutine. For example, to declare subroutine number 7, select the Subroutine header type and enter **7** in the Label or Subroutine Number field. If you enter a value that is already used elsewhere in the program, an error message is displayed when you press Ok.

4. In the **Title** field, enter up to 122 characters to describe the purpose of the rung.
5. In the **Comment** field, enter up to 4 lines of 122 characters of text to further document the purpose of the rung.
6. Select **Ok** to close the Rung Header dialog box and update the Ladder Viewer. Select **Cancel** to close the dialog box, leaving the rung header unchanged.

7.8 Using the Ladder Editor Tools menu

The Ladder Editor window is used for inserting, creating, and editing rungs in a ladder program. The table shown in appendix A.2-4 in part G lists the options on the Ladder Editor Tools menu and the corresponding buttons on the tool bar.



7.8-1 Validate Program

Use Validate Program to compile a program and check for errors. See Section 10.1 for more details on validating a program.

7.8-2 Validate Rung

Use Validate Rung to validate an individual rung from the Ladder Editor window.

1. Select **Validate Rung** from the Ladder Editor Tools menu or the Ladder Editor tool bar.
2. If the rung does not have any errors, the Ladder Editor window closes and the Ladder Viewer window is updated with the validated rung.
3. If the rung has errors, an error message appears describing the error.amme.

7.8-3 Cancel Rung

Use Cancel Rung to exit the Ladder Editor window and return to the Ladder Viewer window, without adding any changes to the current rung.amme.

7.8-4 New Rung

Use New Rung to validate and store the current rung in the Ladder program and start a new rung.

1. Select **New Rung** from the Ladder Editor tools menu or the Ladder Editor tool bar.
2. The Ladder Editor window is cleared for the insertion of a new rung. The rung number will be the next sequential number in the Ladder program. The Ladder Viewer window displays the validated rung updated from the Ladder Editor window.

7.8-5 Clear Rung

Use Clear Rung to clear the current rung in the Ladder Editor window. The Ladder Editor window remains open with an empty programming grid.

1. Select **Clear Rung** from the Ladder Editor tools menu or the Ladder Editor tool bar.
2. The Ladder Editor window displays the current rung cleared of all previous instructions.

7.8-6 Previous Rung

Use Previous Rung to validate and store the current rung and select the preceding rung.

1. Select **Previous Rung** from the Ladder Editor tools menu or Ladder Editor tool bar.
2. The current rung is validated and stored to the Ladder program. The Ladder Editor window then displays the rung preceding the current rung.

7.8-7 Next Rung

Use Next Rung to validate and store the current rung and select the following rung in the Ladder program.

1. Select **Next Rung** from the Ladder Editor tools menu or the Ladder Editor tool bar.
2. The current rung is validated and stored to the Ladder program. The Ladder Editor window then displays the following rung.

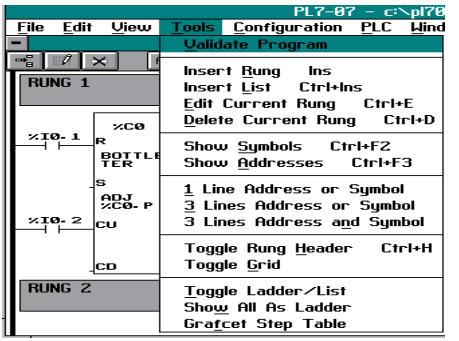
7.8-8 Toggle Grid

Use Toggle Grid to switch from having the grid displayed to having it not displayed in the Ladder Editor window.

1. Select **Toggle Grid** from the Ladder Editor Tools menu or from the Ladder Editor tool bar.
2. The Ladder Editor window is displayed without the grid. Select **Toggle Grid** again to display the grid.

7.9 Using the Ladder Viewer Tools menu

The Ladder Viewer window is used to display and view a program. The Tools menu for this window has a few editing options and many display options. The table in appendix A.2-5 in part G lists the options on the Ladder Viewer Tools menu and the corresponding buttons on the tool bar.



7.9-1 Validate Program

Use Validate Program to compile a program and check for errors. See Section 10.1 for more details on validating a program.

7.9-2 Insert Rung

Use Insert Rung to insert a new rung just before the rung selected in the Ladder Viewer window.

To insert a rung :

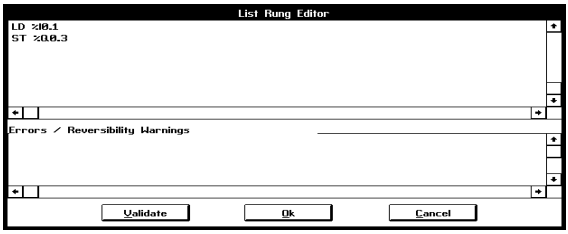
1. Select the location where the new rung will be inserted. Select **Insert Rung** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar. The Ladder Editor window appears above the Ladder Viewer window.

7.9-3 Insert List

Use Insert List to insert a new rung just before the rung selected using the List Rung Editor.

1. Select the location where the new rung will be inserted. Select **Insert List** from the Ladder Viewer Tools menu. The List Rung Editor window appears.
2. After entering the desired instructions in the List Rung Editor window, select **Validate** to check the new rungs for errors and reversibility warnings. Select **Ok** to return to the Ladder Viewer window with the new rung shown in the Ladder format. Select **Cancel** to return to the Ladder Viewer window without adding the new rung.

   If the rung inserted using the List editor is not reversible, the new rung in the Ladder Viewer window displays the List instructions instead of the Ladder format. Reversibility rules are explained in Sections 7.4 and 7.5 of part B.



7.9-4 Edit Current Rung

Use Edit Current Rung to make changes to the selected rung.

1. Select the rung to be edited. Select **Edit Current Rung** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar. The selected rung is displayed in the Ladder Editor window. It is also possible to open the Ladder Editor window by double-clicking on the rung with the left-hand mouse button.

C

7.9-5 Delete Current Rung

Use Delete Current Rung to remove the selected rung or block of rungs (highlighted) from the Ladder program. This includes the rung, rung header, and any label or subroutine declarations.

1. Select the rung to be deleted. Select **Delete Current Rung** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The Ladder Viewer window will update with the selected rung deleted.

7.9-6 Show Symbols

Use Show Symbols to display the symbols for instructions in the Ladder program. Either one or three lines of symbols are displayed, depending upon the option selected as described in Sections 7.9-8, 7.9-9, or 7.9-10.

1. Select **Show Symbols** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The symbols will be displayed above the instructions in either a one-line or three-line format.

7.9-7 Show Addresses

Use Show Addresses to display the address for instructions in the Ladder program. Either one or three lines of addresses are displayed, depending upon the option selected as described in Sections 7.9-8, 7.9-9, or 7.9-10.

1. Select **Show Addresses** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The address will be displayed above the instructions in either a one-line or three-line format.

7.9-8 1 Line Address or Symbol

Use 1 Line Address or Symbol to display one line of either the address or symbol, depending on which one is selected as described in Sections 7.9-6 and 7.9-7.

1. Select **1 Line Address or Symbol** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The addresses or symbols will be displayed above the instructions in a one-line format.

7.9-9 3 Lines Address or Symbol

C

Use 3 Lines Address or Symbol to display three lines of either the address or symbol, depending on which one is selected as described in Sections 7.9-6 and 7.9-7. This option affects coils and contacts only. List rungs, compare blocks, and operate blocks will show only one line; either Symbols or Addresses depending on which one is selected in the Display Attributes field of the Preferences dialog box.

1. Select **3 Lines Address or Symbol** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The addresses or symbols will be displayed above the instructions in a three-line format.

7.9-10 3 Lines Address and Symbol

Use 3 Lines Address and Symbol to display three lines of both the address and symbol. This option affects coils and contacts only. List rungs, compare blocks, and operate blocks will show only one line; either Symbols or Addresses depending on which one is selected in the Display Attributes field of the Preferences dialog box.

1. Select **3 Lines Address and Symbol** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The addresses and symbols will be displayed above the instructions in a three-line format. The symbols will be listed above the addresses.

7.9-11 Toggle Rung Header

Use Toggle Rung Header to switch from having the rung headers displayed to having them not displayed.

1. Select **Toggle Rung Header** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.
2. The Ladder Viewer window will be displayed without the rung headers. Select **Toggle Rung Header** again to display the rung headers.

7.9-12 Toggle Grid

Use Toggle Grid to switch from having the grid displayed to having it not displayed.

1. Select **Toggle Grid** from the Ladder Viewer Tools menu or from the Ladder Viewer tool bar.

2. The Ladder Viewer window will be displayed without the grid. Select **Toggle Grid** again to display the grid.

7.9-13 Toggle Ladder/List

Use Toggle Ladder/List to switch between the Ladder format and the List format for the selected rung or rungs. To toggle the entire program between the two formats, select List Editor or Ladder Editor from the **View** menu.

If a rung is being toggled to go from the List format to the Ladder format and the reversibility rules are not met, the rung will stay in the List format. See part B for an explanation of the reversibility rules.

1. Select the rung to be toggled. Select **Toggle Ladder/List** from the Ladder Viewer Tools menu.
2. The selected rung will change from the Ladder format to the List format. For the selected rung, select **Toggle Ladder/List** again to return to the Ladder format.

7.9-14 Show All As Ladder

Use Show All As Ladder to display the entire program in the ladder format. This would be used if some of the rungs had been toggled to the List format and you wanted to switch the entire program back to the Ladder format without proceeding one rung at a time.

If a rung is in the List format and the reversibility rules are not met, that rung will stay in the List format. See part B for an explanation of the reversibility rules.

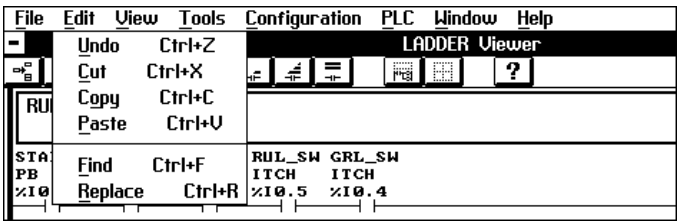
1. Select **Show All As Ladder** from the Ladder Viewer Tools menu.
2. The Ladder Viewer window appears with all reversible rungs in the Ladder format. All non-reversible rungs will stay in the List format.

7.9-15 Table of Grafcet steps (see Section 9.3 Part C)

7.10 Using the Ladder Viewer Edit menur

7.10-1 Introduction

In the Ladder Viewer window, you can edit a program using the options in the Edit menu.



A table in appendix A.2-8 in part G lists the Edit menu options and how they are selected.

7.10-2 Marking a block

To use the Cut or Copy options, you need first to "mark" or highlight the rungs that you want to cut or copy. You cannot mark a portion of a rung; you must mark the entire rung.

To mark a Ladder block :

1. Position the cursor at the beginning of the first rung you want to mark.
2. Hold down the **<shift>** key.
3. Use the up or down arrow keys to move to the end of the last rung you want to mark.
4. Release the **<shift>** key. The highlighted area is the marked block.

7.10-3 Undo

Use Undo to reverse the last Cut, Paste, or Delete operation performed. For example, use Undo to restore a block of rungs that you cut, or to delete a block of rungs that you pasted from the clipboard.

From the Ladder Viewer window :

1. Select **Undo** from the Edit menu.
2. The last editing operation will be reversed.

7.10-4 Cut

Use Cut to move rungs from one place to another within the same program or from one program to another. Cut can be used in the offline or online stopped states.

The clipboard is an internal memory buffer that stores blocks of lines that you cut or copy. If you close one program file and open another, the contents of the clipboard are retained. The contents of the clipboard are overwritten if you cut or copy another block of rungs.

From the Ladder Viewer window :

1. Mark the block of rungs you want to cut as described in Section 7.10-2.
2. Select **Cut** from the Edit menu. The marked block is deleted from the Ladder program, but retained in the clipboard.

If you want to delete the marked block without copying it to the clipboard, press **<delete>**. Use **<delete>** when you want to remove rungs that you do not intend to use elsewhere in the program. When you use **<delete>**, the contents of the clipboard remain the same. Additionally, the current rung in a program can be removed by pressing **<delete>**. The rung does not have to be marked for this to occur.

When any program rung is deleted, the Ladder Viewer window closes the rung and automatically re-numbers the remaining program rungs.

7.10-5 Copy

Use Copy to duplicate a marked block from the program to the clipboard. Copy does not remove the block of marked rungs. Copy is used with the Paste option to duplicate rungs within the same program or to copy a block of rungs from one program to another. Copy can be used in the offline or online stopped states.

From the Ladder Viewer window :

1. Mark the block of rungs you want to copy as described in Section 7.10-2.
2. Select **Copy** from the Edit menu. The marked block is copied to the clipboard for a future Paste operation.

7.10-6 Paste

Use Paste to insert rungs cut or copied to the clipboard at a new location in the program or in a different program. The Paste option does not change the contents of the clipboard. Paste can be used in the offline or online stopped states.

From the Ladder Viewer window :

1. Cut or copy a marked block as described in Sections 7.10-2, 7.10-4, and 7.10-5.
2. Select the rung where you want to insert the marked block.
3. Select **Paste** from the Edit menu. The marked block is inserted before the rung selected in the Ladder program.

   Paste can be used to copy rungs from one part of a program into another part of a program. It can also be used to copy lines from a source program to a completely separate target program.

7.10-7 Find

Use Find to locate each occurrence of an operand, rung, or comment string in a Ladder program. The Find option can be used in the offline, online stopped, or online running states.

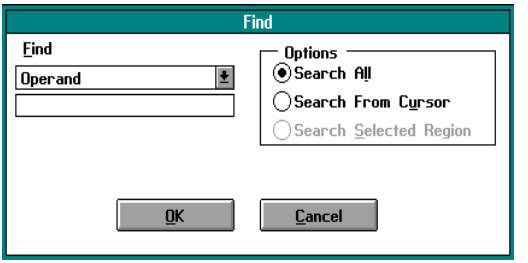
For operands :

- There is no implicit inheritance; that is, if function block %TM0 is to be found, only function block %TM0 will be found. 'AND %TM0.Q' will not be found.
- Subroutines SRn: and labels %Li: are treated as operands.

For Comment Strings, the search operates on comments, operands, operators, labels, and subroutines.

From the Ladder Viewer window :

1. If you want only to search a specified portion of the Ladder program, mark the block of rungs to be searched.
2. Select **Find** from the Edit menu. The Find dialog box appears.



C

3. In the **Find** field, select Operand, Rung, or Comment String. **Operand** is used to find an address or symbol in the Ladder program. It does not matter whether addresses or symbols are shown for a search to be conducted for the other. For example, if addresses are currently shown in the Ladder program, you can conduct a search for a symbol.

   Rung is used to locate a specific rung in the Ladder program by rung number. If a rung number is entered that is greater than the last rung number of the program, the last rung appears.

   Comment String is used to locate each occurrence of a specific string of text in the rung headers.

   Range : Operand, Rung, or Comment String

   Default : Operand

Note : If the program was originally written in List and reversed to Ladder, comments may be present in the program that are "hidden"; that is, they are not shown on the Ladder rung header. However, if a search is conducted for a Comment String that includes any of these comments, they will be found in the search.

4. In the blank field under the Find field, enter the value for Operand, or the Rung number to be found. For a Comment String, enter the text to be found.

   If **Operand** is selected in the Find field, enter an address or symbol.

   If **Rung** is selected in the Find field, enter a program rung number.

   If **Comment String** is selected in the Find field, enter a specific string of text.

5. In the **Options** field :

   Select **Search All** to search from the beginning of the program, or in a marked block.

   Select **Search From Cursor** to search from the current position of the cursor up to the end of the program. Do not select Search From Cursor to search in a marked block of rungs.

   Select **Search Selected Region** to search only in a marked block.

6. Select **Ok** to begin a search. Select **Cancel** to return to the Ladder Viewer window.

   For each instance of the value that is found in the Ladder program, the following dialog box is displayed.

7. After the last instance of the value or text is found or if the search does not find any instance of the value or text you specified, an Information dialog box will appear stating "Item not found." Select **Ok** to end the search and return to the Ladder Viewer window.

7.10-8 Replace

Use Replace to locate each occurrence of an operand or comment string and replace it with another operand or comment string. Replace can be used in the offline or online stopped states.

For operands, you can only replace :

- bits by bits (for example, %I0.0 with %M2),
- words by words (for example, %MW100 with %SW12)
- function blocks with like type function blocks (for example, %TM0 with %TM2 is allowed, whereas %TM0 with %C3 is not allowed)
- immediate values with other immediate values.

There is no implicit inheritance; that is, if function block %TM0 is replaced with %TM2, only the function blocks are replaced. %TM0.Q is not replaced with %TM2.Q.

Replace will not work if either the source or target operand is an unresolved symbol.

Replace label and subroutine will not operate on the declaration of the label or subroutine. A label or subroutine declaration may be replaced with another label or subroutine declaration.

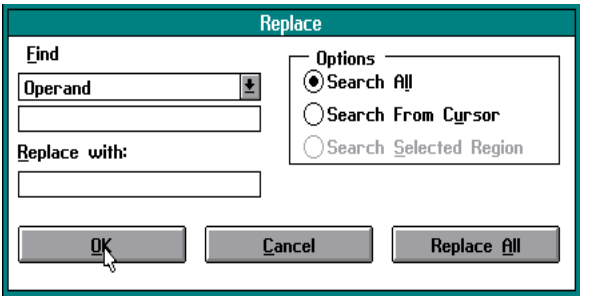
For Comment Strings, anything that can be found by a string in the Ladder can be replaced.

From the Ladder Viewer window :

1. If you want only to find and replace an operand or comment string in a specified portion of the Ladder program, mark the block of rungs to be searched.
2. Select **Replace** from the Edit menu. The Replace dialog box appears.
3. In the **Find** field, select either **Operand** or **Comment String**.

   Operand is used to find and replace an address or symbol in the Ladder program.

   Comment String is used to find and replace a specific string of text in the rung headers.



C

4. In the blank field under the Find field, enter the value for the Operand, or the text for the Comment String, to be found and replaced.

   If **Operand** is selected in the Find field, enter an address or symbol.

   If **Comment String** is selected in the Find field, enter a specific string of text.

5. In the **Replace with**: field, enter the value or text that is to replace the found value or text.

   If **Operand** is selected in the Find field, enter an address or symbol.

   If **Comment String** is selected in the Find field, enter a specific string of text.

6. In the **Options** field :

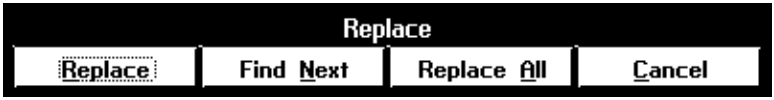
   Select **Search All** to find and replace from the beginning of the program, or in a marked block.

   Select **Search From Cursor** to find and replace from the current position of the cursor up to the end of the program. Do not select Search From Cursor to search in a marked block of rungs.
   Select **Search Selected Region** to find and replace only in a marked block.

7. Select **Replace All** to find each occurrence of the specified value or text and replace with the value or text specified in the Replace with: field. These operations are performed automatically. After all of the occurrences have been found and replaced, an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the Ladder Viewer window.

8. Select **Ok** to begin the find and replace operation. Select **Cancel** to return to the Ladder Viewer window.

   For each instance of the value or text that is found in the Ladder program, the Replace dialog box is displayed.



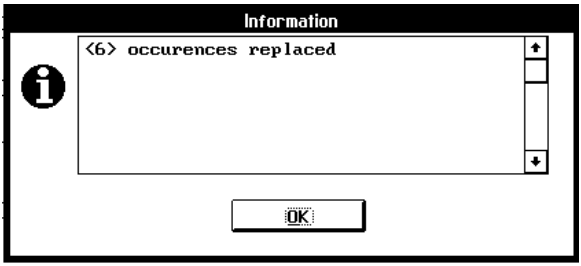
9. In the **Replace** dialog box, select **Replace**, **Find Next**, **Replace All**, or **Cancel**.

   Replace changes the current value or text to the value or text specified in the Replace with: field.

   Find Next makes no changes to the value or text currently highlighted and searches for the next occurrence of the value or text.

   Replace All finds each occurrence of the specified value or text and replaces each occurrence with the value or text specified in the Replace with: field. After all of the occurrences have been found and replaced, an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the Ladder Viewer window.

   Cancel exits the find and replace operation and an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the Ladder Viewer window.



10. After the last occurrence has been found and replaced, an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the Ladder Viewer window.

8.1 Introduction

The List Editor is a simple line editor for writing and editing List programs. The List Editor is selected using the Preferences option from the View menu.

You can use the List Editor in the online or offline state. However, in the online run state, you can only insert, delete, or modify certain instructions and use certain options.

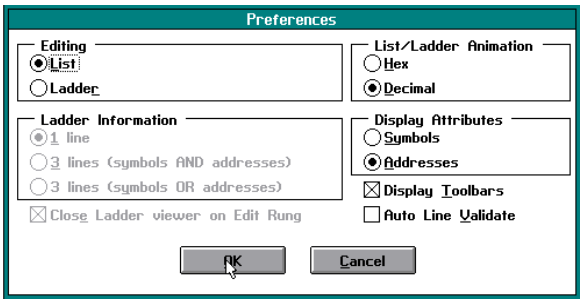
Some instructions require balancing or complementary instructions. For example, a BLK instruction requires an END_BLK instruction. An instruction modified by a parenthesis requires a closing parenthesis—all in the same scan.

To maintain an effective scan rate, the PLC accepts a single instruction line at a time. Consequently, there is no mechanism to write these complex instructions without severely diminishing the performance of the PLC. Therefore, you cannot insert, modify, or delete some instructions while the PLC is in the online state and running.

8.2 Configuring the List Editor

To configure the List Editor:

1. Select **Preferences** from the View menu to display the Preferences dialog box.



2. In the **Editing** field, select **List**. The Ladder Information field is not active when List is selected.
3. After selecting List in the Editing field, the **Auto Line Validate** field becomes active.

   In the online state, program lines are automatically validated as you enter them. Select **Auto Line Validate** to automatically validate program lines as you enter them in the offline state.

   Use Auto Line Validate to help debug a program as it is written. Auto Line Validate does not replace the Validate Program option. Before you transfer a program to the PLC, run Validate Program.

Do not select Auto Line Validate if you prefer to write a program with syntactical errors and unresolved symbols. If Auto Line Validate is selected, errors and unresolved symbols must be corrected before you can exit the line.

Default: Auto Line Validate is not selected.

4. You animate a program to display the current value of a variable in the PLC. In the **List/Ladder Animation** box, select the number format for displaying the current values when the program is animated.

   Range: Hexadecimal or Decimal

   Default: Decimal

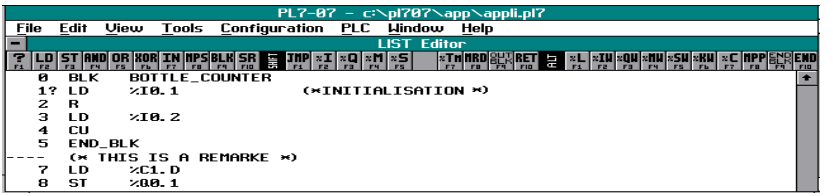
5. In the **Display Attributes** field, select the desired attribute, either Symbols or Addresses, to be displayed with the line number and operator.

   Default: Addresses

6. Select **Display Toolbars** to display the tool bars in all editor windows.
7. Select **Ok** to commit the values you selected. Select **Cancel** to exit the dialog box without changing the preferences selected.

8.3 Using the List editor

1. Select the **List Editor** from the View menu. The List Editor window is displayed.



2. Type instructions directly from the keyboard or select operators and operands from the instruction bar using either the keyboard or a mouse. Maintain a space between the operator and the symbol or between the operator and an address.

   The List instruction bar displays the most commonly used List operators and operands. The instruction bar enhances the speed and accuracy of writing a List program. For more information about the instructions in the bar, see appendix A.2-6 in part G.

3. You can enter a symbol, such as START_SWITCH, rather than an address, such as %I0.7. For more information about using symbols in your program, see section 6, "Defining symbols."

4. Optionally, you can enter a line comment at the end of a program line. A *line comment* is free-format text you enter to document a program. Write the comment between two markers or *comment delimiters*. The comment delimiters are an open parenthesis followed by an asterisk (* before the comment and an asterisk followed by a close parenthesis *) after the comment, as shown below:

   (*THIS IS A COMMENT.*)

5. Maintain at least one space between the operator, operand, and comments, as shown below:

   LD START_SWITCH (*START CONVEYOR*)

6. After you write a line, press <Enter>.

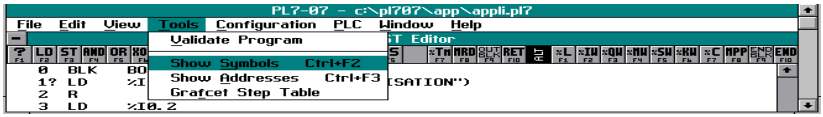
   PL7-07 PC Programming Software automatically reformats the line for you and assigns the completed line a line number. The cursor advances to a new line.

   The number to the left of a program line is called a *program line number*. These numbers help you locate individual program lines when you want to debug problems found during Validate Program. These numbers correspond to the program line numbers in the FTX 117 Hand Held Programmer.

C

8.4 Using the List Editor Tools menu

The List Editor Tools menu is used for validating the program and showing either symbols or addresses.



8.4-1 Validate Program

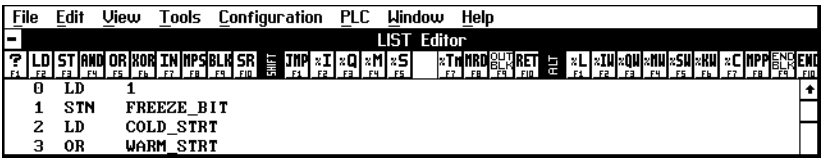
Use Validate Program to compile a program and check for errors. See section 10.1 for more details on validating a program.

8.4-2 Show Symbols

Use Show Symbols to display symbols, instead of addresses, in the List Editor. If an address has a corresponding symbol, the symbol appears in the program. If an address has no corresponding symbol, the address appears.

Use Show Symbols in either the offline or online state. Selecting Show Symbols does not restrict your ability to insert symbols or addresses in any way.

To display symbols in the List Editor, select **Show Symbols** from the Tools menu. Symbols, rather than addresses, are displayed.



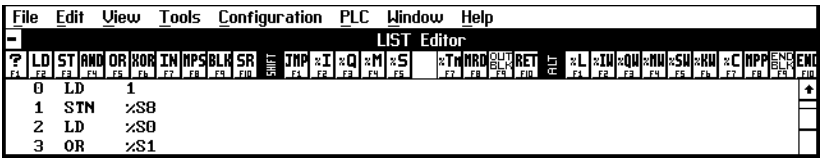
C

8.4-3 Show Addresses

Use Show Addresses to display addresses, instead of symbols, in the List Editor. The address appears in the program whether or not it has a corresponding symbol.

Use Show Addresses in either the offline or online state. Selecting Show Addresses does not restrict your ability to insert addresses or symbols in any way.

To display addresses in the List Editor, select **Show Addresses** from the Tools menu. Addresses, instead of symbols, are displayed.

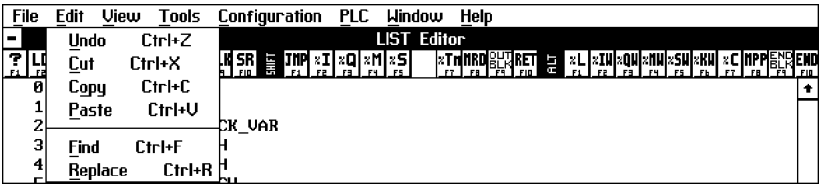


8.4-4 Grafcet Step Table (see Section 9.3 Part C)

8.5 Using the List Editor Edit menu

8.5-1 Introduction

In the List Editor window, you can edit a program using the options in the Edit menu.



A table in appendix A.2-8 in part G lists the Edit menu options and how they are selected.

8.5-2 Marking a block

To use the Cut or Copy options, you need first to mark or highlight the lines that you want to cut or copy. You cannot mark a portion of a line; you must mark the entire line.
To mark a List block:

1. Position the cursor at the beginning of the first line you want to mark.
2. Hold down the **<shift>** key.
3. Use the up or down arrow keys to move to the end of the last line you want to mark.
4. Release the **<shift>** key. The highlighted area is the marked block.

8.5-3 Undo

Use Undo to reverse the last Cut, Paste, or Delete operation performed. For example, use Undo to restore a block of lines that you cut or to delete a block of lines that you pasted from the clipboard.
From the List Editor window:

1. Select **Undo** from the Edit menu.
2. The last Cut, Paste, or Delete operation will be reversed.

8.5-4 Cut

Use Cut to move lines from one place to another within the same program or from one program to another. Cut can be used in the offline or online stopped states.

The clipboard is an internal memory buffer that stores blocks of lines that you cut or copy. If you close one program file and open another, the contents of the clipboard are retained. The content of the clipboard is overwritten if you cut or copy another block of lines.
From the List Editor window:

1. Mark the block of lines you want to cut as described in section 8.5-2.
2. Select **Cut** from the Edit menu. The marked block is deleted from the List program, but retained in the clipboard.

If you want to delete the marked block without copying it to the clipboard, press **<delete>**. Use **<delete>** when you want to remove lines that you do not intend to use elsewhere in the program. When you use **<delete>**, the contents of the clipboard remain the same. Additionally, the current line in a program can be removed by pressing **<delete>**. The line does not have to be marked for this to occur.

When any program line is deleted, the List Editor window closes the line and automatically re-numbers the remaining program lines.

8.5-5 Copy

Use Copy to duplicate a marked block from the program to the clipboard. Copy does not remove the block of marked lines. Copy is used with the Paste option to duplicate lines within the same program or to copy a block of lines from one program to another. Copy can be used in the offline or online stopped states.
From the List Editor window:
1. Mark the block of lines you want to copy as described in section 8.5-2.
2. Select **Copy** from the Edit menu. The marked block is copied to the clipboard for a future Paste operation.

C

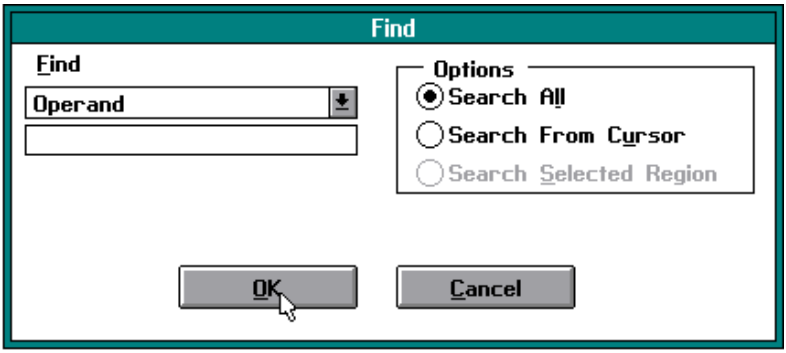
8.5-6 Paste

Use Paste to insert lines cut or copied to the clipboard at a new location in the program or in a different program. The Paste option does not change the contents of the clipboard. Paste can be used in the offline or online stopped states.
From the List Editor window:
1. Cut or copy a marked block as described in sections 8.5-2, 8.5-4, and 8.5-5.
2. Select the line where you want to insert the marked block.
3. Select **Paste** from the Edit menu. The marked block is inserted into the List program.

   Paste can be used to copy lines from one part of a program into another part of a program. It can also be used to copy lines from a source program to a completely separate target program.

8.5-7 Find

Use Find to locate each occurrence of an operand, line, or string in a List program. The Find option can be used in the offline, online stopped, or online running states.
For operands:
• It is not necessary to qualify the operand by a specific instruction (for example, find %M1 in all occurrences of LD).
• There is no implicit inheritance, that is, if function block %TM0 is to be found, only function block %TM0 will be found. 'AND %TM0.Q' will not be found.
• Subroutines SRn: and labels %Li: are treated as operands.
For Text Strings, the search operates on comments, operands, operators, labels, and subroutines.
From the List Editor window:
1. If it is desired to only search a specified portion of the List program, mark the block of lines to be searched.
2. Select **Find** from the Edit menu. The Find dialog box appears.



3. In the **Find** field, select **Operand**, **Line**, or **Text String**.

   Use **Operand** to find an address or symbol in the List program. It does not matter whether addresses or symbols are shown for a search to be conducted for the other. For example, if addresses are currently shown in the List program, you can conduct a search for a symbol.
   Use **Line** to locate a specific line in the List program by line number.
   Use **Text String** to locate each occurrence of a specific string of text in the program lines.
   Range: Operand, Line, or Text String
   Default: Operand

4. In the blank field under the Find field, enter the value for the Operand or Line number to be found. For a Text String, enter the text to be found.
   If **Operand** is selected in the Find field, enter an address or symbol.
   If **Line** is selected in the Find field, enter a program line number.
   If **Text String** is selected in the Find field, enter a specific string of text.

5. In the **Options** field:

   Select **Search All** to search from the beginning of the program, or in a marked block.

   Select **Search From Cursor** to search from the current position of the cursor up to the end of the program. Do not select Search From Cursor to search in a marked block of rungs.
   Select **Search Selected Region** to search only in a marked block.

6. Select **Ok** to begin a search. Select **Cancel** to return to the List Editor window.

   For each instance of the value that is found in the List program, the following dialog box appears.

7. After the last instance of the value or text is found or if the search does not find any instance of the value or text you specified, an Information dialog box appears stating "Item not found". Select **Ok** to end the search and return to the List Editor window.

C

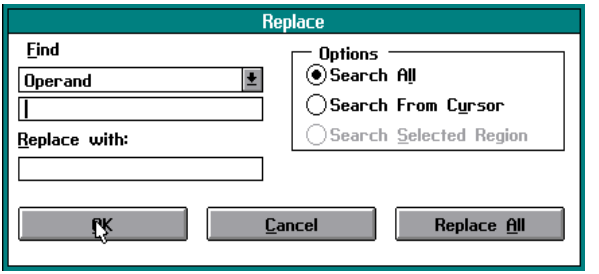
8.5-8 Replace

Use Replace to locate each occurrence of an operand or text string and replace it with another operand or text string. Replace can be used in the offline or online stopped states.

For operands, you can only replace:
- bits by bits (i.e. %I0.0 with %M2),
- words by words (i.e. %MW100 with %SW12)
- function blocks with like type function blocks (i.e. %TM0 with %TM2 is allowed, whereas %TM0 with %C3 is not allowed)
- immediate values with other immediate values

There is no implicit inheritance; that is, if function block %TM0 is replaced with %TM2, only the function blocks are replaced. %TM0.Q is not replaced with %TM2.Q.
Replace will not work if either the source or target operand is an unresolved symbol. Replace label and subroutine will not operate on the declaration of the label or subroutine. A label or subroutine declaration may be replaced with another label or subroutine declaration.
For Text Strings, anything that can be found by a string in the List can be replaced.

From the List Editor window:
1. If it is desired to only find and replace an operand or text string in a specified portion of the List program, mark the block of lines to be searched.
2. Select **Replace** from the Edit menu. The Replace dialog box appears.



3. In the **Find** field, select either **Operand** or **Text String**.

   Operand is used to find and replace an address or symbol in the List program. Use **Text String** to find and replace a specific string of text in the line headers.
   Range: Operand or Text String
   Default: Operand

4. In the blank field under the Find field, enter the value for the Operand or the text for the Text String to be found and replaced.
   If **Operand** is selected in the Find field, enter an address or symbol.
   If **Text String** is selected in the Find field, enter a specific string of text.

5. In the **Replace with:** field, enter the value or text that is to replace the found value or text.
   If **Operand** is selected in the Find field, enter an address or symbol.
   If **Text String** is selected in the Find field, enter a specific string of text.

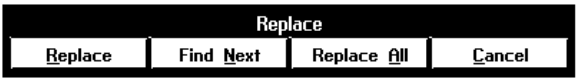
6. In the **Options** field:

   Select **Search All** to search from the beginning of the program, or in a marked block.

   Select **Search From Cursor** to search from the current position of the cursor up to the end of the program. Do not select Search From Cursor to search in a marked block of rungs.
   Select **Search Selected Region** to search only in a marked block.

7. Select **Replace All** to find each occurrence of the specified value or text and replace with the value or text specified in the Replace with: field without confirming each instance. After all of the occurrences have been found and replaced, an Information dialog box is displayed stating the number of occurrences replaced. Select **Ok** to return to the List Editor window.

8. Select **Ok** to begin the find and replace option. Select **Cancel** to return to the List Editor window.
   For each instance of the value or text that is found in the List program, the Replace dialog box appears.



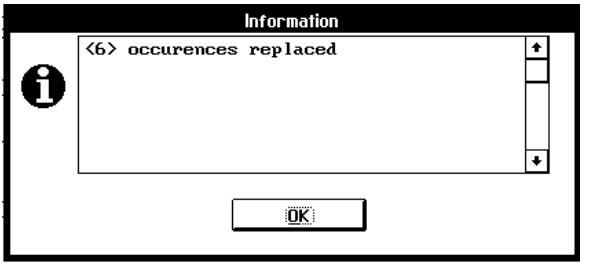
9. In the **Replace** dialog box, select **Replace**, **Find Next**, **Replace All**, or **Cancel**.

   Replace changes the current value or text to the value or text specified in the Replace with: field.

   Find Next makes no changes to the value or text currently highlighted and searches for the next occurrence of the value or text.

   Replace All finds each occurrence of the specified value or text and replaces each occurrence with the value or text specified in the 'Replace with:' field. After all of the occurrences have been found and replaced, an Information dialog box is displayed stating the number of occurrences replaced. Select **Ok** to return to the List Editor window.

   Cancel exits the find and replace operation and an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the List Editor window.



10. After the last occurrence has been found and replaced, an Information dialog box appears stating the number of occurrences replaced. Select **Ok** to return to the List Editor window.

9.1 Introduction

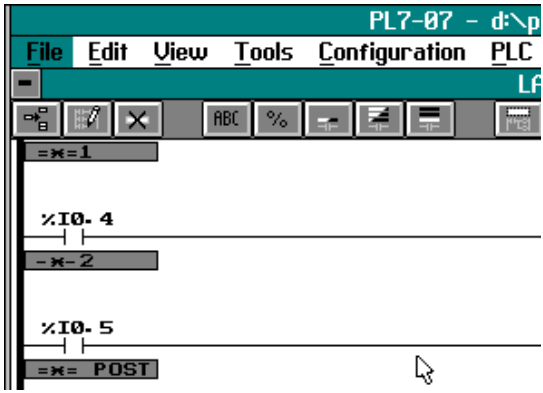
This section describes the Grafcet help functions available from Ladder.

9.2 Showing Grafcet instructions

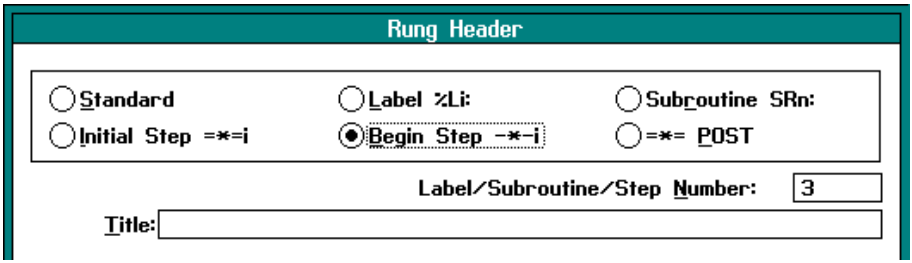
Grafcet instructions can be divided into two types.

Group 1	Group 2
= * = i	#
_ * _ i	# i
= * = POST	# Di

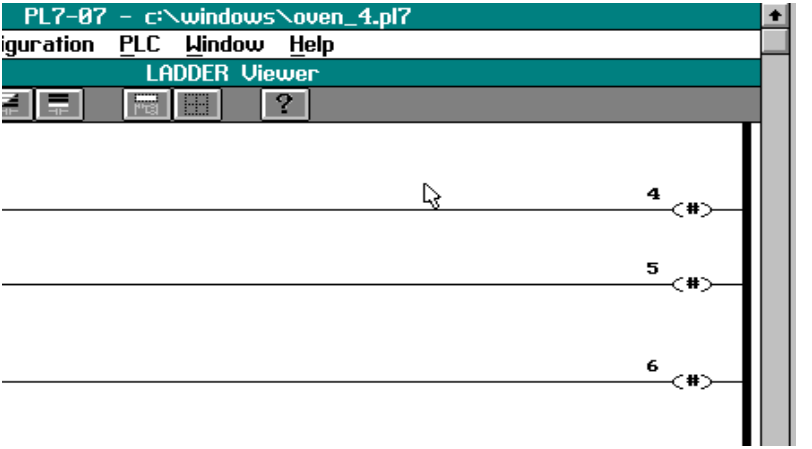
Type 1 instructions are treated as labels or subroutines. Declarations of steps or post-processing are shown continuously in the rung headers.



These instructions are available in the rung header dialog box. When incorrect data is entered (value or declaration of an existing step), an error message is displayed in the status bar after clicking on OK.

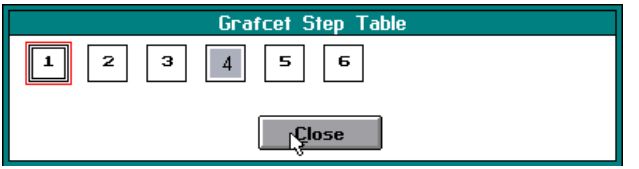


Type 2 instructions are shown as reversible instructions having the same properties as coils (S, ST ...). These Grafcet objects are available on the Extended Ladder Editor Palette.



9.3 Grafcet Step Table

In order to facilitate displaying the Grafcet status, a table of Grafcet steps is available from the Tools menu in the List or Ladder editors.



This table shows the list of steps defined in the program in ascending order of steps. The initial step is represented by a double square (example: step 1).

If the software is online, the window is animated and the active step or steps are highlighted.

It is possible to move through the various steps using the keyboard (keys and ¬) or the mouse. The selected step is outlined in red. By double-clicking on this step or pressing <Enter>, the part of the program associated with this step can be directly accessed in the List or Ladder editor.

10.1 Validating a program

Validate Program compiles a program and checks it for errors. *Compiling* a program means to translate a program into binary machine code, a low-level language which can be executed by the PLC.

In addition to compiling a program, Validate Program also :

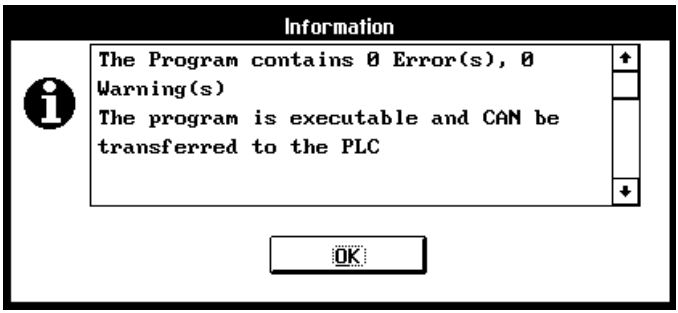
- checks that the syntax of each program line or rung is correct.
- checks that each symbol used in a program has a corresponding address.
- checks that the structure of the program is correct.
- creates a binary program file.
- posts messages in the Validation Errors window.

You can validate a program, in the offline or online states, using these options :

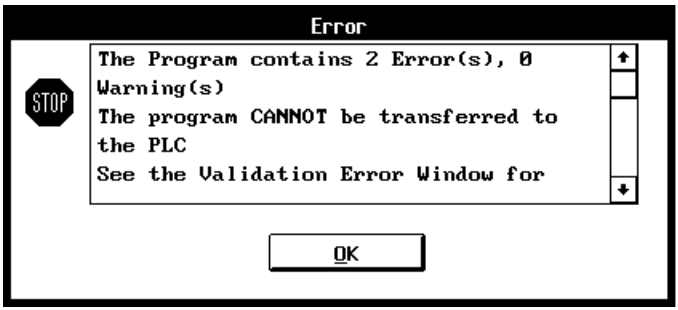
1. Validate Program (Offline)—In the offline state, select Validate Program from the Tools menu to check and compile a program at any time, from any editor.
2. Validate Program (Online)—In the online state, each program line you enter is automatically validated, before it is sent to the PLC. Therefore, you do not need to run Validate Program, although you can choose to run it.

After you run Validate Program, one of two messages is displayed.

If the program has no errors, the following appears :



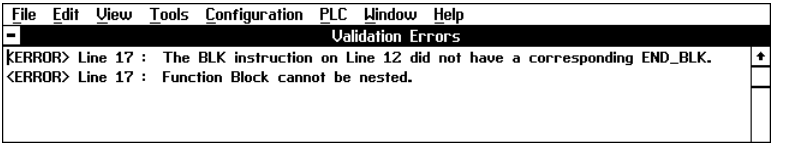
If the Validate Program process detects an error, the following appears :



10.2 Viewing validation errors

Validation Errors displays error and warning messages posted by the Validate Program process.

1. Select **Validation Errors** from the View menu to display the Validation Errors window.



C

The format of each message is :

- type of message—*error* or *warning.*
- the number of the offending line or rung.
- an explanation of the problem.

There are two types of messages—*error* and *warning* messages. An *error* message indicates a problem in the application that prevents the creation of an executable program. A *warning* message indicates instructions that are not reversible, or instructions that, in relation to other instructions, may cause operational uncertainties.

10.3 Reversing a program

An application program written in Ladder can be converted - or reversed - to List. Likewise, a program written in List can be reversed to Ladder if certain reversibility rules are observed. These rules are further explained in Sections 7.3 and 7.4 of Part B.

Note :
In the case where a program written in List is reversed to Ladder and the rules are not met, the corresponding rungs in the Ladder Editor window will display the List instructions instead of the Ladder format.

To reverse from the List Editor to the Ladder Editor :

- Select **Ladder Editor** in the View menu
- Adjust the display by selecting **Preferences** in the View menu

To return from Ladder Editor to List Editor :

- Simply select **List Editor** in the View menu.

11.1 Introduction

An application file is archived by selecting either Save or Save As from the File menu. A binary file *.APP is archived by selecting Save As from the File menu.

- Save is used to commit changes to an existing application.
- Save As is used to save the current application or binary file to a new file.

11.2 Save

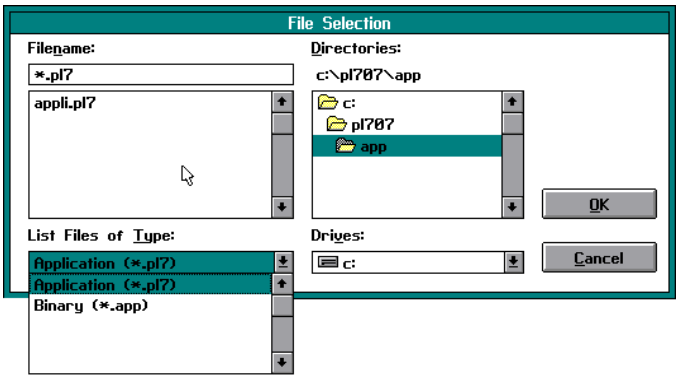
To commit changes to an existing application file:

1. Select **Save** from the File menu. The file will be saved to the current directory.

11.3 Save As

To save the current application or binary file to a new file:

1. Select **Save As** from the File menu. The File Selection dialog box appears.



2. In the **List Files of Type** field, open the selection box and select either the .pl7 or .app file type. The .pl7 file extension identifies application files. The .app file type identifies binary files.

   A binary file (.app extension) is used for transferring a file to the FTX 117 terminal via a PC card (memory card). See appendix A.7 in part G for more information.
3. In the **Drives** field, select the drive where you want to store the file.

4. In the **Directories** field, select the directory where you want to save the file.
5. In the **Filename** field, over type the asterisk (*) with a standard DOS file name.

   If a filename does not comply with DOS file naming conventions, an "Invalid Filename" message appears.

   If you type a filename that already exists in the directory, an error message appears: "The file selected already exists. Overwrite?" Select **Ok** or **Cancel**.
6. Select **Ok** to save the file. Select **Cancel** to close the File Selection dialog box without saving the file.

12.1 Introduction

PL7-07 software provides 2 methods of protecting the application while it is being debugged :

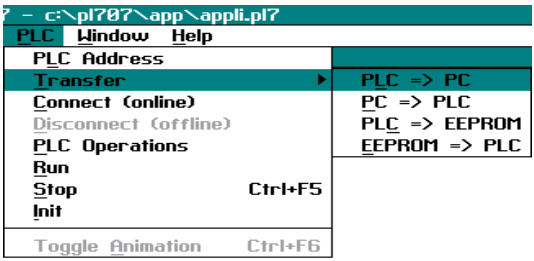
- Total read and write protection.
  This option particularly prohibits duplicating programs thus guaranteeing the inviolability of the programmer's expertise.
  It is executed when the application is transferred into the PLC memory.
- A write protection, with access to the application display and to the variable settings.
  A password provides this protection (see Section G-A.4).

12.2 Transferring an application

Select **Transfer** from the PLC menu to copy an application to one of three hardware storage areas :

1. PC RAM (Random Access Memory).
2. PLC RAM.
3. The PLC EEPROM (Electrically Erasable Programmable Read-Only Memory), a secondary or backup memory storage area on the PLC.

The **Transfer** option in the PLC menu displays four sub-menu options :



12.2-1 PLC => PC

Select **PLC => PC** to transfer the application from the PLC RAM to the PC RAM :

If you begin from the initial state, with no application open :

1. Select **PLC => PC**.

   From the initial state, the transferred application becomes the current application. The word *default* appears in the title bar of the application.

If you begin from the offline state, with an application open :

1. Select **PLC => PC**.

   The PLC application overwrites the program and configuration in the PC, but keeps the symbols.

If the application is password protected, you are prompted to change security levels.

1. Select **Ok** to display the Security dialog box. In the **Enter Password** field, type the correct password. Select **Ok**. The application is displayed at the *supervisor* level.
2. Select **Cancel** to display the application at the *operator* level.

For more information about software Security features, see appendix A.4 in part G.

C

12.2-2 PC => PLC

Select **PC => PLC** to copy the current application on the PC to the PLC.

To transfer the application from the PC RAM to the PLC RAM :

1. Open the application file (.pl7) or binary file (.app) that you want to transfer.
2. Select **PC => PLC**.
3. If the application and PLC versions are not the same, the message, "The application and PLC are different versions" appears.

   Select **Ok** to continue transfer.

   Select **Cancel** to change the version of the PLC. For more information, see Section 5.21.
4. If the application in the PLC contains a password, you are prompted to confirm transfer of the application.

   If you know the password, select **Ok** to transfer over the protected application.

   If you don't know the password, select **Cancel** to end the transfer process.
5. If you select Ok, a Security dialog box appears. Enter the correct password and select **Ok**.
6. If the PLC and PC applications are different, then you are prompted to select whether you want to overwrite the application in the PLC.

   Select **Ok** to overwrite the application.

   Select **Cancel** to end the transfer process. (To back up the PLC application to the PC, see Section 12.2-1.)
7. You are prompted to select whether you want to protect the application.

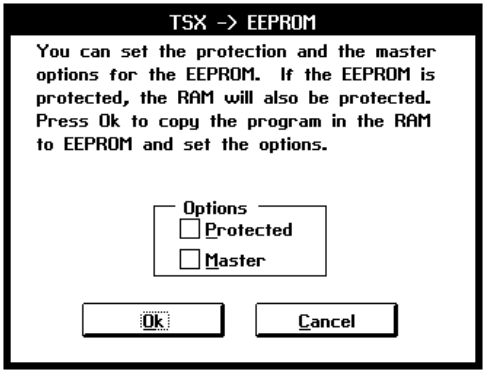
   Select **Yes** to protect the application in the PLC or **No** to leave the application unprotected.
8. When the transfer to the PLC is completed, the message, "Transfer Successful," appears in the status bar.

12.2-3 PLC => EEPROM

Select **PLC => EEPROM** to copy an application from the PLC RAM to the PLC EEPROM. The PLC EEPROM can store one application. It is strongly recommended to transfer an application into the EEPROM as soon as it has been debugged.

To transfer the application from the PLC RAM to the PLC EEPROM :

1. Select **PLC => EEPROM** from the Transfer sub-menu.
2. The PLC => EEPROM dialog box has two options :



The **Protected** option lets you protect the application in the PLC, unless you have already protected it during the PC => PLC transfer process. If protected in the PLC, an error message warns you that you cannot copy or write the application to the EEPROM because neither read or write is allowed.

The **Master** option lets you set the application in the EEPROM as the *master*, which means that each time you power up, the PLC compares the application which is present in RAM with that in EEPROM and, if a difference is detected, the application stored in EEPROM is copied to the PLC RAM, and the PLC is set to RUN (unless an R/S input configured in the application is at STOP).

To modify a master application :

- retrieve the application to the PLC or PC RAM
- modify the application then validate the modifications
- transfer the application back to EEPROM by selecting the Master option.

3. After the application is transferred from the PLC RAM to the EEPROM, the message, "Transfer Successful," appears in the status bar.

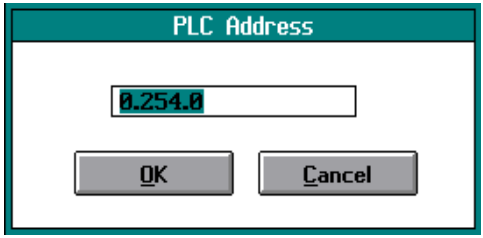
12.2-4 EEPROM => PLC (EEPROM to PLC)

Select **EEPROM => PLC** to transfer the application that you backed up to the EEPROM to the PLC primary storage area.

To transfer an application stored in the EEPROM to the PLC, simply select **PLC => EEPROM** from the Transfer sub-menu. After the application is transferred, the status bar displays the message "Transfer Successful."

13.1 PLC Address

In order to be able to communicate directly with a TSX Nano UNI-TELWAY Slave, PL7 07 software enables the target address to be defined.
After having chosen this address, the user selects the required action in the PLC menu (Transfer, Connect, PLC Operations).



The default address offered is the system address 0.254.0.
The user must then enter the address of the PLC to which he wishes to connect using the following format : Network.Station.Gate.Rack/Module.Slave Address:

- **Network** is the number of the network which enables the target address to be reached (0 to 127), the default value is 0.
- **Station** is the number of the station on the network (0 to 254), the default value is 254.
- **Gate** is the mechanism which enables the unit of communication within the selected station to be chosen. As 0 is the station system gate (its UNITE server), 5 is the gate which enables communication with a remote PLC. The default value is 0 (in this case it is not necessary to complete the other fields).
- **Rack/Module** is used if the gate number is 5. It corresponds to the physical situation of the UNI-TELWAY Master module. The default value is 254 and means that the TSX Nano is connected to the same UNI-TELWAY bus as the PL7 07.
  The value 0 indicates another UNI-TELWAY bus (PCMCIA for example).
- **Slave address** is the address of the TSX Nano PLC on the UNI-TELWAY bus.
  The default value is 4 in local mode and 104 in remote mode.

To help the user, 3 key words are available and can be entered directly :

- **SYS** corresponds to the system address 0.254.0,
- **LOC** corresponds to the local address 0.254.5.254.4. The slave address 4 should be changed if it does not correspond with that of the target PLC.
- **REM** corresponds to the remote address 0.254.5.0.104. The Rack/Module and Slave addresses should correspond to the address of the target TSX Nano.

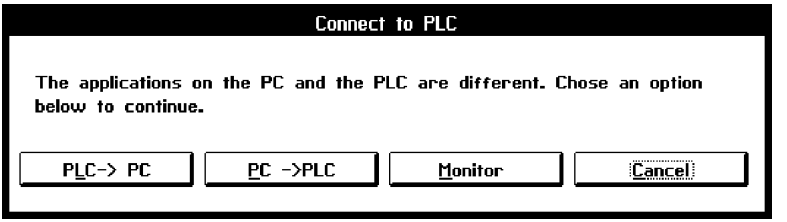
OK validates the values entered.
Cancel exits without changing the address values.

13.2 Connect

Select **Connect** from the PLC menu to initiate communication between the PC and the PLC.

To connect the PC to the PLC :

1. Select **Connect** from the PLC menu.
2. If the applications in the PC and the PLC are the same and the application in the PLC is not protected, the PC connects to the PC and the application state changes from *offline* to *online*.
3. If the application in the PLC is protected, you are prompted to choose whether you want to monitor the PLC. Select **Ok** to monitor the PLC or **Cancel** to end the connection process and return to the *offline* or *initial* state.
4. If the applications in the PC and the PLC are different and the application in the PLC is not protected, the Connect to PLC dialog box appears. Select an option :



Select **PLC -> PC** to transfer the application in the PLC to the PC. The application state changes from *offline* to *online*.

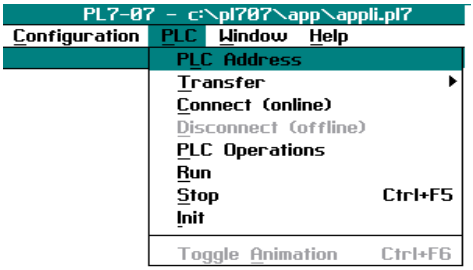
Select **PC -> PLC** to transfer the application open on the PC to the PLC. An Information dialog box appears informing you that you are about to overwrite the PLC application. Select **Ok** to continue the transfer or **Cancel** to abort. If you select **Ok**, the transfer is completed and the PC connects to the PLC. The application state changes from *offline* to *online*.

Select **Monitor** if you only want access to the application data pages and do not want to modify the application program, configuration, or symbols. The application state changes from *offline* to *monitor*.

Select **Cancel** to end the connection process and return to the *offline* state.

13.3 Stop/Run/Init

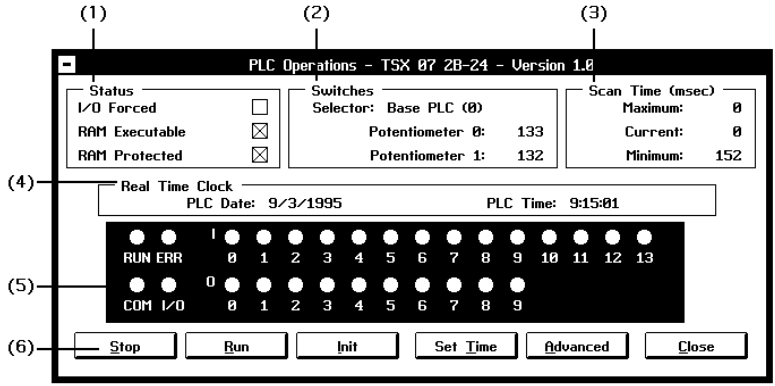
From the PLC menu it is possible to run, stop or initialize the PLC without displaying the PLC operations window. To do this, directly select the actions RUN, STOP or INIT. A dialog box confirming these choices will appear before the request is executed.



C

13.4 PLC Operations

Select **PLC Operations** from the PLC menu to display the PLC Operations dialog box.



The fields are numbered and described below :

(1) The **Status** field check boxes indicate whether :

One or more input or output bit is forced either 1 or 0.

An executable application is stored in the PLC RAM.

The PLC RAM protection bit was set to 1 during the Transfer process.

(2) The **Switches** field displays the settings you selected for the controls on the PLC. For information about setting the PLC function code and potentiometer switches, see Sections 1.8 and 1.10 in Part A.

(3) The **Scan Time (msec)** field, displays the minimum, current, and maximum scan time, expressed in milliseconds. For more information about program execution, see Section 1.3 in Part A.

(4) The **Real Time Clock** displays the current date and time. You can change the date and time by selecting the **Set Time** pushbutton at the bottom of the window. For more information, see Section 13.4-2.

(5) The LEDs in the PLC Operations window simulate the LEDs on the base PLC. For more information, see Section 1.9 in Part A.

(6) The **Stop/Run/Init/Set Time/Advanced/Close** pushbuttons let you control application execution on the PLC and view PLC system information.

13.4-1 Stop/Run/Init

To start executing an application in the PLC :

1. Select **Run**.
2. A warning message prompts you to confirm your decision to execute the application on the PLC.

   Select **Ok** to run the PLC. Select **Cancel** to return to the PLC Operations dialog box, leaving the state of the PLC unchanged.

To stop executing an application in the PLC :

1. Select **Stop**.
2. A warning message prompts you to confirm your decision to stop the execution of the application in the PLC.

   Select **Ok** to stop the PLC. Select **Cancel** to return to the PLC Operations dialog box, leaving the status of the PLC unchanged.

Select **Init** to initialize the PLC RAM. The initialization resets all memory variables.

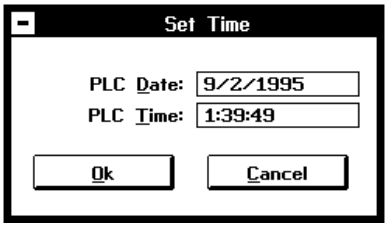
1. Select **Init**.
2. A warning message dialog box prompts you to confirm your decision to initialize the PLC.

   Select **Ok** to initialize the PLC. Select **Cancel** to return to the PLC Operations dialog box, leaving the status of the PLC unchanged.

13.4-2 Set Time

To set the PLC real time clock :

1. Select **Set Time** from the PLC Operations window.
2. In the **PLC Date** field, enter the current date, displayed as month/day/year.
3. In the **PLC Time** field, enter the current time. The time value appears in military format, hours:minutes:seconds. For example, 12 noon is 12:00:00, 2:15 PM is 14:15:00, and midnight is 00:00:00.



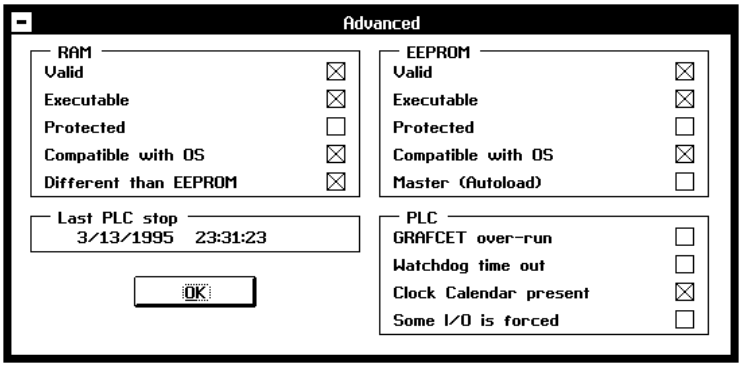
4. Select **Ok** to update the PLC date and time. Select **Cancel** to return to the PLC dialog box.

C

13.4-3 Advanced

Select **Advanced** to display read-only PLC system information, which you can view but cannot change.

A checked box indicates that the state is active. Use this information to check the PLC RAM and PLC EEPROM status and to diagnose problems.



RAM options

Valid	Indicates whether there is an application stored in the PLC RAM.
Executable	Indicates whether the application in the PLC RAM is executable.
Protected	Indicates whether you chose to protect the PLC RAM during the transfer from PC to PLC. (See Section 12.4)
Compatible with OS	Indicates whether the PLC version declared in the application configuration is the same as the target PLC operating system.
Different than EEPROM	Indicates whether the application in the PLC is different from the application in the EEPROM.

EEPROM options

Valid	Indicates whether there is an application stored in the PLC EEPROM.
Executable	Indicates whether the application in the EEPROM is executable.
Protected	Indicates whether you chose to protect the EEPROM during the PLC => EEPROM transfer process. (For more information, see Section 12.4.)
Compatible with OS	Indicates whether the PLC version declared in the application configuration is the same as the target PLC operating system.
Master (Autoload)	Indicates whether you selected the Master option during the PLC => EEPROM transfer process. (For more information, see Section 12.4.)

PLC options

GRAFCET over-run	Indicates whether the time needed to execute the Grafcet structure was greater than the scan time allowed.
Watchdog Time out	Indicates whether the scan time of a program exceeded 150 ms, causing the Watch dog timer to stop the PLC. (For more information, see Section 1.3 in Part A.)
Clock Calender present	Indicates whether the PLC has the Real Time clock feature.
Some I/O is forced	Indicates whether one or more input or output bits is forced ON or OFF.

14.1 Introduction

This section explains the various features of the PL7-07 PC programming software that can assist you in debugging and adjusting an application. These include animating a program, working with data variables, and using data pages.

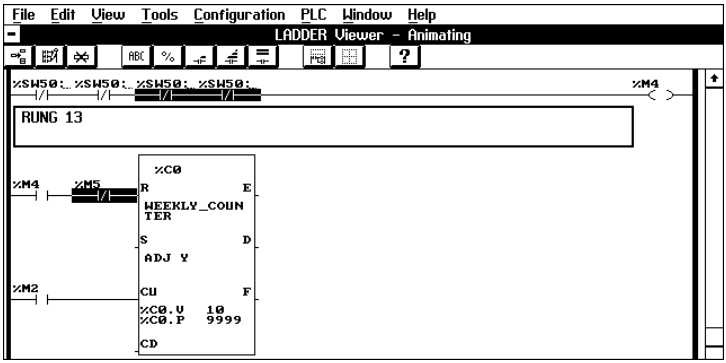
14.2 Animating a program

Animating a program allows you to view the values of the variables when a program is online, either running or stopped. This is useful for debugging because it allows you to view the change in values as the program runs to compare the actual values against expected values.

14.2-1 Animating a Ladder program

With the Ladder Viewer window displayed and the program online, either running or stopped :

1. Select **Toggle Animation** from the PLC menu.
2. The Ladder Viewer window is displayed with the following :
   - the title bar shows the word, "animating".
   - contacts, coils, and special objects with a logical result of 1 are highlighted.
   - the data variables of function blocks, compare block, and operate blocks are shown. This includes current and preset values. For other than binary values, the number is displayed in either hexadecimal or decimal, depending on the format selected in the Preferences dialog box.

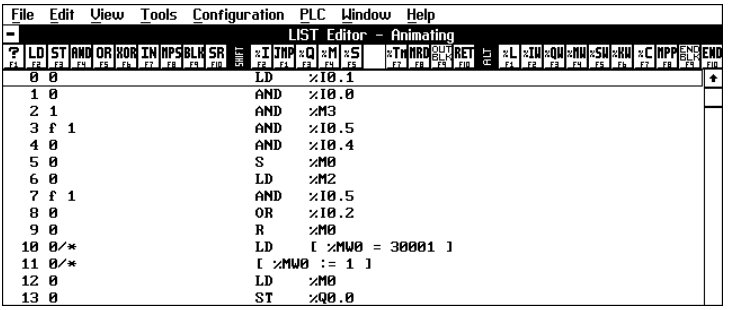


3. When the Ladder Viewer window is animated, select **Toggle Animation** from the PLC menu to turn off animation.

14.2-2 Animating a List program

With the List Editor window displayed and the program online, either running or stopped :

1. Select **Toggle Animation** from the PLC menu.
2. The List Editor window is displayed with an additional column just to the right of the line number. The column contains the value of the operand for that line. When an instruction line has more than one operand, the value of each of the operands is displayed and separated by slashes. Additionally, the title bar shows the word, "animating".



Binary operands are displayed as either 1 or 0. Word operands are displayed in either decimal or hexadecimal format, depending on the format selected in the Preferences dialog box.

The following values are not animated, but rather are represented by an asterisk :

- labels (%Li)
- subroutines (SRn)
- instructions that require no operand, NOT, NOP, END
- immediate values
- indexed words
- bits extracted from words
- table of words
- string of bits, such as %M0:5

Forced bits are displayed with an 'f', paired with the forced state, either 0 or 1. A bit operand that is forced on, is displayed as 'f 1', and a bit operand that is forced off, is displayed as 'f 0'. See Sections 14.4-7 through 14.4-10 for more information on forced values.

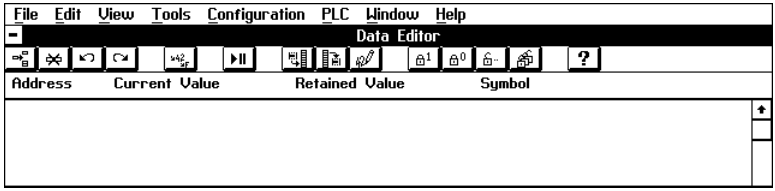
3. When the List Editor window is animated, select **Toggle Animation** from the PLC menu to turn off animation.

14.3 Using the Data Editor

The PL7-07 PC programming software uses the variables listed in appendix A.3 of part G to assist in writing the program. The Data Editor is used to view and modify these variables to assist in debugging a program. Additionally, the Data Editor can be used to force the values of input or output bits.

In the Data Editor window, you can define a list of PLC variables that you want to monitor and save. This list is called a "data page".

To display the Data Editor window, select **Data Editor** from the View menu.



The following gives an explanation of each column of the data page.

Address A specific location in memory, always preceded by a percent sign (%).

Current Value The actual value of the variable in the PLC. The value of the variable changes as the program runs. In the online state, you can animate the data page and watch the Current Value change as the program runs. An asterisk (*) appears in this column before the first time the data page is animated. After the first time the data page is animated and subsequently turned off, the Current Value column retains the last updated value.

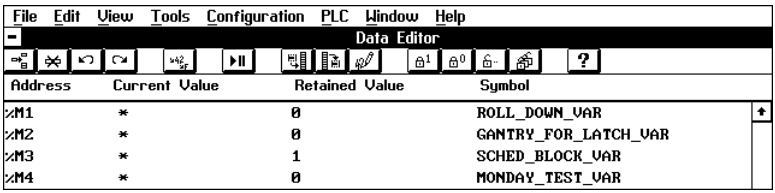
Retained Value The value specified as an initial value. When the Write Retained V Values function is performed, these values are written to the PLC.

Symbol The name assigned to an address in the Symbol Editor to identify the purpose of the variable.

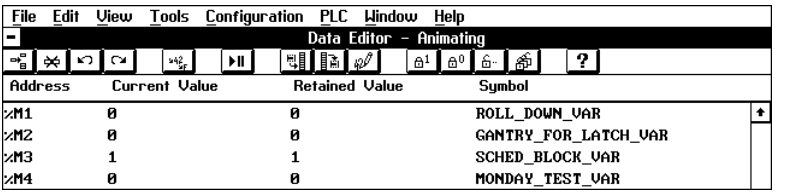
14.3-1 Animating a data page

Animating a data page displays and updates the Current Value column of the data page as the PLC program runs.

1. Before the data page is animated for the first time, the Current Value column contains asterisks (*).



2. With the data page displayed and the PLC online, either running or stopped, select **Toggle Animation** from the PLC menu to animate the data page.
3. The Current Value column is displayed with the current values from the PLC for the addresses listed in the data page. The title bar shows the word, "Animating".

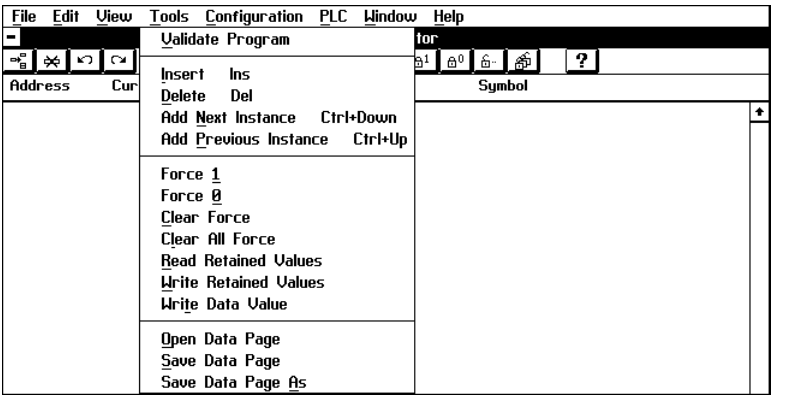


C

4. When the data page is animated, select **Toggle Animation** from the PLC menu to turn off animation. The Current Value column retains the last updated value.

14.4 Using the Data Editor Tools menu

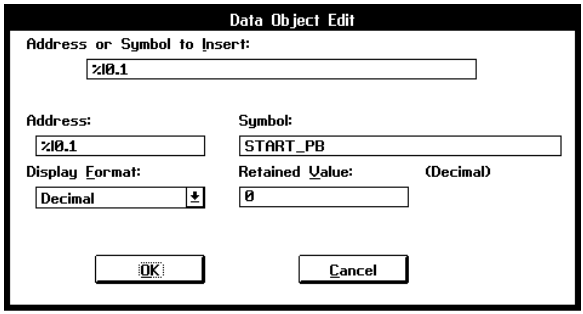
The Data Editor Tools menu is used to construct, edit, and save a data page. Additionally, from the Data Editor Tools menu you can modify and force values for selected variables in the program. The table in appendix A.2-7 of part G lists the options in the Data Editor Tools menu and the corresponding buttons on the tool bar.



14.4-1 Editing a data variable

To edit a single data variable on a data page :

1. With the data page displayed, double-click or press **<enter>** on the row with the data variable to be edited. The Data Object Edit dialog box appears.



2. The **Address** and **Symbol** fields display the address and symbol for the data variable to be edited. A symbol must have an assigned address, however, an address does not necessarily have an assigned symbol.
3. In the **Display Format** field, select the number format to determine how the value appears on the data page.

   Range : Decimal, Hexadecimal, Binary, ASCII

   Default : Decimal
4. In the **Retained Value** field, enter the desired initial value for the variable. This value is written to the PLC with all the other initial values of the data page when you perform the Write Retained Value operation.
5. Select **Ok** to commit the changes to the data variable displayed in the dialog box or **Cancel** to return to the Data Editor window without making any changes.

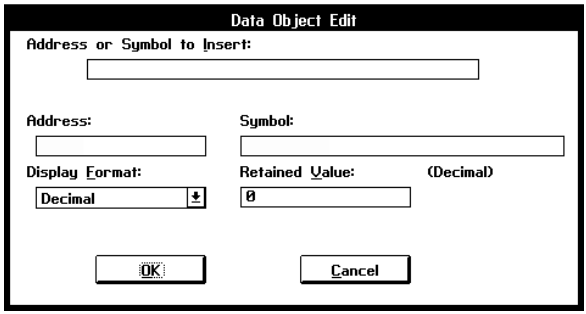
14.4-2 Validate Program

Use Validate Program to compile a program and check for errors. See Section 10.1 for more details on validating a program.

14.4-3 Insert

Use Insert to add a variable to the data page.

1. From the Data Editor window, select **Insert**. The Data Object Edit dialog box appears.



2. In the **Address or Symbol to Insert** field, enter the address or symbol to be added to the data page.

   Symbol names must have an associated symbol in the symbol table to be used in the data page. If a symbol name is entered that is not in the symbol table, an Error dialog box appears stating, " Invalid or Undefined symbol or address". Select **Ok** to return to the Data Object Edit dialog box. For more information on symbols, see Section 6, "Defining symbols."

3. The **Address** and **Symbol** fields are used to display the corresponding symbol for an address entered in step 2 or display the corresponding address for a symbol entered in step 2. A symbol must have an assigned address, however, an address does not necessarily have an assigned symbol.

4. In the **Display Format** field, select the number format to determine how the value will be displayed on the data page.

   Range : Decimal, Hexadecimal, Binary, ASCII

   Default : Decimal

5. In the **Retained Value** field, enter the initial value for the variable. This value is written to the PLC with all the other retained values on the data page when you perform the Write Retained Values operation.

6. Select **Ok** to commit the values to the data page or **Cancel** to exit and return to the Data Editor window.

14.4-4 Delete

Use Delete to remove a variable from the data page.

With a data page displayed, highlight the variable to be removed. Select **Delete**. The variable is removed from the data page.

14.4-5 Add Next Instance

Use Add Next Instance to add the following sequential instance of the variable highlighted on the data page.

With a data page displayed, highlight the desired variable of the same type to be added. Select **Add Next Instance**. A new variable of the same type is added to the data page with the following sequential number.

For example, if %I0.3 is currently highlighted, selecting Add Next Instance adds the variable %I0.4 to the data page.

14.4-6 Add Previous Instance

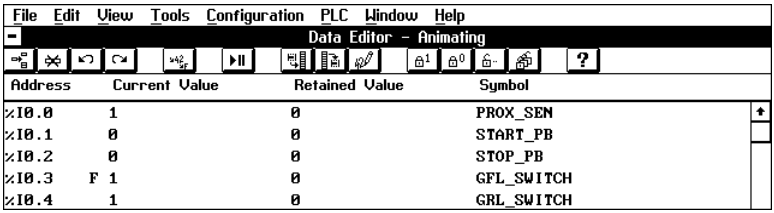
Use Add Previous Instance to add the preceding sequential instance of the variable highlighted on the data page.

With a data page displayed, highlight the desired variable of the same type to be added. Select **Add Previous Instance**. A new variable of the same type is added to the data page with the previous sequential number.

For example, if %I0.3 is currently highlighted, selecting Add Previous Instance adds the variable %I0.2 to the data page.

14.4-7 Force 1

Use Force 1 to set or force an input or output bit to 1, even if the calculated value is different from the forced value. When a variable's value is forced, the value stays forced until it is cleared—even if the PC is disconnected from the PLC and you exit the PL7-07 PC programming software.
Force 1 is available when the PLC is online, either running or stopped. The Current Value column in the data page has a 'F' next to the value if it is a forced value and the data page is animated. The table in appendix A.3 in part G lists the variables that can be forced.
With a data page displayed and animated, select **Force 1**. The selected variable displays a 'F' next to the current value of 1. For example, the following shows input %I0.3 with a forced value of 1.



14.4-8 Force 0

Use Force 0 to set or force an input or output bit to 0, even if the calculated value is different from the forced value. When a variable's value is forced, the value stays forced until it is cleared—even if the PC is disconnected from the PLC and you exit the PL7-07 PC programming software.
Force 0 is available when the PLC is online, either running or stopped. The Current Value column in the data page has a 'F' next to the value if it is a forced value and the data page is animated. The table in appendix A.3 in part G lists the variables that can be forced.
With a data page displayed and animated, select **Force 0**. The selected variable displays a 'F' next to the current value of 0.

14.4-9 Clear Force

Use Clear Force to remove a forced value from a variable on the data page. Clear Force is available when the PLC is online, either running or stopped.

1. With a data page displayed and animated, select **Clear Force** to remove the forced value for the highlighted variable .
2. The Data Editor window displays the variable with the forced value removed.

14.4-10 Clear All Force

Use Clear All Force to remove all forced values from a data page. Clear All Force is available when the PLC is online, either running or stopped.

1. With a data page displayed and animated, select **Clear All Force** to remove the forced values for the entire data page.
2. The Data Editor window displays the data page with all forced values removed.

14.4-11 Read Retained Values

Use Read Retained Values to transfer the Current Values from the PLC to the Retained Values on the data page. Read Retained Values is available when the PLC is online, either running or stopped.

1. With a data page displayed and animated, select **Read Retained Values** to transfer the values in the Current Value column to the Retained Value column.
2. The data page appears with the Retained Value column updated with the values from the Current Value column.

14.4-12 Write Retained Values

Use Write Retained Values to transfer the Retained Values on the data page to the Current Values in the PLC. Write Retained Values is available when the PLC is online, either running or stopped.

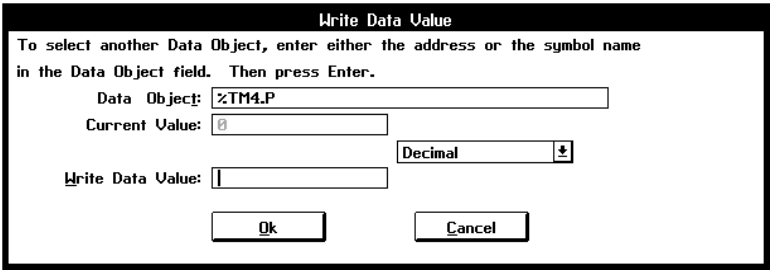
1. With a data page displayed and animated, select **Write Retained Values** to transfer the values in the Retained Value column to the Current Value column for all variables on the data page.
2. The Current Value column appears with the values from the Retained Value column, unless they changed immediately after being written to the Current Value column.

14.4-13 Write Data Value

Use Write Data Value to momentarily send, or write, a single data value to the PLC. Write Data Value is available when the PLC is online, either running or stopped. The table in appendix A.3 in part G lists the variables that can have values written to them.

Write Data Value is available from the Data Editor window, whether or not a data page is opened.

1. With the Data Editor window open, select **Write Data Value**. The Write Data Value dialog box appears.



2. Enter the data variable in the **Data Object** field.
3. The **Current Value** field displays the current value for the selected variable.
4. Select the display format for the value. This format selection is only for the display of the value being entered in the Write Data Value dialog box.

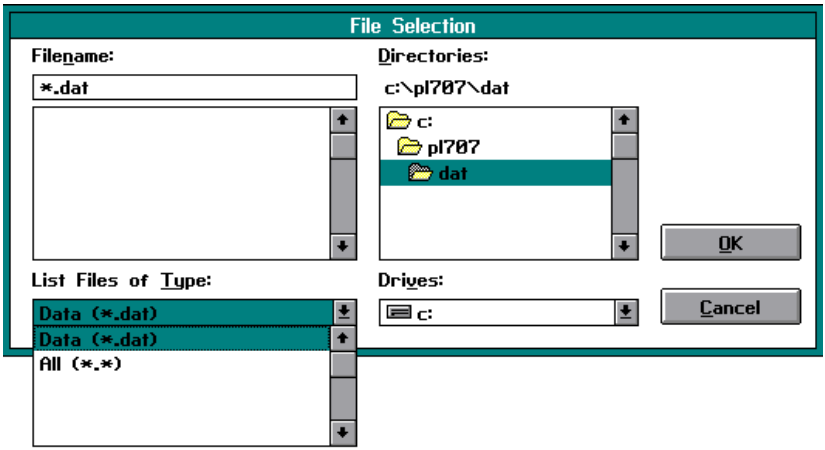
   Range : Decimal, Hexadecimal, Binary, ASCII

   Default : Decimal
5. In the **Write Data Value** field, enter the value you want to write to the PLC for the data variable.
6. Select **Ok** to write the value to the PLC or **Cancel** to return to the data page without writing the value to the PLC.

14.4-14 Open Data Page

Use Open Data Page to open a data page that was previously saved.

1. With the Data Editor window displayed, select **Open Data Page** from the Tools menu. The File Selection dialog box appears.



2. In the **List Files of Type** field, open the selection box and select either the .dat file type or select to have all file types listed (*.*).

   Default : .dat
3. In the **Drives** field, select the drive where the data files are stored.
4. In the **Directories** field, select the PL7-07 directory or the directory where the data files are stored.
5. In the **Filename** field, select the data file to be opened.
6. Select **Ok** to open the data file or **Cancel** to return to the Data Editor window.

14.4-15 Save Data Page

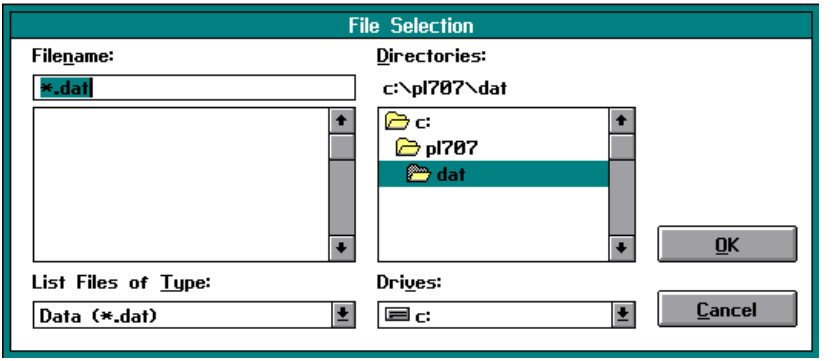
Use Save Data Page to save changes to an existing data page file.

After making the desired changes to a data page, select **Save Data Page** to save the changes to the data page file.

14.4-16 Save Data Page As

Use Save Data Page As to save a data page to a new file.

1. With the Data Editor window open and a data page displayed, select **Save Data Page As**. The File Selection box appears.



2. In the **List Files of Type** field, select the default file extension, .dat.
3. In the **Drives** field, select the drive where the data file will be stored.
4. In the **Directories** field, select the PL7-07 directory or the directory where the data files will be stored.
5. In the **Filename** field, over type the asterisk (*) with a standard DOS file name.

   If the file name does not comply with DOS file naming conventions, an "Invalid Filename" message is displayed.

   If a file name is selected that already exists in the directory, an error message is displayed : "The file selected already exists. Overwrite?" Select **Ok** to overwrite the file or **Cancel** to return to the File Selection dialog box.
6. Select **Ok** to save the data file or **Cancel** to return to the Data Editor window.

14.5 Modifying the program in RUN mode

The List program editor allows the List program to be modified when the PLC is in RUN mode.

CAUTION

UNEXPECTED EQUIPMENT OPERATION

For obvious safety reasons, it is advisable to program a PLC in STOP.

It is possible, however, to program a PLC which is in RUN mode so that program modifications not requiring the application to be stopped can be performed. However, these modifications remain the responsibility of the user.

Before any modifications are performed, certain conditions for programming a PLC while running must be fulfilled. It is essential to know the consequences of the modifications on the application, and to take any necessary actions to determine what these are.

Failure to observe this precaution can result in equipment damage, severe personal injury, or death.

Modifying in RUN mode

The process for modifying in RUN mode is identical to that for programming in RUN mode. The modification takes effect as soon as the current entry is confirmed.

Restrictions

The following table shows the restrictions on use and on modification when the PLC is in RUN mode :

Mode/Function	Access
Configuration	Access to display only
Programming	All modifications/insertions/cancellations of instructions are permitted, except for instructions which modify the structure of the program : • parentheses • Grafcet instructions • labels • jump : JMP and SR subroutine calls • master control relays MCR and MCS • block instructions : BLK, OUT_BLK, END_BLK • MPS, MPP The following instructions are not active : • find and replace • program transfer to EEPROM

15.1 Introduction

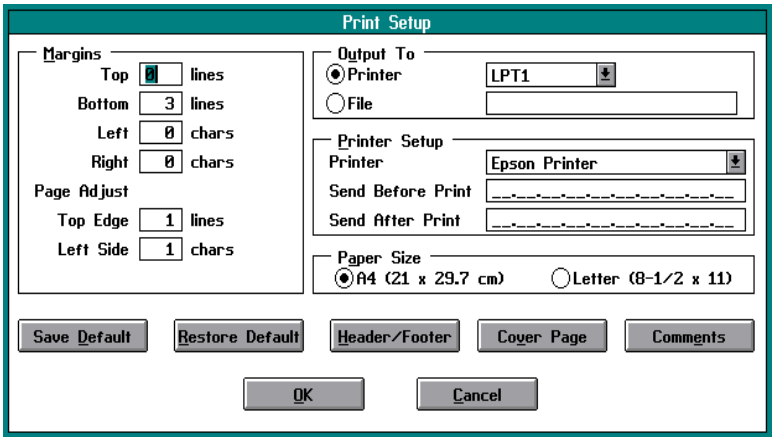
PL7-07 PC Programming Software can print the whole or part of an application.

15.2 Print Setup

Print Setup is used to define a printer ID or file name and the page layout.

To setup a printout:

1. Select **Print Setup** from the File menu. The Print Setup dialog box appears.



2. In **Printer Setup** open the printer selection box to select a printer type:

   Values: Text Printer, HP Compatible, Epson Printer.

   Default: Epson Printer
   The lines Send Before Print and Send After Print enable specific commands to be sent to the printer before exiting the file and after printing.
   Default : None.

3. Select the printing format in **Paper Size**:

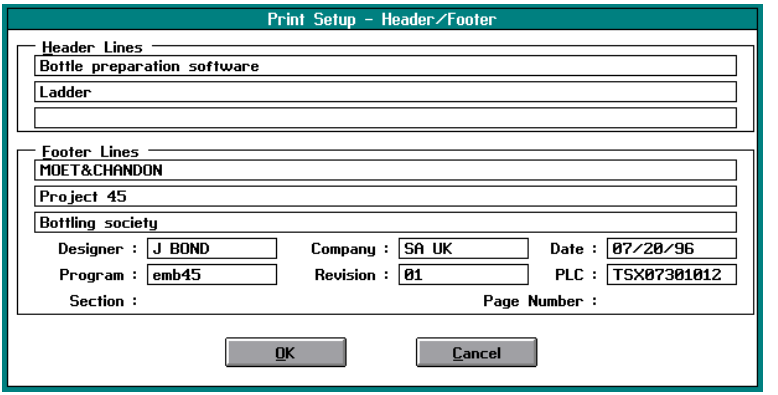
   Values: A4 (21 x 29.7 cm), Letter 8 1/2 x 11

   Default: A4

4. In the **Output to** (printer or file) field, type either the printer ID, such as *LPT2*(parallel port #2), or a file name, such as *PRINTOUT.TXT,* using a standard DOS file name.

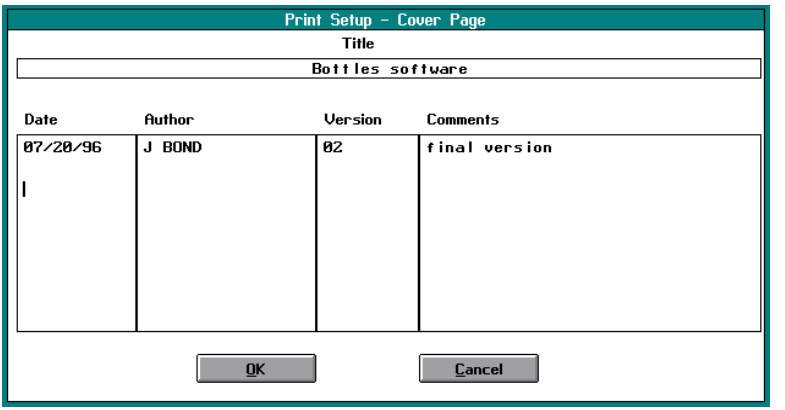
   Default: LPT1 (parallel port #1)

5. To define the page header and footer, select the **Header/Footer** button to display the following window:



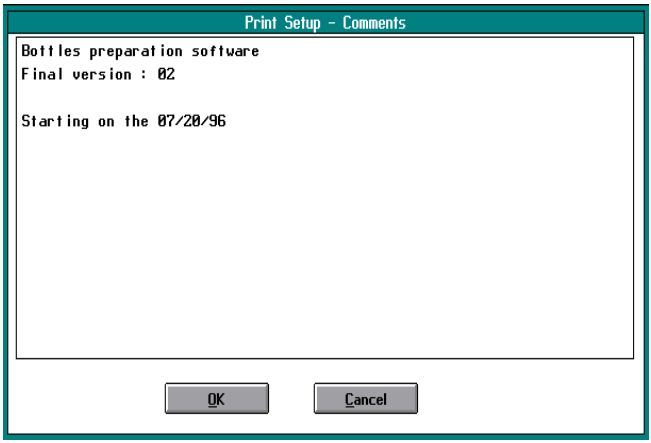
You can type up to three lines of text to display at the top of each page. You can also type up to three lines of text to display at the bottom of each page.
Similarly, it is possible to print at the bottom of the page the name of the originator, the company and the program as well as the date, the version and the type of PLC used.

6. To define the cover page of the application file, select the **Cover Page** button in the **Print Setup** dialog box:



Enter a title of up to 20 characters and the development history of the application: Date, author, version and comments.

7. If a comments page is required at the beginning of the application file, select the **Comments** button in the **Print Setup** dialog box:



This editor enables a page of 60 lines of 70 characters each to be entered. Line advance is obtained by using carriage return (ENTER) when entering.

8. In **Margins**, enter a number in the field Adjust Page Left to increase or decrease the width of the left margin by the number of characters required. Enter a number in the field Page Adjust Top Edge to increase or decrease the top margin by the number of lines required.

9. In **Margins**, define the width of the left and right margins in numbers of characters. Define the width of the top and bottom margins in numbers of lines.
It is necessary to make sure that the number of margin lines correspond with the header and footer.

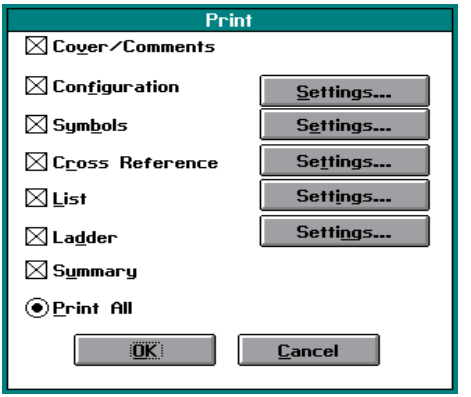
10. Select **Save Default** to save the printer and page layout values you selected as the default values for future printing operations.

11. Select **Restore Default** to overwrite the current values with the values that you saved earlier by using the Save Default button.

12. Select **Ok** to save your settings. Select **Cancel** to exit without saving the settings you defined.

15.3 Print

Select **Print** from the File menu. The Print dialog box appears.

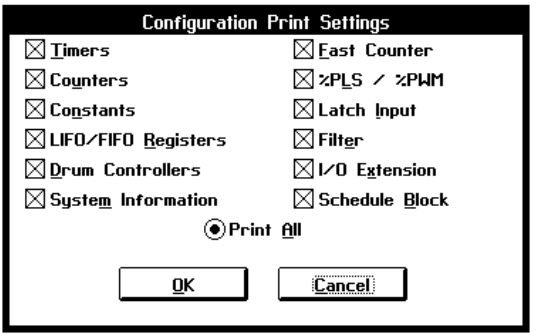


The options in the Print dialog box let you define the scope of your printout. Select **Print All** to print the entire application. Or, print only specific application parts by selecting the appropriate checkboxes. Select **Ok** to print. Select **Cancel** to exit the Print dialog box without printing.

15.3-1 Configuration Print Settings

If you want to print some, but not all, configuration data:

1. In the Print dialog box, select the **Configuration Settings** button to display the Configuration Print Settings dialog box.



2. Select the checkboxes for the Configuration data that you want to print. Select **Print All** to print all configuration data for an application.

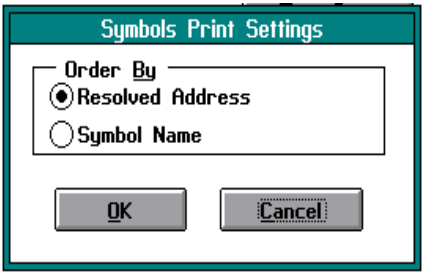
   Default: All configuration data items selected.

3. Select **Ok** to save your changes. Select **Cancel** to return to the Print dialog box without saving your selections.

15.3-2 Symbols Print Settings

To define a printout of application symbols:

1. In the Print dialog box, select the **Symbols Settings** button to display the Symbols Print Settings dialog box.



2. In the **Order By** field, select **Resolved Address** to sort the operands by address or select **Symbol Name** to sort the operands by symbol name.

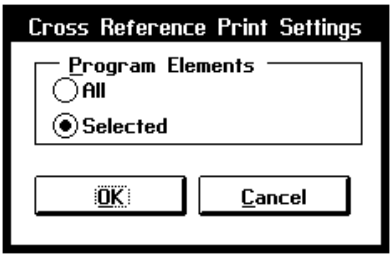
   Default: Resolved Address

3. Select **Ok** to save your changes. Select **Cancel** to return to the Print dialog box without saving your selections.

15.3-3 Cross Reference Print Settings

To define a printout of cross reference program elements:

1. In the Print dialog box, select the **Cross Reference Settings** button to display the Cross Reference Print Settings dialog box.



2. In the **Program Elements** field, select **All** to print all elements of a program or **Selected** to only print those elements that you selected in the Generate Cross Reference dialog box. If you want to re-select those elements, select Cross Reference from the View menu and regenerate the cross reference.

   Default: All

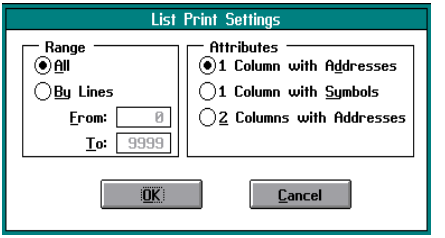
3. Select **Ok** to save your changes. Select **Cancel** to return to the Print dialog box without saving your selection.

C

15.3-4 List Print Settings

To define a printout of List program lines:

1. In the Print dialog box, select the **List Settings** button to display the List Print Settings dialog box.



2. In the **Range** field, select **All** to print all list lines. Or, select **By Lines** and enter the first and last numbers of the program block that you want to print.

   Default: All

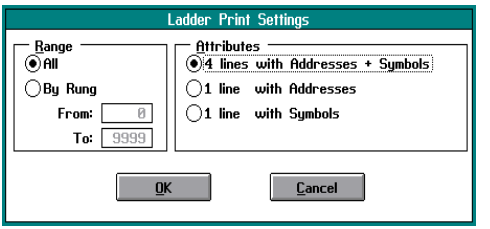
3. In the **Attributes** field, select:
   • **1 Column with Addresses** to print the application code with the addresses of the objects.
   • **1 Column with Symbols** to print the application code with the symbols associated with the objects.
   • **2 Columns with Addresses** to print the code in a condensed form in two columns with the addresses of the objects.
   Default : 1 Column with Addresses.

4. Select **Ok** to save your changes. Select **Cancel** to return to the Print dialog box without saving your changes.

15.3-5 Ladder Print Settings

To define a printout of Ladder rungs:

1. In the Print dialog box, select the **Ladder Settings** button to display the Ladder Print Settings dialog box.



2. In the **Range** field, select All to print all ladder rungs in the file. Or, select By Rung and enter the first and last rung numbers of the Ladder block that you want to print.

   Default: All

3. In the **Attributes** field, select:
   • **4 lines with Addresses + Symbols** to print the application code with the symbols and addresses of the objects. 3 lines are available to display the entire symbol and one line for the address. These 4 lines are displayed above the graphic element in the Ladder rung.
   • 1 **line with Addresses** to print the application code with the addresses of the objects.
   • 1 **line with Symbols** to print the application code with the symbols associated with the objects.
   Default : 4 lines with Addresses + Symbols.

4. Select **Ok** to save your changes. Select **Cancel** to return to the Print dialog box without saving your selections.

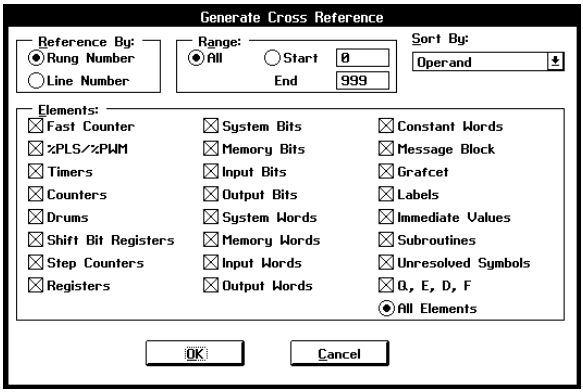
16.1 Introduction

The cross reference list provides a useful list of the operands, symbols, lines and operators. For example, during troubleshooting or debugging, it allows the item of interest to be easily located and cross referenced to other locations in the program without having to search through the entire program.

16.2 Generating a new Cross Reference list

To generate a cross reference list for the first time after an application has been opened:

1. Select **Cross Reference** from the View menu. An Information dialog box appears. Select **Ok** to display the Generate Cross Reference dialog box. Selecting **Cancel** will display the Cross Reference window without generating the cross reference list.



Note:
It is advisable to validate the program before generating cross references.
The cross reference list is not saved when the application is closed. Therefore, when the application is re-opened, the cross reference list must be generated.

2. In the **Reference By** field, select whether you want to reference by ladder rung number or list line number.
3. In the **Range** field, define the scope of the display.

   Select **All** to display all list lines or all ladder rungs in a program.

   Default: All

Select **Start/End** to mark a specific block of the program that you want to display. The range for both fields is 0 to 999. For **Start**, the default value is 0. For **End**, the default value is 999.

4. In the **Sort By** field, select one field to act as the key field when the cross reference list is displayed. These options correspond to the **Sort** options in the **Tools** menu.

   Values: Operand, Symbol, Line/Rung Number, Operator

   Default: Operand

For example, if Sort by Operand is selected, the following window appears—sorted alphabetically and numerically by operand.

PL7-07 - c:\pl707\app\appli.pl7

File Edit View Tools Configuration PLC Window Help

Cross Reference

Operand	Symbol	Line	Operator
%I0.0	START_INPUT	1	LD
%I0.1	AUTO_INPUT	4	LD
%I0.2	MANUAL_INPUT	5	ORN
		8	LD
%I0.3	RESET_INPUT	9	ORN
		12	LD
%I0.4	COUNT_INPUT	15	LD
%I0.5	OVER_FLOW_INPUT	18	LD
%M0	RESET	2	ST
%M1	HIGHT_LEVEL	10	ST
%M2	FLOP	6	ST
%M4	FLIP	16	ST

When Sort By Operand or Sort By Symbol is selected, a separation line frames each operand or symbol so that the information is easier to read.

5. In the **Elements** field, select the check boxes that correspond to the program items that you want listed.

   The selection of specific program items from the Elements field allow you to define the scope of your display more narrowly. For example, you can choose to list only program lines or rungs that have timer function blocks.

16.3 Updating an existing Cross Reference list

An existing cross reference list will need to be regenerated in the following cases:

- Modification of the Rung Number system of reference to Line Number or vice versa in the Reference By field,
- Modification of the range of the cross reference,
- Modification of the current cross reference list (adding or deleting elements),
- Any change to the application program and/or symbols.

1. If the cross reference list is to be updated while the cross reference list is displayed, select Generate Cross Reference from the Tools menu.

   The Generate Cross Reference dialog box will appear. After selecting all the desired fields in the dialog box, select **Ok** to generate a new cross reference list.

   Select **Cancel** to return to the existing cross reference list.

2. If you do not have the cross reference list displayed, select Cross Reference from the View menu.

   The current cross reference list will be displayed. Select Generate Cross Reference from the Tools menu. The Generate Cross Reference dialog box will appear.

   After selecting all the desired fields in the dialog box, select **Ok** to generate a new cross reference list. Select **Cancel** to return to the existing cross reference list.

3. If you only desire to re-order the existing cross reference list by a different key field (while the cross reference list is displayed), select the desired key field from the Tools menu.

   The options are Sort by Operand, Sort by Symbol, Sort by Line/Rung, and Sort by Operator.

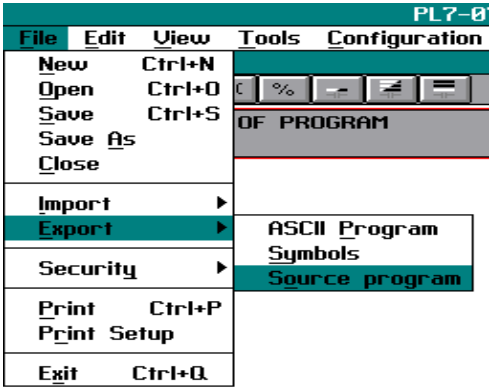
   After the desired key field is selected, the program will re-order the cross reference list by the selected key field.

17.1 Introduction

Using a PL7-07 application it is possible to generate a text file using a source program export function. This text file is compatible with the PL7 Micro import function.
This operation transfers a PL7-07 application for TSX Nano PLC to the PL7 Micro programming software for TSX Micro PLCs.

17.2 Principle

To perform the export, the application must previously have been open in PL7-07. The function can be accessed from the File-Export-Source program menu:



A browser window is used to select the name and the target directory where the file to be exported will be placed.

The file extension, *.IL for List or *.LAD for Ladder, is automatically selected according to the editor selected in preferences. If the user changes the export file extension, the format (List or Ladder) of the file will change. It is therefore possible to select the format of the export file independently of the language chosen to create the application.

Note

The application to be exported must be valid in List or reversible List/Ladder.

An export file contains:

- An information header
- The List or Ladder program
- The configuration objects compatible with PL7 Micro : %TMi, %Ci, %Ri, %DRi, %Kwi
- The symbols associated with objects compatible with PL7 Micro : %Mi, %MWi, %Si, %SWi, %Ii.j, %Qi.j

The contents of the exported program, to conform with the application structure of PL7 Micro do not contain:

- Line or end of line comments placed in the rung
- Subroutines
- Grafcet steps
- Post-Grafcet processing (POST)
- Objects which are not compatible with PL7 Micro : Fast counters, etc.

In the exported program, positions containing unsupported object addresses are replaced by a blank.

D

Section **Page**

1.1 Troubleshooting using status LEDs

The user is informed about the operating mode together with any PLC operating faults by the status LEDs on the front panel.

Note :
Each time the PLC is switched on, all the LEDs are lit for a period of approximately one second corresponding to the self-test phase. Outputs are not activated.

1.1-1 Troubleshooting the "base PLC" PLC or peer PLC

LED status		Meaning	Probable cause
RUN LED	○	PLC off or application not executable	PLC not powered or syntactically invalid application.
	⊠	PLC in STOP	Required stop (control using RUN/STOP %I0.• input or the terminal) or stop caused by an execution fault (watchdog timeout, or use of a function which has not been implemented or is prohibited).
	●	PLC in RUN	Normal operation.
ERR LED	○	Operation OK	Normal operation.
	⊠	Application not executable	Application not present, invalid, checksum fault, watchdog timeout.
	●	Internal faults	• Hardware watchdog timeout. • Self-tests indicate a fault (Prob accessing RAM, EEPROM or schedule block). • Mis-wiring on extension port
I/O LED	○	Operation OK	Normal operation.
	●	I/O fault	I/O faults : • Outputs failed • Sensor supply fault. • Configuration fault.
COM LED	○	No exchange present on extension link	
	●	Exchanges present on extension link	
	⊠	Exchanges present on Modbus link	

○ LED off ⊠ LED flashing ● LED lit continuously

D

1.1-2 Troubleshooting the I/O extension

LED status		Meaning	Probable cause
RUN LED	◯	Extension off or not connected to extension link	Extension not powered, not connected or connection error on extension connection cable.
	⊗	Base PLC in STOP (same as base PLC LED)	Identical to those of base PLC.
	●	Base PLC in RUN (same as base PLC LED)	Normal operation.
ERR LED	◯	Operation OK	Normal operation.
	●	Internal faults	• Hardware watchdog timeout. • Self-tests show a fault. • Mis-wiring on the extension port
I/O LED	⊗	Operation OK	Normal operation.
	●	Extension I/O fault	I/O faults : • Outputs failed • Sensor supply fault.
COM LED	◯	No exchange present on extension link	
	●	Exchanges present on extension link	

◯ LED off ⊗ LED flashing ● LED lit continuously

D

1.1-3 Troubleshooting the "base PLC" PLC, peer PLC or I/O extension

LED status		Meaning	Probable cause
LED I0 to I13	○	Input not active	Normal operation if sensor not conducting.
	●	Input active	Normal operation if sensor is conducting
LED O0 to O9	○	Output not active	Normal operation if output not activated.
	●	Output active	Normal operation if output activated.

○ LED off ● LED lit continuously

If LEDs I5 (TSX Nano 10 I/0), I7 (TSX Nano 14 I/O), I8 (TSX Nano 16 I/0), I11 (TSX Nano 20 I/O) or I13 (TSX Nano 24 I/0) are flashing (series of 5 short flashes every second), the memory display mode is enabled. (1)
LEDs I0 to I7 and O0 to O7 show state 0 (off) or 1 (lit) of internal bits %M112 to %M127.

LED	Meaning	Type	LED	Meaning	Type
I0	%M112	TSX Nano	O0	%M120	TSX Nano
I1	%M113	10/14/16	O1	%M121	10/14/16
I2	%M114	20/24 I/O	O2	%M122	20/24 I/O
I3	%M115		O3	%M123	
I4	%M116	TSX Nano	O4	%M124	TSX Nano
I5	%M117	20/24 I/O	O5	%M125	20/24 I/O
I6	%M118		O6	%M126	
I7	%M119		O7	%M127	

(1) In this case, system bit %S69 is at state 1.

1.1-4 Troubleshooting TSX AMN 4000/4001 analog modules

LED status		Meaning	Probable cause
RUN LED	○	Module off or application not executable	Module not powered or connected
	●	Application in RUN	Normal operation.
ERR LED	○	Operation OK	Normal operation.
	⊗	Incorrect module addressing	Modify the position of the selector switch on the front panel of the module
	●	Internal faults	• Self-calibration fault. • Self-tests indicate a fault.
I/O LED	○	Operation OK	Normal operation.
	●	High or low stops exceeded on analog inputs	
COM LED	○	Communication in progress	
	●	Exchanges in progress on extension link	

○ LED off ⊗ LED flashing ● LED lit continuously

D

1.2 Fault analysis using system bits and words

If a fault is detected, the PLC system sets a system bit or word corresponding to the error which has been detected. The application program can use this information.
The FTX 117 terminal or PL7-07 Data editor (on FTX 417/507, FT 2000 or compatible PC) can display system bits (%S key) and system words (%SW key). See Data editor, Part C, Section 9 in the FTX 117 manual, or Part C, Section 14 of the PL7-07.

1.2-1 System bits

System bit	Function	Description
%S10	I/O fault	Normally at 1. It is set to state 0 when an I/O fault is detected on the base PLC or the peer PLC (configuration fault, exchange fault, hardware fault, disconnection of protected solid state output). Bits %S118 and %S119 indicate in which PLC the fault is and words %SW118 and %SW119 detail the nature of the fault. (See Section 5.2) Bit %S10 is reset to state 1 when the fault is cleared.
%S11	Watchdog timeout	Normally at 0, it is set to state 1 by the system when the program execution time (scan time) exceeds the maximum scan time (150 ms software watchdog). Watchdog timeout causes PLC to change to STOP).
%S19	Scan period overrun (periodic task)	Normally at 0, this bit is set to 1 by the system in the event of a scan period overrun (scan time greater than the period defined by the user at configuration or programmed in SW0). This bit must be reset to 0 by the user.
%S71	Exchange via extension link	Initially set to 0. It is set to 1 when an exchange via the extension link is detected. This bit is set to 0 when no exchange is performed via the extension link. Word %SW71 in the base PLC shows the list and status of the available extensions.
%S118	I/O fault on base PLC	Normally at 0. It is set to 1 when an I/O fault is detected on the base PLC. Word %SW118 determines the nature of the fault. Bit %S118 is reset to 0 when the fault is cleared.
%S119	I/O fault on peer PLC	Normally at 0. It is set to 1 when an I/O fault is detected on the peer PLC. Word %SW119 determines the nature of the fault. Bit %S119 is reset to 0 when the fault is cleared.

1.2-2 System words

System word	Function	Description
%SW71	Devices on PLC comm link	Shows the comm status of each peer PLC and I/O ext. with the base PLC : bit 1 : I/O extension bit 2 : peer PLC n°2 bit 3 : peer PLC n°3 bit 4 : peer PLC n°4 Bit at 0 if there is no extension or peer, no power, not wired or a fault. Bit at 1 peer PLC present and exchange with the base PLC.
%SW118	Base PLC status	Shows faults detected on the base PLC. bit 0 : 0= failure of one of the outputs (1) bit 3 : 0= sensor supply fault bit 8 : 0= TSX 07 internal fault or hardware fault bit 9 : 0= external fault or comm. fault bit 11 : 0= PLC performing self-tests bit 13 : 0= configuration fault All other bits of this word are at 1 and are reserved. Thus, for a PLC which has no fault, the value of this word is : 16#FFFF.
%SW119	Peer PLC status	Shows faults detected on peer PLC (this word is only used by the base PLC). The bit assignment of this word is identical to that of word %SW118, except for : bit 13 : not significant bit 14 : peer PLC now missing although it was present at initialization

(1) after a shortcircuit or overload of the protected transistor outputs.

Important

The example described in this section is for information only. When used in an industrial application it will require modifications depending on the safety rules which apply to the sector of activity concerned.

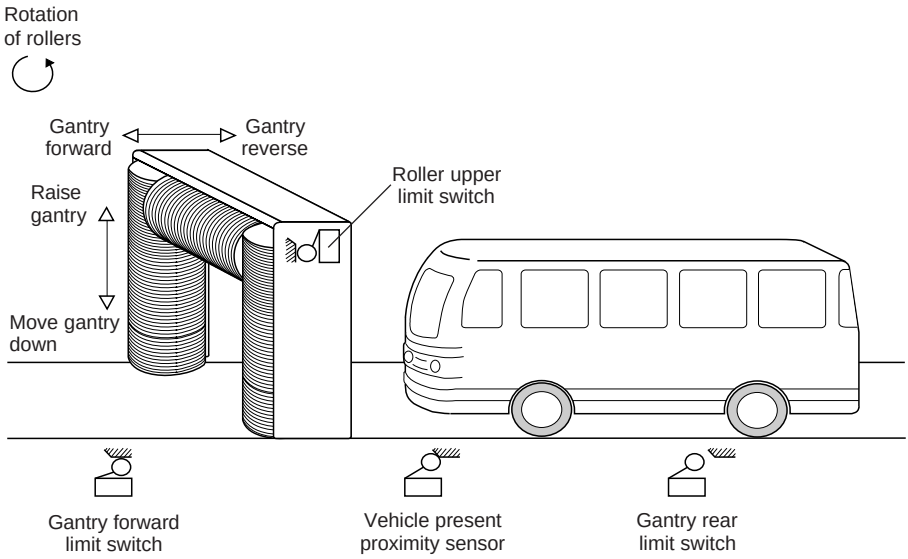
1.1 Description of the application

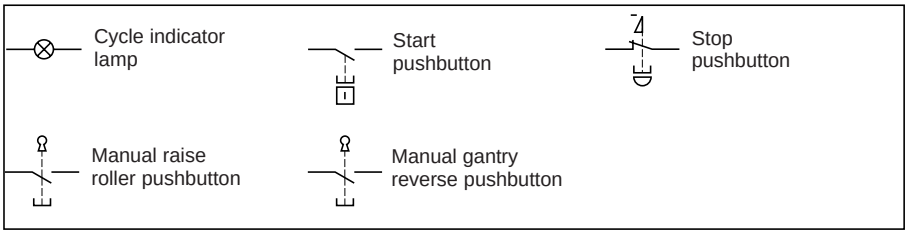
A car wash comprises :

- A gantry supporting a horizontal roller and vertical rollers, driven by a motor with two operating directions (forward and reverse).
- A motor to rotate the horizontal roller and vertical rollers.
- A motor to raise and lower the horizontal roller.

Limit switches control :

- Upper limit of the horizontal roller
- Forward and rear limit of the gantry.





E

1.2 Application operation

1.2-1 Automatic wash cycle

Initial conditions : The gantry is at the "rear" position (rear limit switch) and the horizontal roller is in the raised position (upper limit switch). A vehicle is present in the washing area (vehicle present proximity sensor).

When the initial conditions are fulfilled, pressing the start pushbutton starts the following cycle :

- Cycle indicator lamp is lit, followed by a ten second wait period (KA0).
- The horizontal roller moves down (KM1) for five seconds (KA1).
- The rollers start to rotate (KM3) and the gantry moves forward (KM4). It is presumed in this example that the water spray pumps are activated at the same time as the motor which rotates the rollers.
- Forward movement of the gantry is stopped by the gantry forward limit switch and the gantry reverse command is given (KM5).
- Gantry reverse and rotation of the rollers are stopped by the gantry rear limit switch. The command is given to raise the horizontal roller (KM2) up to the upper limit switch which ends the cycle.

A real-time clock controls the times (Monday to Saturday, 8.00 to 19.30). Outside these times, no start request is accepted.

A weekly counter counts the number of washes performed. It is automatically reset to zero every Monday at 8.00 a.m. Another module counts the total number of washes performed over the weeks.

E

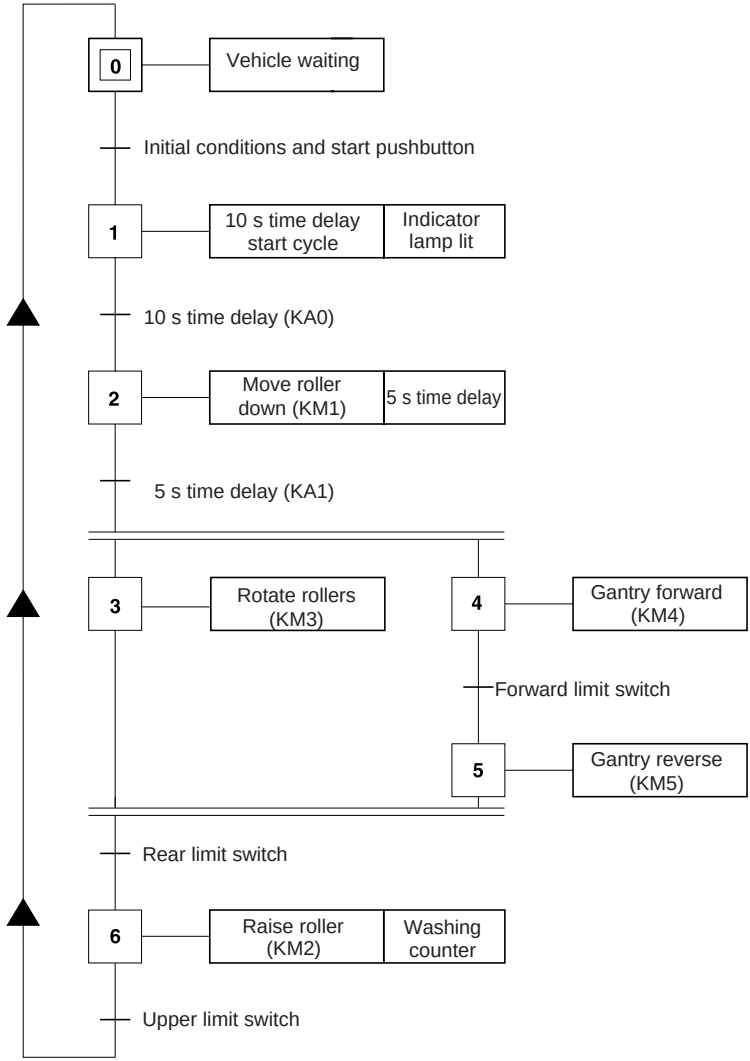
1.2-2 Manual immediate stop

A latching pushbutton can stop the cycle at any time (immediate stop of all motors). To start a new cycle, the following operations must be performed :

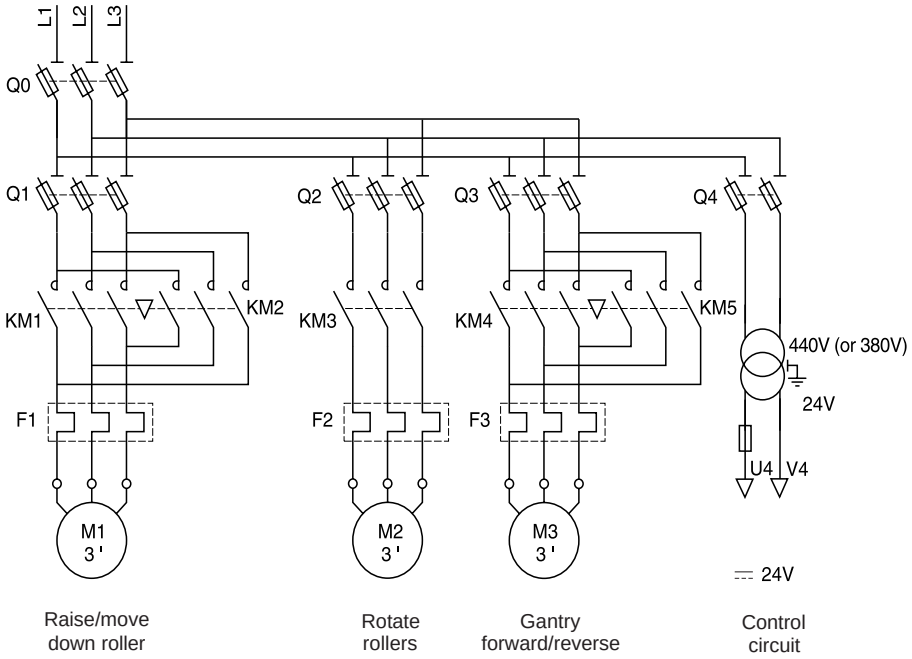
- Raise the horizontal roller to the upper position (to upper limit switch) by holding down the manual raise roller pushbutton.
- Return the gantry to the rear position (up to rear limit switch) by holding down the manual gantry reverse pushbutton.
- Release the stop pushbutton.

1.3 Graphic representation of the wash cycle

The Grafcet chart below is a graphic representation of the automatic operation of the car wash.



2.1 Power circuit

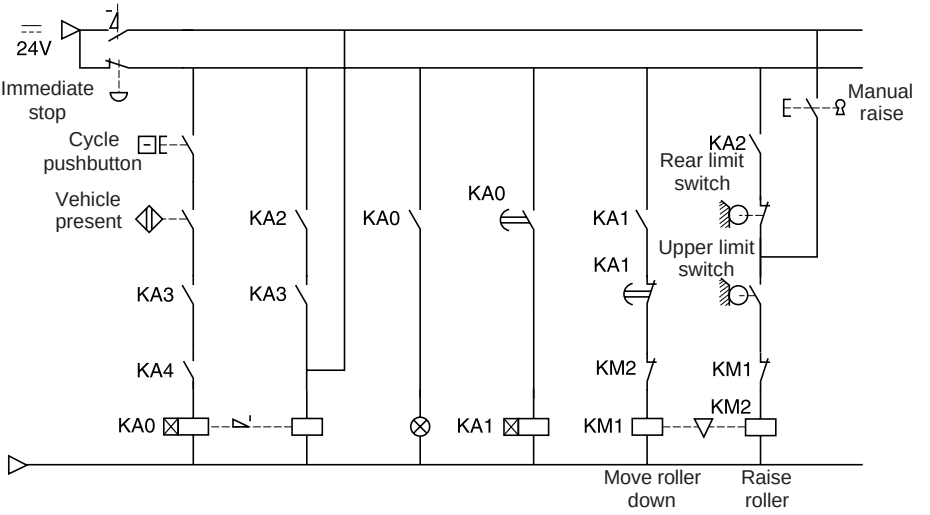


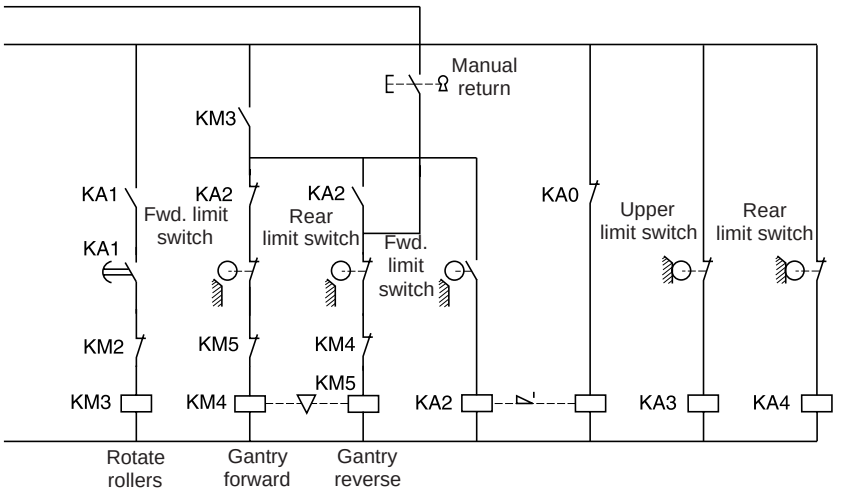
2.2 Control circuit

The diagram overleaf shows the automatic operation and manual stop. The real-time clock and totalling counter are not shown in this diagram.

Two control relays with mechanical latching memories (KA0 and KA2) memorize the stage reached in the cycle. This means that after a power outage, the cycle continues from the state where it stopped. The two time delays are provided by two time-delay options on KA0 and KA1.

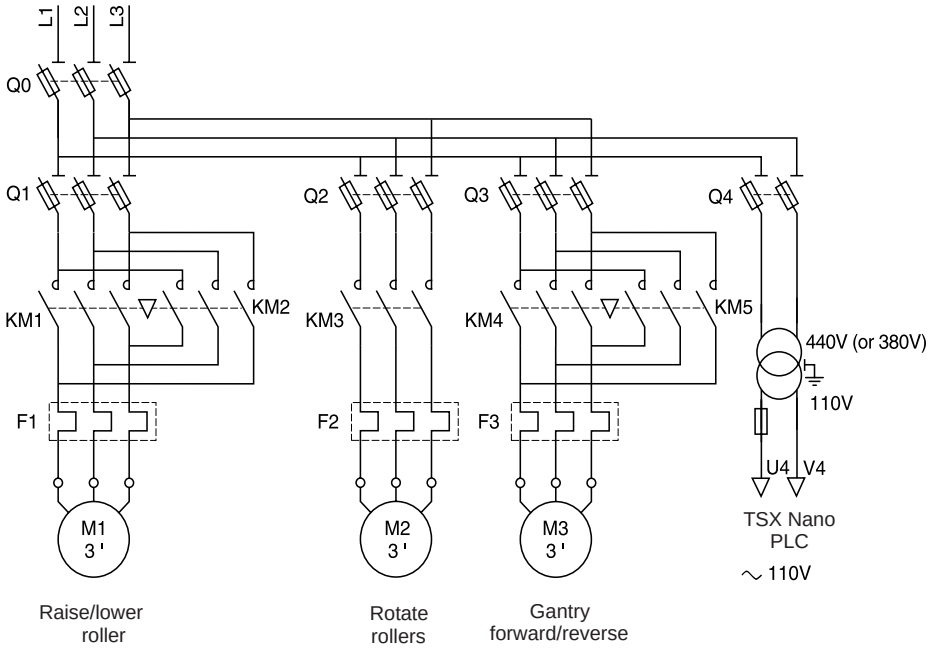
The rear and forward limit switches require two additional control relays (KA3 and KA4) due to the large number of contacts used in the layout.


24V
Immediate stop
Cycle pushbutton
Vehicle present
KA3
KA4
KA0
KA2
KA3
KA0
KA0
KA1
KA1
KA1
KA2
Rear limit switch
Upper limit switch
KM2
KM1
KM2
KM1
Manual raise
Move roller down
Raise roller

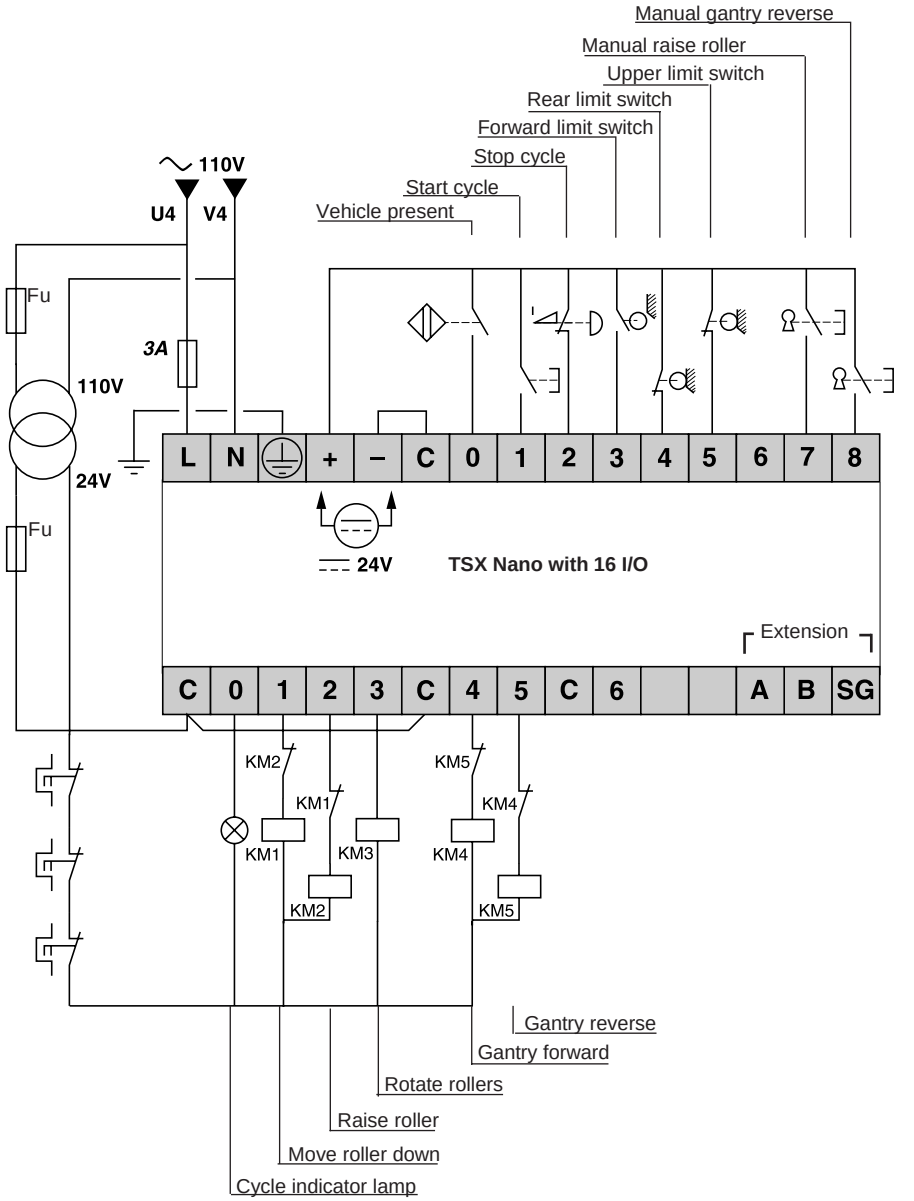

Manual return
KM3
KA1
KA1
KM2
KM3
KA2
Fwd. limit switch
KM5
KM4
KA2
Rear limit switch
KM5
KA2
Fwd. limit switch
KM4
KA0
KA2
Upper limit switch
KA3
Rear limit switch
KA4
Rotate rollers
Gantry forward
Gantry reverse

3.1 Power circuit

This layout is identical to that for the hard-wired logic solution. The 110VAC supply to the TSX Nano PLC is provided via a 440VAC(or 380 VAC)/110VAC transformer. The supply voltage to the outputs is 24VAC and provided by a second transformer.



3.2 PLC connection diagram



E

3.3 Parts list

3.3-1 PLC inputs

Item	Address	Description
Vehicle present	%I0.0	*Vehicle proximity sensor*
Start cycle	%I0.1	*Start pushbutton*
Stop	%I0.2	*Stop pushbutton*
Forward limit switch	%I0.3	*Gantry forward limit switch*
Rear limit switch	%I0.4	*Gantry rear limit switch*
Upper limit switch	%I0.5	*Roller upper limit switch*
Manual raise	%I0.7	*Manual raise roller pushbutton*
Manual return	%I0.8	*Manual gantry reverse pushbutton*

3.3-2 PLC outputs

Item	Address	Description
Indicator lamp	**%Q0.0**	**Cycle indicator lamp**
KM1	%Q0.1	Move roller down contactor
KM2	%Q0.2	Raise roller contactor
KM3	%Q0.3	Rotate rollers contactor
KM4	%Q0.4	Gantry forward contactor
KM5	%Q0.5	Gantry reverse contactor

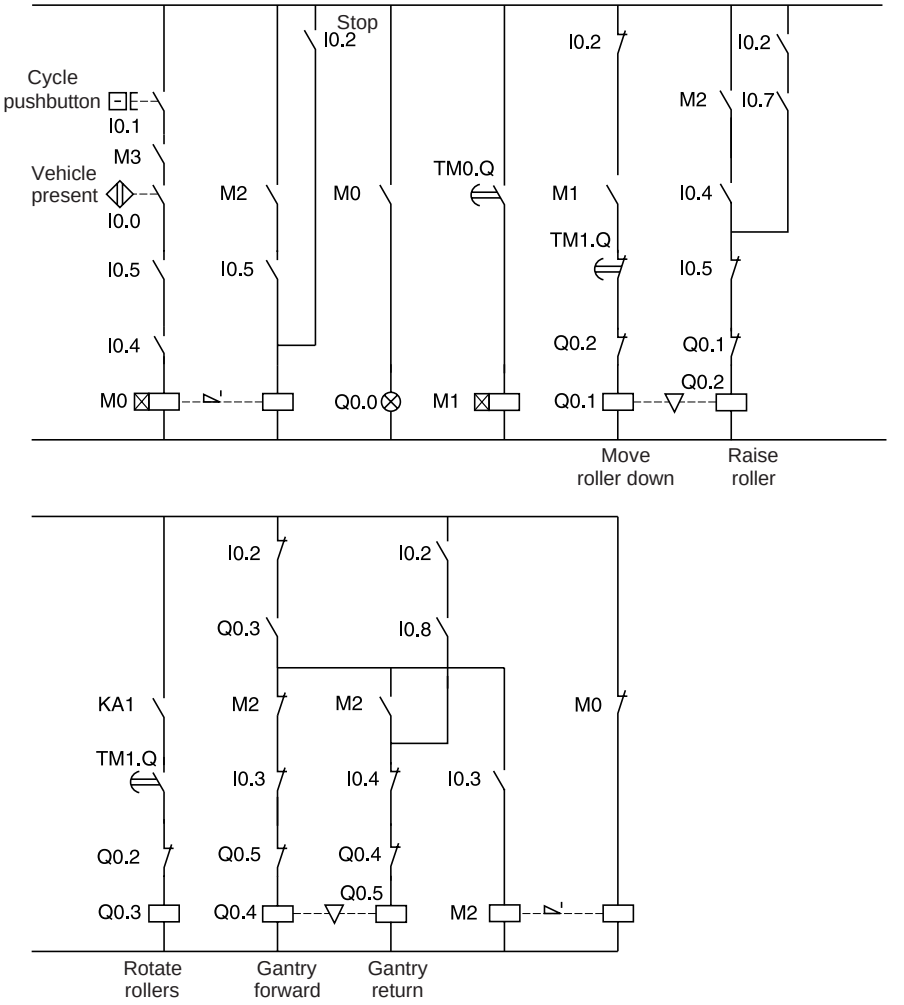
E

3.3-3 PLC internal variables

Type	Address	Description
Internal bit	%M0	Start latch variable (KA0)
Internal bit	%M1	Move roller down variable (KA1)
Internal bit	%M2	Gantry forward latch variable (KA1)
Internal bit	%M3	Schedule block output variable
Internal bit	%M4	Monday test variable
Internal bit	%M5	Variable to create pulse on %M4
Internal bit	%M6	Variable to create pulse on %M
Internal bit	%M7	Variable to test timer output on %TM1
System word	%MW0	Wash number totalling counter
System word	%SW50	Real-time clock current seconds/day
Timer function	%TM0	Start time delay
Timer function	%TM1	Move roller down time delay
Counter function	%C0	Weekly wash counter
Schedule block function	RTC0	Schedule block

3.4 Equivalent electrical scheme

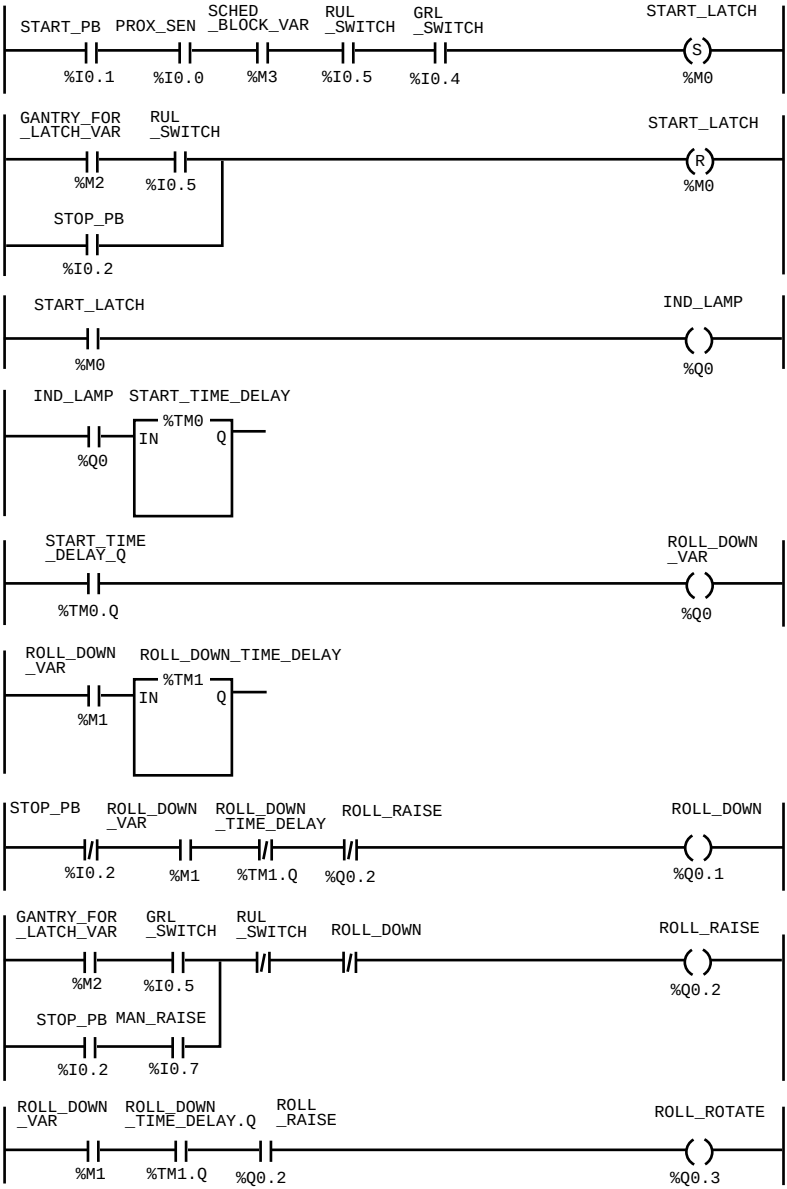
The circuit diagram modified for the PLC solution, which is directly based on the control circuit of the hard-wired solution in Section 2.2, is as follows.

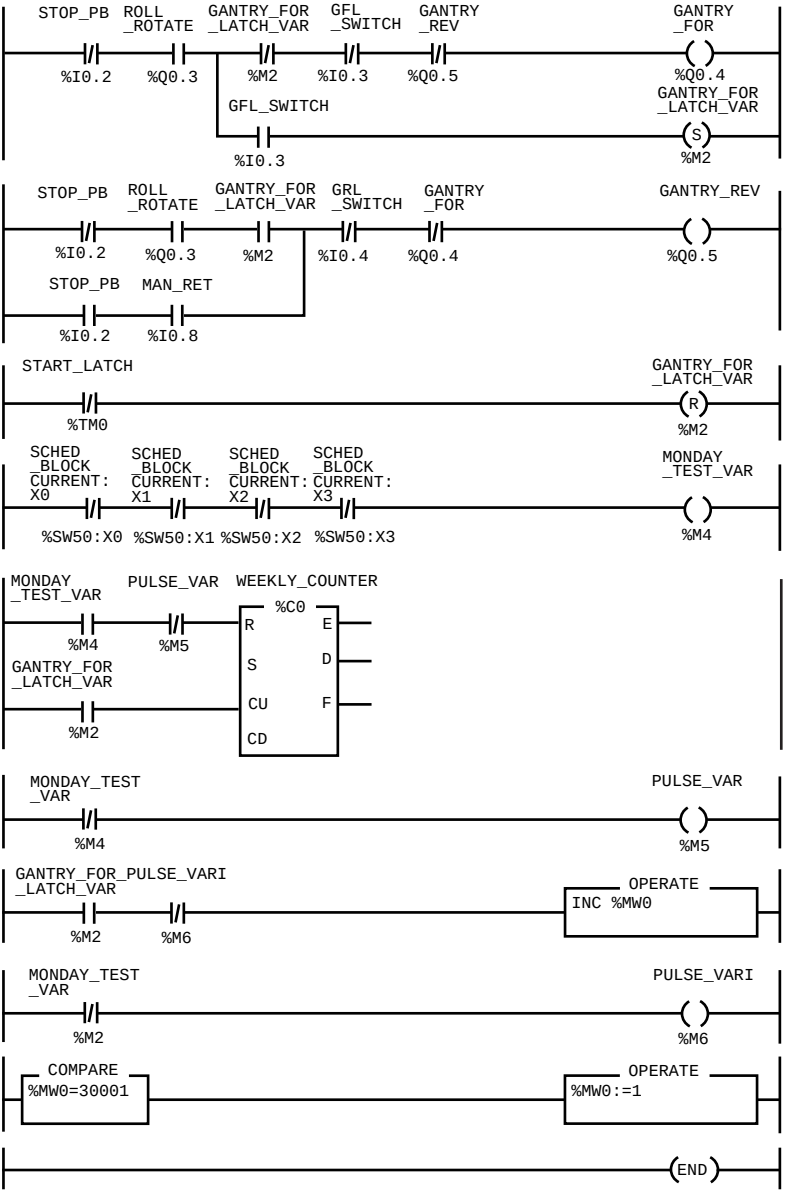


Control relays KA0, KA1 and KA2 are replaced by internal bits %M0, %M1 and %M2. Control relays KA3 and KA4 are not applicable, since the number of tests on a single PLC variable is not limited in a program.
The rising edge at internal bit %M4, set to state 1 every Monday (by testing the first four bits of system word %SW50) sets the weekly wash counter (counter %C0) to zero.
The totalling counter for the total number of washes (internal word %MW0) is automatically reset to 1 when it reaches 30 001.

3.5 Ladder program

The reversible Ladder program is as follows:

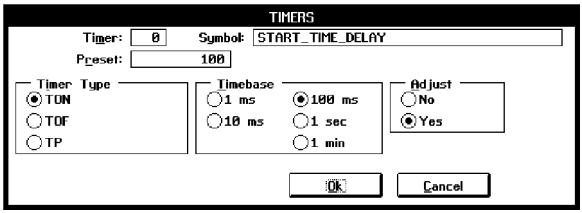



STOP_PB
%I0.2
ROLL _ROTATE
%Q0.3
GANTRY_FOR _LATCH_VAR
%M2
GFL _SWITCH
%I0.3
GANTRY _REV
%Q0.5
GANTRY _FOR
%Q0.4
GFL_SWITCH
%I0.3
GANTRY_FOR _LATCH_VAR
S
%M2
STOP_PB
%I0.2
ROLL _ROTATE
%Q0.3
GANTRY_FOR _LATCH_VAR
%M2
GRL _SWITCH
%I0.4
GANTRY _FOR
%Q0.4
GANTRY_REV
%Q0.5
STOP_PB
%I0.2
MAN_RET
%I0.8
START_LATCH
%TM0
GANTRY_FOR _LATCH_VAR
R
%M2
SCHED _BLOCK CURRENT: X0
%SW50:X0
SCHED _BLOCK CURRENT: X1
%SW50:X1
SCHED _BLOCK CURRENT: X2
%SW50:X2
SCHED _BLOCK CURRENT: X3
%SW50:X3
MONDAY _TEST_VAR
%M4
MONDAY _TEST_VAR
%M4
PULSE_VAR
%M5
GANTRY_FOR _LATCH_VAR
%M2
WEEKLY_COUNTER
%C0
R
S
CU
CD
E
D
F
MONDAY_TEST _VAR
%M4
PULSE_VAR
%M5
GANTRY_FOR_PULSE_VARI _LATCH_VAR
%M2
%M6
OPERATE
INC %MW0
MONDAY_TEST _VAR
%M2
PULSE_VARI
%M6
COMPARE
%MW0=30001
OPERATE
%MW0:=1
END

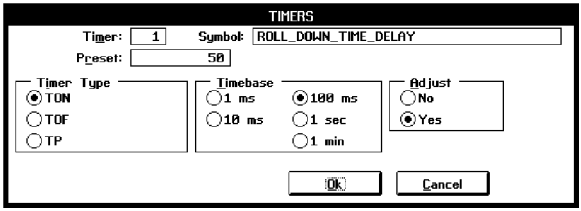
3.6 Function block configuration

The function blocks can be configured for timers %TM0 and %TM1and counter %C0 from the Configuration menu.

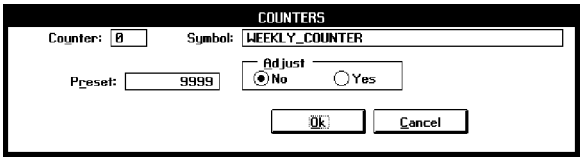
- Start time delay %TM0



- Move roller down time delay %TM1



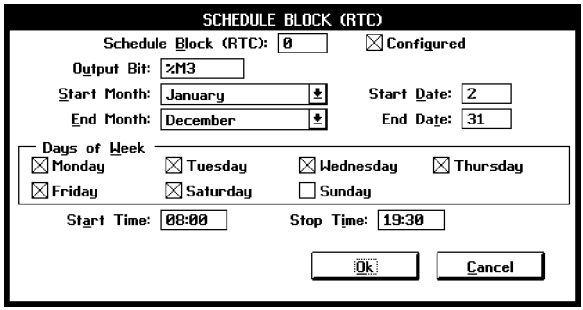
- Weekly wash counter %C0



3.7 Programming the RTC schedule block

The schedule block RTC0 is configured from the Configuration menu. This programming corresponds to the opening of the car wash :

- From 2 January to 31 December
- Monday,Tuesday, Wednesday, Thursday, Friday, and Saturday
- From 8:00 to 19:30.



E

F

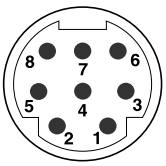
F

1.1 Characteristics of the terminal port

- **Characteristics**

Type of link	: RS485
Protocol	: UNI-TE format V2 (V1 format for TSX Nano ≤ V3.1).
Baud rate	: 9600/19200 bits/s 1 start bit, 8 bit data, odd parity, 1 stop bit
Type of connector	: 8-pin rapid connect mini-DIN
Max. link distance	
FTX117	: 10 m
UNITELWAY devices	: 10 m (without junction box)
ASCII devices	: 10 m

- **Connector pinout - UNI-TELWAY devices**



1	D+
2	D-
3	Not connected
4	/DE
5	/DPT
6	Not connected
7	0 V
8	5 V

The signal /DPT is used to select the operating mode of the terminal port :

/DPT = 1 UNI-TELWAY Master mode

/DPT = 0 UNI-TELWAY Slave mode or ASCII mode.

Connect pins 5 to 7 to select ASCII mode.

F

Notes :

Equipment should be connected with the power off (except for programming terminals).
The use of ASCII and UNI-TELWAY devices is exclusive.
An ASCII device must be disconnected to allow the use of a programmer (FTX117 or PC).

1.2 General information about the terminal port

The TSX Nano terminal port can operate in three different modes :

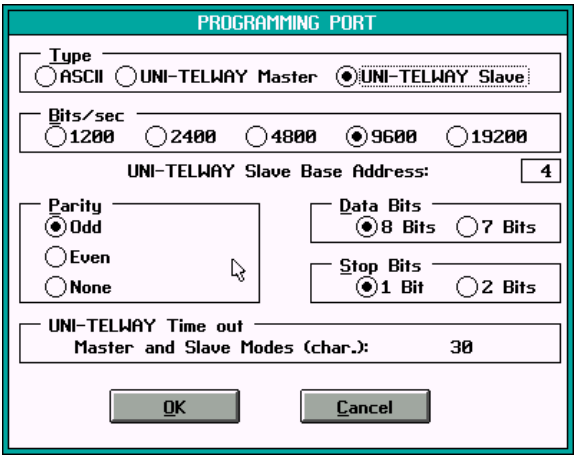
- UNI-TELWAY Master (TSX 07 2. TSX 07 3.),
- UNI-TELWAY Slave (TSX 07 3. only),
- ASCII (TSX 07 2. TSX 07 3.).

The operating mode of the terminal port is selected by software configuration and using signal /DPT (pin n°5) on the Mini DIN socket :

- When signal /DPT is at 1 (pin n°5 not connected), the terminal port is in UNI-TELWAY Master mode.
- When signal /DPT is at 0 (pin n°5 connected to pin n°7=0V), the terminal port is in ASCII mode or UNI-TELWAY Slave mode. The selection is made by software configuration using the programming tools PL707 and FTX 117 (ASCII mode by default).

The state of /DPT is shown by system bit %S100.

Configuration screen :



F

1.3 TSX Nano in ASCII mode

This character mode, simplified on the TSX Nano is used to transmit (TSX 07 2. and TSX 07 3.) and/or receive (TSX 07 3. only) a character string to/or from a simple device (printer or terminal) without flux control.

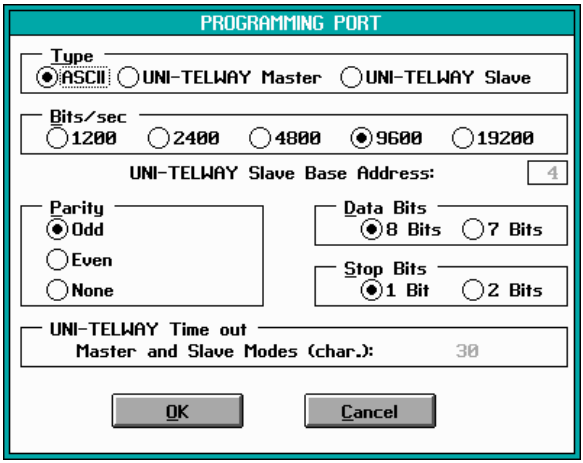
This mode is designed to operate in a point-to-point link.

The terminal port configuration can be modified on TSX 07 3. modifications are made in the PLC configuration screen :

- Type : Half-Duplex
- Speed : 1200, 2400, 4800, 9600 or **19200 bds**
- Format : 1 start bit, 7 or **8 data bits**, **1** or 2 **stop bits**
- Parity : even, **odd**, none.

The values in **bold** are the default values, they cannot be modified on TSX 07 2..

ASCII configuration screen :



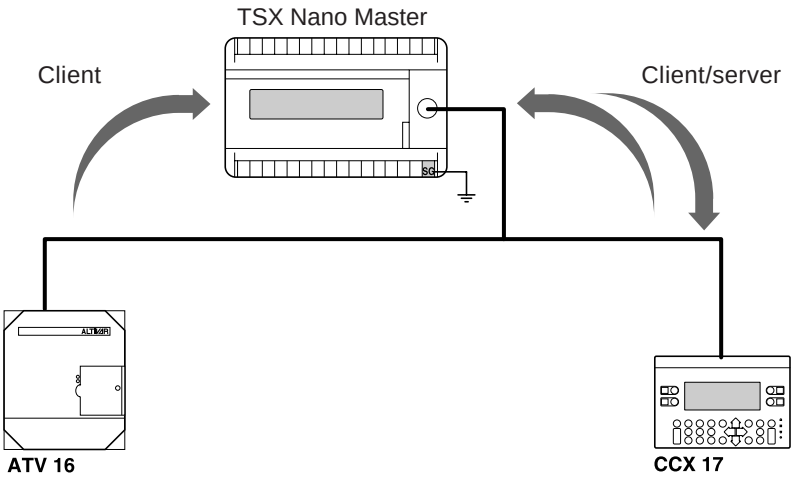
Programming in ASCII mode is possible using the EXCH instruction, as well as the exchange control function block %MSG. See section 3.4-6, part B.

F

1.4 TSX Nano Master on UNI-TELWAY

The TSX Nano is compatible with other UNI-TELWAY devices such as man-machine interfaces (MMI), identification devices and drives. This section provides general guidelines for connecting these devices on the UNI-TELWAY bus. Refer to the user manuals for the devices to obtain specific information.
When the TSX Nano is the UNITELWAY communication master for slave devices which are connected to the TSX Nano terminal port, it controls the network and polls the slave devices at periodic intervals.

Example of architecture



The TSX Nano is normally a UNI-TE server. However it can provide limited UNI-TE client services for TSX 07 2• :

• Configurations

Most UNI-TELWAY devices require the use of two addresses. The first address is configured physically by the user using microswitches; the second address is generally the physical address +1. The TSX Nano can communicate with a programming terminal and up to two additional UNI-TELWAY devices.

Address assignment :

0	:	TSX07 (communication master)
1	:	programmer (FTX117 hand-held terminal or PL7-07 software)
2-3-5	:	client device only (TSX 07 2•)
4	:	client and/or server device (TSX 07 2•)
2-3-4-5	:	client and/or server devices (TSX 07 3•)

Summary of devices that can be connected to various addresses.

Device	Reference	Permissible Addresses 1	2,3	4,5
Programming terminal		Yes	No	No
MMI		No	Yes	Yes
Identification devices		No	Yes	Yes
Speed controller		No	Yes (1)	Yes

(1) TSX Nano V3 only

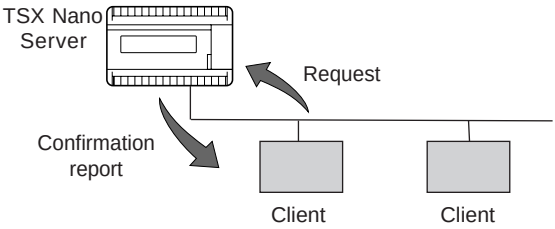
The terminal port configuration cannot be modified in UNI-TELWAY Master mode. It manages 5 slave link addresses depending on the transmission format as follows :

- Type : Half-Duplex
- Speed : 9600 bds
- Format : 1 start bit, 8 data bits, 1 stop bit
- Parity : odd.

The Time-out parameters can be set in the configuration screen.

• UNI-TE server function :

The TSX Nano responds to commands initiated by the client. A client is an intelligent device which can initiate communication with the TSX Nano. It can write data to or read data from the TSX Nano.



Addressing

The TSX Nano is always address 0 (master).

- The TSX Nano polls for addresses 1 to 5
- The programmer (either the FTX117 or the PL707 software) is always address 1
- Generally, the address of a UNI-TELWAY device is set by microswitches at a junction box or by cable connection.

F

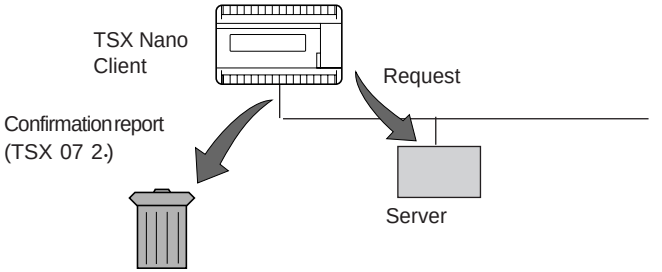
Notes :

- Communication between slaves is not possible when the bus master is a TSX Nano.
- The TSX 07 2• cannot be a UNI-TELWAY slave.

All the bus devices can interrogate the TSX Nano server system by using the target address 0.254.0. The TSX Nano UNITE server only responds to frames sent to this address. Messages containing a different target address will be ignored.

• TSX Nano as a client :

The TSX Nano initiates communication with the server. The server is an intelligent device which executes the commands sent by the TSX Nano.



The TSX 07 3• Master can send a request to any slave with an address between 1 and 5 using the EXCH instruction. It uses source address 0.254.16.

The TSX 07 2• can only send a request to slave at address 4 (using the EXCH instruction). Consequently, only WRITE and UNSOLICITED DATA (unconfirmed) requests may be used. Source address 0.254.10 is used.

F

WARNING :

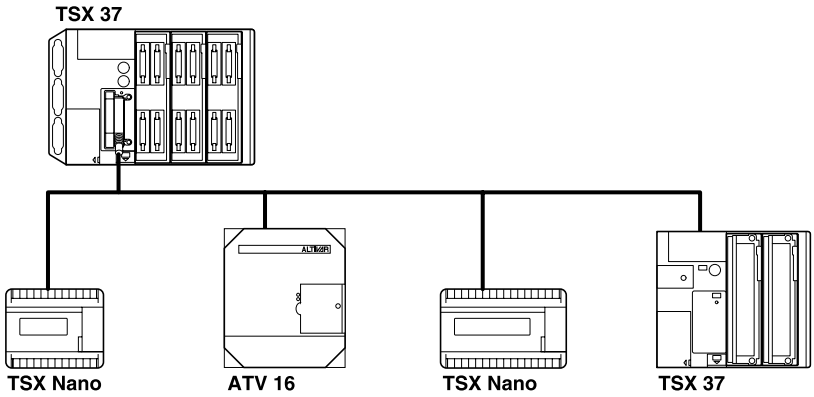
The UNI-TE client function of the TSX 07 3• reverses the data transmitted (most significant/ least significant) in relation to TSX 07 2•. Applications which operate with TSX 07 2• must be modified to take account of this reversal if they are loaded onto a TSX 07 3•.

Programming in UNI-TELWAY Master mode is possible using the EXCH instruction, as well as the exchange control function block %MSG. See section 3.4-6, part B.

1.5 TSX Nano Slave on UNI-TELWAY

This protocol is only available with TSX Nano version 3 or later (TSX 07 3.).
It is used for simultaneous, multidrop connection of several devices (PLC, programming terminal, MMI device, speed controller, etc.).

Example of architecture



Unlike Master mode, the terminal port configuration can be modified by the user in UNI-TELWAY slave mode using the programming tools PL7 07 or FTX 417 in the PLC configuration screen :

- Type : Half-duplex
- Speed : 1200, 2400, 4800, 9600 or **19200 bds**
- Format : **1 start bit**, 7 or **8 data bits**, **1** or 2 **stop bits**
- Time-out : **30** to 255 (see section 1.6)
- Address : **4** (1 to 97)
- Parity : even, **odd** or none.

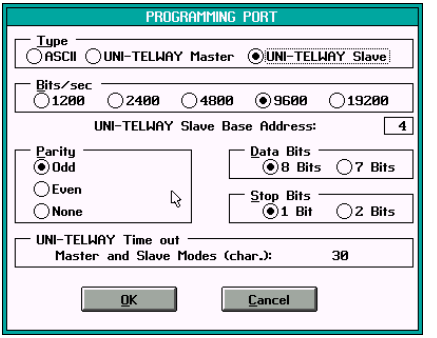
The values in **bold** are the default values.

The TSX Nano uses 2 consecutive logic addresses :

- AD0 : base address (that of the configuration), known as server address. Any device (local or remote) can address the TSX Nano system server using AD0 as target address.
- AD1 = AD0 + 1 known as client address. With this address the TSX Nano can transmit a request to any UNI-TELWAY bus device (Master or Slave) using the EXCH instruction.

See section 3.4-6 part B.

F



The modification of terminal port characteristics made in the configuration screen is only taken into account on a power restart or on modification of the signal /DPT on the Mini DIN socket (disconnection / reconnection of the terminal port).

1.6 UNI-TELWAY Time-out (TSX 07 3.)

The link layer of the UNI-TELWAY protocol (Master or Slave) uses a time-out.
This Time-out corresponds to the duration of the transmission of a number of characters sent on the line :

- TSX 07 3.: 30 characters by default. This duration is increased to 125 characters if the TSX Nano does not have an application.

Once a frame has been transmitted, a device (Master or Slave) declares a Time-out. If no acknowledgment is received before the end of the time-out, the exchange is not acknowledged.
The transmitter repeats its frame as soon as the protocol allows.

The terminal port configuration screen is used to configure a Time-out of between 30 and 8000 characters.
Values between 30 and 250 inclusive correspond to an equivalent number of characters. Values between 251 and 255 inclusive correspond to the following values :

- 251 = 500 characters
- 252 = 1000 characters
- 253 = 2000 characters
- 254 = 4000 characters
- 255 = 8000 characters.

On power-up or on modification of the signal /DPT, the configured value is loaded into the least significant bit of system word %SW14 (the most significant bit is ignored). Thus, the Time-out value can be fine-tuned by writing the new value in the system word %SW14. It will be taken into account at the end of the PLC scan.

A longer Time-out allows the TSX Nano to be connected to slow devices, for example modem type.

> It is possible to reduce the value of the Time-out to 10 characters by using system word %SW14.

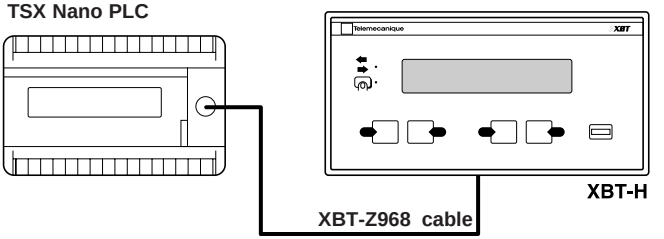
Important :

When connecting certain devices (eg modems) to a TSX Nano via the UNI-TELWAY link, their response times mean that the time-out parameters must be set to value 255 (approximately 8 s at 9600 bd/s).
This value prevents all subsequent communication between the TSX Nano and a programming terminal using the UNI-TELWAY link with standard parameters.
It is therefore necessary to :
- modify the value of the AUTOSPEED parameter (0 instead of 1) and the number of UNI-TELWAY addresses (5 instead of 3) in the DUNTLW.001 file under DOS, Windows 3.1/95/98 or using the XWAY tool under Windows NT.
- modify the timeout value of the PLC UNI-TELWAY link,
- reassign the initial values in DUNTLW.001 under DOS, Windows 3.1/95/98 or using the XWAY tool under Windows NT.

1.7 XBT operating terminals or CCX 17 operator panels

- **PLC connection**

The XBT operating terminal or CCX 17 operator panel is connected via the TSX Nano PLC terminal port using cable XBT Z968.



- **Characteristics of the link**
  - RS 485 link
  - UNI-TE protocol
- **XBT terminals which can be connected via the TSX Nano PLC terminal port**
  - XBT-A8 •
  - XBT-B8 •
  - XBT-BB8 •
  - XBT-C8 •
  - XBT-K8 •
  - XBT-M8 •
  - XBT-A8 •
  - XBT - H
  - XBT - P
  - XBT - E
- **CCX 17 operator panels which can be connected via the TSX Nano PLC terminal port**
  - TCCX 17 20 F
  - TCCX 17 20 FW
  - TCCX 17 20 FPS
  - TCCX 17 20 L
  - TCCX 17 20 LW
  - TCCX 17 20 LPS
  - TCCX 17 30 L
  - TCCX 17 30 LW
  - TCCX 17 30 LPS
- **Setup :**

  - XBT operator terminal :
  (see technical documentation relating to the XBT being used)

  - TSX Nano / XBT exchanges
  The PL7 program sends messages to the XBT terminal via :
  - EXCH instruction : to transmit messages
  - %MSG function block : to control data exchanges

  (see Part B, Section 3.4-6 : transmitting messages and controlling data exchanges).
- **Comments**
  XBT and TSX 07 data syntax (1)

Data	XBT or CCX17 syntax	TSX Nano syntax
Internal bit	Bi	%Mi
Internal word	Wi	%MWi

(1) except XBT - H/P/E.

F

1.8 UNI-TE requests supported by TSX Nano (server)

Standard requests

Service	Request	Request		Confirm		Meaning
		Hex.	Dec.	Hex.	Dec.	
Data (read)	Read a bit	00	00	30	48	Read a %Mi bit.
	Read a word	04	04	34	52	Read a %MWi word.
	Read objects	36	54	66	102	Read objects (%Mi, %Mi:L, %Mwi, %Mwi:L).
Data (write)	Write a bit	10	16	FE	254	Write a %Mi bit
	Write a word	14	20	FE	254	Write a %MWi word.
	Write objects	37	55	FE	254	Write objects (%Mi, %Mi:L, %Mwi, %Mwi:L).
Operating modes	RUN	24	36	FE	254	Set device to RUN.
	STOP	25	37	FE	254	Set device to STOP.

Special requests

Service	Request	Request		Confirm		Meaning
		Hex.	Dec.	Hex.	Dec.	
Data (read)	Read a system bit	01	01	31	49	Read a %Si bit
	Read a constant word	05	05	35	53	Read a %KWi word
	Read a system word	06	06	36	54	Read a %SWi system word
	Read Grafcet steps	2A	42	5A	90	Read %Xi Grafcet steps
Data (write)	Write a system bit	11	17	FE	254	Write a %Si bit
	Write a system word	15	21	FE	254	Write a %SWi word

F

F

2.1 Characteristics of the MODBUS / JBUS extension port

The extension port of the TSX 07 3● can be used to connect other TSX Nano PLCs as I/O extension (see part A, section 1.4) or even as peer PLC (part A, section 1.10). From TSX 07 3● , this extension port allows a MODBUS Slave type link.

Characteristics of the MODBUS link:

- Physical layer: RS485 non isolated, maximum length 200 meters
- Link layer: Asynchronous transmission
  ASCII frame (7 bits) or **RTU (8 bits)**
  Data rate: 1200, 2400, 4800, **9600** or 19200 bds
  Parity: **Even,** odd or none
  Number of stop bits: **1** or 2
  Inter-character time: **3** (1 to 127) characters
- Physical configuration: 28 devices max.
  98 logic addresses (**1** to 98)
- Services: Bits: 128 bits on request
  Words: 120 words of 16 bits on request
  Safety: on control parameter (CRC16) for each frame (in RTU).

The values in **bold** are the default values.

Note: Devices should be connected with the power off.

F

2.2 MODBUS / JBUS on TSX Nano

2.2-1 General

The data exchange services offered (%Mi and %MWi) are common to MODBUS and JBUS, which allows MODBUS devices to communicate with JBUS devices.

The MODBUS / JBUS protocol allows data exchange between a Master and Slaves, it does not allow direct communication between Slave devices, nor network transparency.

MODBUS communication in ASCII mode:

- Operation in ASCII mode is designed for connecting simple devices (screens, etc). The frame is complete but the frames are twice as long as in RTU mode.

Detail of an ASCII frame:

Header(3A) 1 byte	Slave no. 2 bytes	Function code 2 bytes	Data 2n bytes	LRC 2 bytes	CR 1 byte	LF* 1 byte

*The system word %SW67 can be used to set the parameters of the end delimiter character (LF). It is written to 16#000A by the system on a cold start.
The user can modify this system word by program or by adjustment when the Modbus Master uses an end delimiter character other than 16#000A.

MODBUS communication in RTU mode:

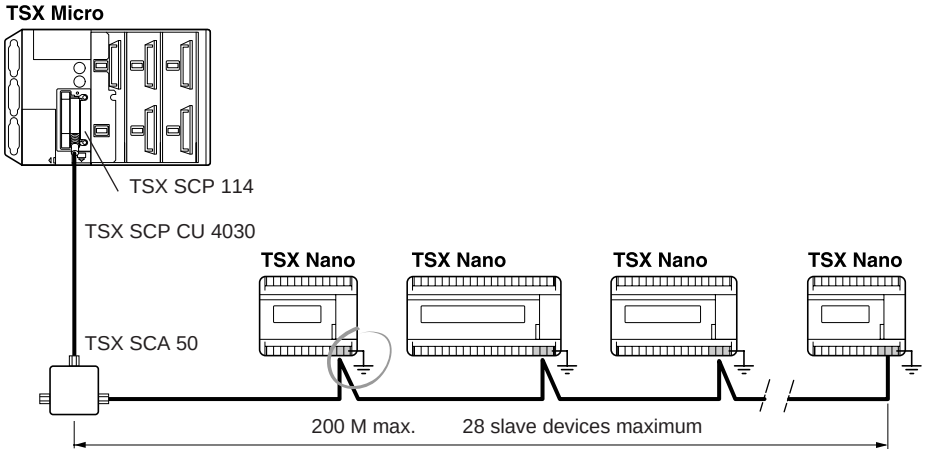
- A frame in RTU mode does not have a header byte nor an end of message delimiter byte:

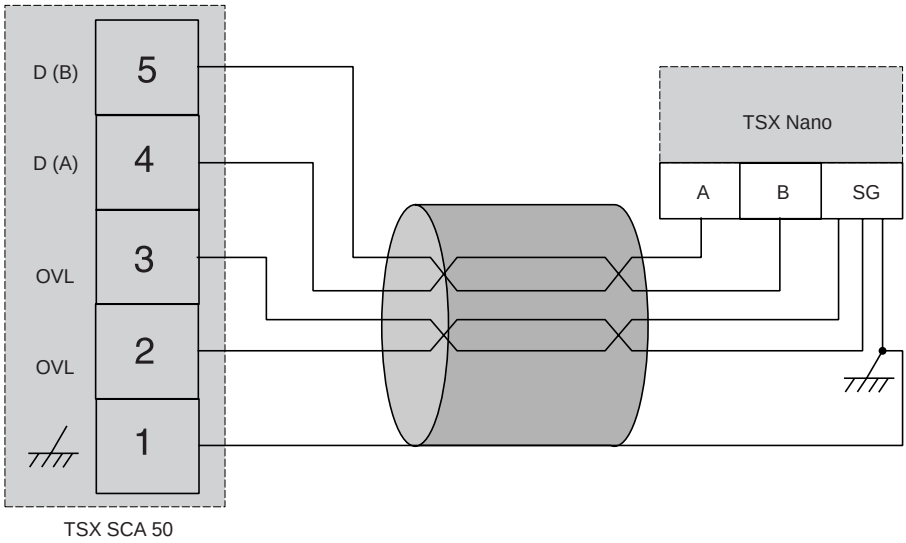
Slave no. 1 byte	Function code 1 byte	Data n bytes	CRC16 2 bytes

F

Example of connection

By daisy chaining

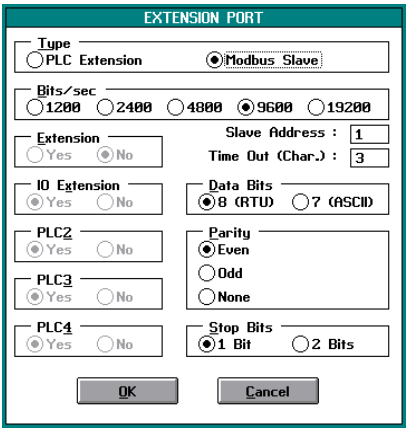




F

2.2-2 Configuring the MODBUS link

The extension link in MODBUS / JBUS is configured from the **Configuration** menu of the PL7 07 by selecting **Extension port.**
The following dialog box appears:



In dynamic operation, the parameters (speed and format) cannot be modified.
There is no mechanism for adapting the speed of the TSX Nano Slave in relation to the speed of the Master.
Any modification made to the speed and protocol format is taken into account by the TSX Nano as soon as the configuration screen is enabled.
The **Extension, I/O Extension, AP2, AP3 and AP4** fields are not significant if MODBUS protocol is selected.
See part C section 5.19 for more details.

F

2.2-3 Requests supported by the TSX Nano in MODBUS

The TSX Nano supports the requests listed below, others will be rejected with the exception code **01: Function not known:**

The MODBUS function only processes one request at a time, as in MODBUS protocol the Master must wait for a response from the slave or for a time-out to be triggered before sending a new request to the slave.

Standard MODBUS requests :

- 01 / 02 : Read n consecutive internal bits : Access to bits %M0 to %M127
- 05 : Write an internal bit : Access to bits %M0 to %M127
- 15 : Write n consecutive internal bits: Access to bits %M0 to %M127
- 03 / 04 : Read n consecutive words : Access to words %MW0 to %MW255
- 06 : Write an internal word: Access to words %MW0 to %MW255
- 16 : Write n consecutive words: Access to words %MW0 to %MW255.

Details of these requests are given in section 2.3.

Access requests to the UNITE server of the TSX Nano:

- 0F : Identification
- 4F : Read CPU
- 24 : RUN
- 25 : STOP
- 33 : INIT

Details of these requests are given in section 2.4.

Exception codes:

An exception code is returned by the slave when it does not know how to process the request.
The response frame thus contains:

- The function code received, incremented by the value 16#0080
- The exception code which depends on the type of error.

The two exception codes processed by the TSX Nano are:

- 01: Function not known (request not supported by the TSX Nano)
- 03: Invalid data (bit or word number not managed by the TSX Nano, writing a bit with a value other than 16#FF00 or 16#0000 etc).

F

2.2-4 Managing the COM LED

Once the response to a request has been sent, the TSX Nano switches on the communication LED for 50ms.
The LED then flashes, the frequency of the flashes depends on the exchanges between the Master and the TSX Nano .

2.2-5 Associated bits and system words

When a MODBUS request is processed, the TSX Nano sets the system bit %S70 to 1. This bit can be used by the user. Reset to 0 is the responsibility of the user.

The system word %SW67 can be used to set the parameters of the end of frame delimiter character (LF) in ASCII mode.
It is written to 16#000A by the system on a cold start.
The user can modify this system word by programming or by adjustment when the Master uses an end of frame delimiter character other than 16#000A.

2.3 Standard MODBUS requests

These requests can be used to exchange MODBUS objects by accessing objects %Mi and %MWi of the TSX Nano.
The requests are detailed below in RTU mode. In ASCII mode, the data is the same, CRC 16 is replaced by LRC.

2.3-1 Reading n internal bits %Mi

Function 01 or 02

Question :

Slave number	1 or 2	No. of 1st bit		Number of bit		CRC 16
		PF	Pf	PF	Pf	
1 byte	1 byte	2 bytes		2 bytes		2 bytes

Response :

Slave number	1 or 2	Number of bytes read	Value	•••••••••	Value	CRC 16
1 byte	1 byte	2 bytes				2 bytes

Example : reading bit %M3 from Slave 2

Question	02	01	0003	0001	CRC 16
Response	02	01	01	xx	CRC 16

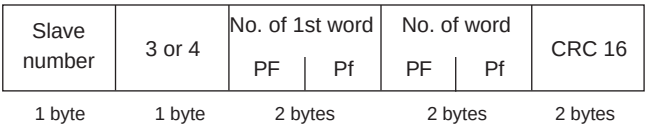
xx: 00 if %M3 = 0; 01 if %M3 = 1

F

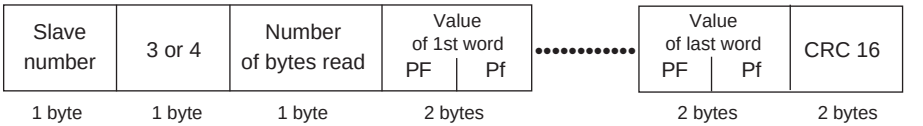
2.3-2 Reading n internal words %MWi

Function 03 or 04

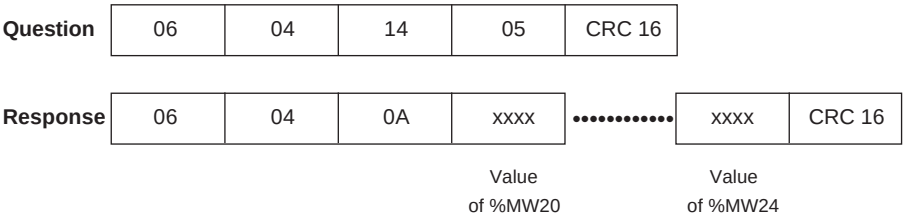
Question :



Response :



Example : reading words %MW20 to %MW24 from Slave 6



F

2.3-3 Writing an internal bit %Mi

Function 05

Question :

Slave number	5	No. of bit PF	Pf	Value of bit	CRC 16
1 byte	1 byte	2 bytes		2 bytes	2 bytes

The "Value of bit" field has two possible values to the exclusion of all others :

- bit at 0 = 0000,
- bit at 1 = FF00.

Response :

Slave number	5	No. of bit PF	Pf	Value of bit	CRC 16
1 byte	1 byte	2 bytes		2 bytes	2 bytes

Example : writing the value 1 in the bit %M3 of Slave 2

Question	02	05	03	FF00	CRC 16
Response	02	05	03	FF00	CRC 16

F

2.3-4 Writing an internal word %MWi

Function 06

Question :

Slave number	6	No. of word		Value of word		CRC 16
		PF	Pf	PF	Pf	
1 byte	1 byte	2 bytes		2 bytes		2 bytes

Response :

Slave number	6	No. of word		Value of word		CRC 16
		PF	Pf	PF	Pf	
1 byte	1 byte	2 bytes		2 bytes		2 bytes

Example : writing the value 16#3A15 in the word %MW12 of Slave 5

Question	05	06	0C	3A15	CRC 16
Response	05	06	0C	3A15	CRC 16

2.3-5 Writing n internal bits %Mi

Function 15

Question :

Slave number	0F	Address of 1st bit to write	Number of bits to write	Number of bytes	Value of bits to write	CRC 16
1 byte	1 byte	2 bytes	2 bytes	1 byte	n bytes	2 bytes

Response :

Slave number	0F	Address of 1st bit written	Number of bits written	CRC 16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

2.3-6 Writing n internal words %MWi

Function 16 (H'10')

Question :

Slave number	10	Number of 1st word	Number of words	Number of bytes	Value of 1st word	•••••••••••	CRC 16
1 byte	1 byte	2 bytes	2 bytes	1 byte	2 bytes		2 bytes

Response :

Slave number	10	Number of 1st word	Number of words	CRC 16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

Example : Writing values 1 and 2 to words %MW16 and %MW17 of Slave 11

Question	0B	10	0010	0002	04	0001	0002	CRC 16

Response	0B	10	0010	0002	CRC 16

2.3-7 Calculation of LRC

LRC: sum in hexadecimal, modulo FF, of the contents of the frame, excluding headers, two's complemented and coded in ASCII.

Example of frame

ASCII frame

3A	30	31	30	38	30	30	30	30	36	31	36	32	33	34	0D	0A

Equivalent binary frame

01	08	00	00	61	62	4872

Calculation of LRC

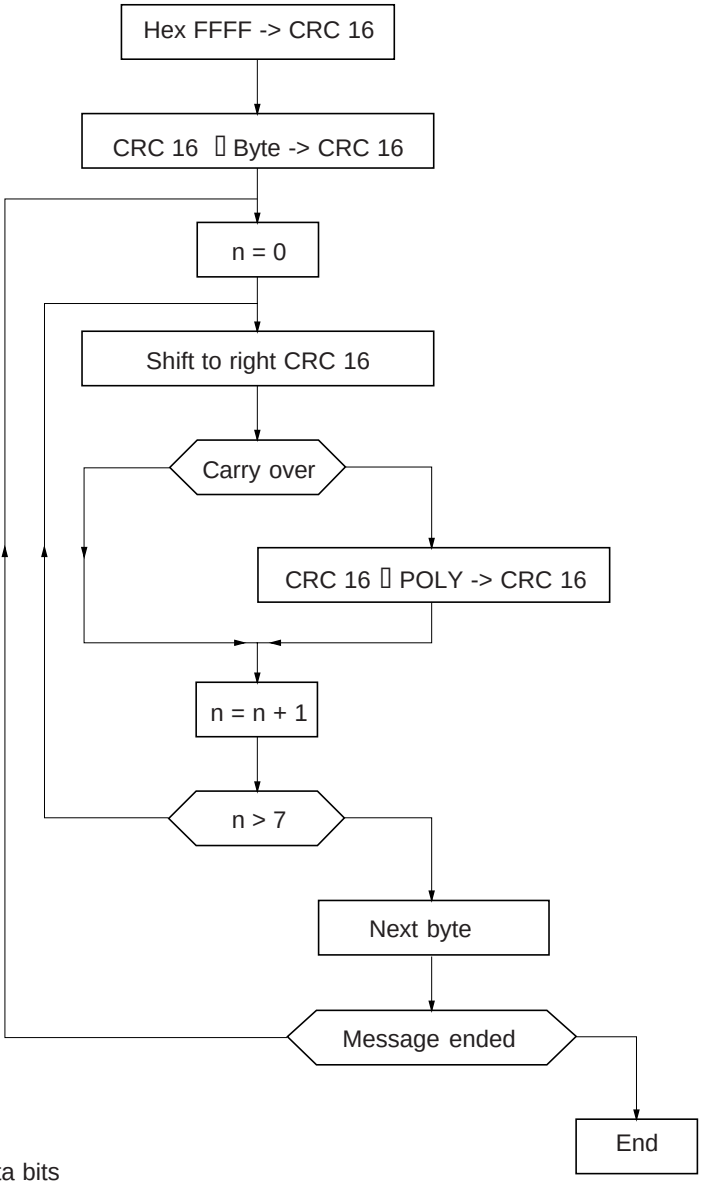
Sum in hexa, modulo FF, of the contents of the frame:
$01 + 08 + 00 + 00 + 61 + 62 = CC_H$ = 1100 1100

one's complement:	0011	0011
add 1:		+ 1
	0011	0100
Hexadecimal conversion	3	4
Coding in ASCII	33	34

LRC = 3334

F

2.3-8 Algorithm for calculating CRC 16



⊕ = or exclusive
n = Number of data bits
POLY = polynomial CRC 16 calculation = 1010 0000 0000 0001
(polynomial generator = 2 = $x^2 = x^{15} = x^{16}$)
In CRC 16, the 1st byte transmitted is the least significant.

F

2.4 Access requests by the TSX Nano to the UNITE server

The principle consists of coding a UNITE request within a MODBUS request in order to interrogate the UNITE server normally servicing the terminal port.
This mechanism uses the specific function code: 65.

Principle:

MODBUS Master request to Slave

Function code: 65	Request code	Category code	Data

MODBUS Slave response to Master

Function code: 65	Response code	Response data

The category code is always 7.

The UNITE response code can be used to find out the result of the operation performed by the server.
There are three cases :

- Request response code = Request code + 30 H: The operation has been performed by the server, additional information gives exact details of the result of the operation.
- Positive response code = 16#00FE: The operation has been performed correctly and no additional information is transmitted in the response.
- Negative response code = 16#00FD: The server has not been able to perform the operation (request not known, out of range value etc.).

F

2.4-1 Identification

This service can be used to provide identification and structuring information on the device which is targeting a request to the UNITE server.
The Identification request also provides basic diagnostic information, by showing the state of the LEDs and the device and module Status.

Request code: 0F
Response code: 3F

Report format: 27 bytes

- **Identification type** (byte): Byte always equal to H'FF'.
- **Product range** (byte): This parameter identifies the product range to which the device belongs: H'0B' = TSX Nano range.
- **Version** (byte): Version number of the device, coded in two BCD 4-bit bytes: H'30' for TSX07 30/31.
- **ASCII string** (character string): Specifies the product reference.
  On the TSX Nano it consists of one byte (H'10') followed by ASCII text (16 bytes) as follows:
  'TSX 07 3'
  '1-' if schedule block is present, '0-' if schedule block is not present
  '10 ' if 10 I/O, '16 ' if 16 I/O, '16AC' if 16 I/O AC, '24 ' if 24 I/O
  H'20'
  H'20'
- **Device status** (byte): Byte always equals 0 for the TSX Nano
- **State of LEDs** (8 bit table): This byte indicates the state of the 4 indicator LEDs of the TSX Nano:
  RUN (bits 0 and 1), ERR (bits 2 and 3), I/O (bits 4 and 5), COM (bits 6 and 7).
  Each LED is coded on 2 bits:
  00: Unlit
  01: flashing
  10: Lit
  11: Not significant.
- **Base module type** (byte): Byte equal to H'30' (type = CPU)
- **Manufacturer's reference** (byte): Byte equal to H'09'
- **Catalogue reference** (byte): Defines the type of device:
  H'01': Module 10 I/O
  H'11': Module 16 I/O
  H'21': Module 24 I/O
  H'12': Module 16 I/O AC

- **Base module status** (8 bit table): The values of these fields meet the standard for configurable devices. A bit positioned at 1 indicates a fault.

Bit no.	Class	Comments
0	ERR-INT	1: TSX Nano hardware fault
1	ERR-EXT	1: Sensor supply fault
2	Reserved	
3	MOD-At	1: Module in selftest
4	Reserved	
5	ERR-CNF	1: Extension configured but absent or NOK
6	Reserved	
7	Reserved	

- **Number of components** (byte): Byte equal to 0 because the module does not have any submodules.

2.4-2 Read-CPU

This service is used to diagnose the state of a TSX Nano PLC processor.

Request code: 4F
Response code: 7F

Format of report (14 bytes):

- **Extension** (byte): This field is used to transmit a transaction number: Not significant.
- **State of LEDs** (8-bit table): see Identification request.
- **PLC status** (8-bit table): Describes the physical state of the PLC:
  Bit 0: RUN (1), STOP (0)
  Bit 1: Application executable (1)
  Bit 2: Cartridge present (always 1)
  Bit 3: Forcing in progress (1)
  Bit 4: Reserved: 0
  Bit 5: Software fault (1)
  Bit 6: Reserved: 0
  Bit 7: Reserved: 0
- **Address of the reserver** (6 byte table): Describes the network address of the application entity which has reserved the processor. On the TSX Nano, this value is fixed at 16#00FF for each byte, to indicate that the processor is not reserved.

- **Type of application error (byte):** Byte always at 0.
- **Debug information** (8-bit table):
  Bit 0: State of forcing (1: Active, 0: No forcing)
  Bits 1 to 7: Always at 0.
- **Product range** (byte): This parameter identifies the product range to which the device belongs: H'0B' = TSX Nano range.
- **Application / PLC information** (8-bit table)**:**
  Bit 0: Application present in RAM (1)
  Bit 1: Program in RAM with checksum OK (1)
  Bit 2: Program in RAM executable (1)
  Bit 3: Program in RAM protected (1)
  Bit 4: Program in RAM differs from EEPROM program (1)
  Bit 5: Application compatible with OS (1)
  Bit 6: Schedule block present (1)
  Bit 7: 0
- **Backup information** (8-bit table):
  Bit 0: Application present (1)
  Bit1: Checksum OK (1)
  Bit 2: EEPROM application executable (1)
  Bit 3: EEPROM application protected (1)
  Bit 4: Auto load application (1)
  Bit 5: Application compatible with OS (1)
  Bit 6: 0
  Bit 7: 0

F

2.4-3 RUN request

Request code: 24
Response code: FE

Negative result if:
- RUN / STOP input configured in STOP
- Application cannot be executed
- Software fault (watchdog overrun, launching nonexistent G7 step).

Sending this request to a TSX Nano already in RUN is not considered an error.

2.4-4 STOP request

Request code: 25
Response code: FE

Negative result if the application in RAM cannot be executed.

A STOP instruction to a TSX Nano already in STOP is not considered an error.

2.4-5 INIT request

Request code: 33
Response code: 63

- **Type of initialization** (byte to transmit): Must always be at 1.
- **Report**: Indicates the result of the initialization operation:
  00: Initialization OK
  01: Type of initialization other than 1

2.5 Limitations

The TSX Nano Slave on MODBUS has the following limitations:

- **Physical layer:** RS485 only. Does not automatically adapt to the speed of the Master.
- **Selector position:** The MODBUS protocol is only enabled if the position of the PLC rotary selection switch (automatically read on power-up) shows that TSX Nano is being used **as a PLC** (positions 0, 5, 6 and 7).
- **Protocol:** The protocol is Slave only.
- **Communication between Slaves:** Direct communication between Slaves is not possible. It can only be performed via the application program of the Master.
- **Changing, dynamically, from ASCII mode to RTU mode** is not possible.

G

Section **Page**

G

A.1 Boolean instructions in List language with Ladder equivalent

LIST Instruction	LADDER Equivalent	Description
LD, LDN, LDR, LDF		Load
ST, STN, R, S		Store
AND, ANDN, ANDR, ANDF		Logical Bit 'AND'
OR, ORN, ORR, ORF		Logical Bit 'OR'
AND(, OR((nest 8 levels)		Parentheses
XOR, XORN, XORR, XORF		Exclusive 'OR'
N	Not Reversible	Logical 'NOT'
END, ENDC, ENDCN ENDCN is not reversible		End of Program
%Li		Label Definition
JMP, JMPC, JMPCN JMPCN is not reversible		Jump to Label 0≤ i <16
SRn		Subroutine Call
RET		Return
MCR, MCS		Master Control Relay
NOP	Not Reversible	No Operation

A.2 Tool bar and instruction bar menu options

In the following tables, the **Keys** refer to the keystrokes necessary to step through the menus. The **Quick Keys** are used to shortcut the menu to activate the desired function.

A.2-1 Configuration Editor menu options

Button	Description	Keys	Quick Keys
	Application Name	Alt + C, N	None
	Timers	Alt + C, T	None
	Counters	Alt + C, C	None
	Constants	Alt + C, O	None
	LIFO/FIFO Registers	Alt + C, R	None
	Drum Controllers	Alt + C, D	None
	Fast Counter	Alt + C, F	None
	%PLS/%PWM	Alt + C, P	None
	Input Filter	Alt + C, I	None
	Latch Input	Alt + C, A	None
	Run/Stop	Alt + C, U	None
	PLC Status (Security)	Alt + C, L	None
	Scan Mode	Alt + C, D	None
	Terminal port	Alt + C, I	None
	Extension Port	Alt + C, X	None
	Schedule Block	Alt + C, S	None
None	Change PLC Version	Alt + C, H	None
	Validate Configuration	Alt + T, L	None
	Cancel Configuration	Alt + T, C	None

G

	Help	None	F1

A.2-2 Symbols Editor tools menu options

Button	Description	Keys	Quick keys
	Insert	Alt + T, I	Insert key
	Delete	Alt + T, D	Delete key
	Sort by Address	Alt + T, A	None
	Sort by Symbol	Alt + T, S	None
	Find	Alt + E, F	Ctrl + F

A.2-3 Ladder Editor instruction bar options

Button	Description	Keys	Quick keys
	Help text window	F1	None
	Contact	F2	B.2.1-3
	Negated Contact	F3	B.2.1-3
	Rising Edge Contact	F4	B.2.1-3
	Falling Edge Contact	F5	B.2.1-3
	Horizontal Connector	F6	None
	Down Connector	F7	None
	Erase Down Connector	F8	None
	Horizontal Connector fill	F9	None
	Compare Block	F10	B.3.1-5
	Coil	Shift + F2	B.2.1-4
	Negated Coil	Shift + F3	B.2.1-4

G

A.2-3 Ladder Editor instruction bar options (continued)

Button	Description	Keys	Reference
	Reset Coil	Shift + F4	B.2.1-4
	Set Coil	Shift + F5	B.2.1-4
	Jump or Subroutine Call	Shift + F6	B.2.4-3, B.2.4-4
	Operate Block	Shift + F7	B.3.1-3 - .1-9
	Timer function block	Shift + F8	B.2.2-3
	Counter function block	Shift + F9	B.2.2-4
	Extended Ladder Palette	Shift + F10	None

A.2-3.1Extended Ladder Palette options

Button	Description	Keys	Reference
XOR	Exclusive OR	Alt + X	B.2.1-7
XORN	Exclusive OR negation	Alt + O	B.2.1-7
XORR	Exclusive OR rising edge	Alt + R	B.2.1-7
XORF	Exclusive OR falling edge	Alt + F	B.2.1-7
OPEN	Open contact	Alt + P	None
SHORT	Short contact	Alt + H	None
%Ri	LIFO/FIFO register function block	Alt + G	B.2.2-5
%SBRi	Shift bit register function block	Alt + B	B.3.4-7
%DRi	Drum controller function block	Alt + U	B.2.2-6
%SCi	Step counter function block	Alt + N	B.3.4-8
%FC	Fast counter function block	Alt + A	B.3.4-5
%PLS	Pulse generator function block	Alt + L	B.3.4-4

A.2-3.1 Extended Ladder Palette options (continued)

Button	Description	Keys	Reference
%PWM	Pulse width modulation	Alt + W	B.3.4-3
%MSG	Message block	Alt + K	B.3.4-6
RET	Return to program from subroutine	Alt + T	B.2.4-4
END	Unconditional end of a program	Alt + E	B.2.4-1
MCS	Master control start	Alt + S	B.2.4-5
MCR	Master control relay	Alt + M	B.2.4-5
-(#)-	Deactivate current step without activating any other step	None	B.2.3-1
-(#i)-	Activate step i after deactivating current step	None	B.2.3-1
-(#D)-	Deactivate step i and the current step	None	B.2.3-1

A.2-4 Ladder Editor Tools menu options

Button	Description	Keys	Quick Keys
None	Validate Program	Alt + T, V	None
	Validate Rung	Alt + T, R	Ctrl + Enter
	Cancel Rung	Alt + T, C	None
	New Rung	Alt + T, N	Ctrl + A
	Clear Rung	Alt + T, L	None
	Previous Rung	Alt + T, P	Ctrl + Up
	Next Rung	Alt + T, X	Ctrl + Down
	Toggle Grid	Alt + T, G	None

A.2-5 Ladder Viewer Tools menu options

Button	Description	Keys	Quick Keys
None	Validate Program	Alt + T, V	None
	Insert Rung	Alt + T, R	Ins
None	Insert List	Alt + T, L	Ctrl + Ins
	Edit Current Rung	Alt + T, E	Ctrl + E
	Delete Current Rung	Alt + T, D	Ctrl + D
	Show Symbols	Alt + T, S	Ctrl + F2
	Show Addresses	Alt + T, A	Ctrl + F3
	1 Line Address or Symbol	Alt + T, 1	None
	3 Lines Address or Symbol	Alt + T, 3	None
	3 Lines Address and Symbol	Alt + T, N	None
	Toggle Rung Header	Alt + T, H	Ctrl + H
	Toggle Grid	Alt + T, G	None
	Toggle Ladder/List	Alt + T, T	None
	Show All As Ladder	Alt + T, W	None
	Table of Grafcet steps	Alt + T, F	None

A.2-6 List Editor instruction bar options

Button	Description	Keys	Reference
	Help	F1	None
	Load	F2	B.2.1-3
	Store	F3	B.2.1-4

G

A.2-6 List Editor instruction bar options (continued)

Button	Description	Keys	Reference
	AND	F4	B.2.1-5
	OR	F5	B.2.1-6
	Exclusive OR	F6	B.2.1-7
	Enable input / instruction	F7	
	Memory push	F8	B.2.1-10
	Begin function block	F9	B.2.2-2
	Call or declare subroutine	F10	B.2.4-4
	Jump	Shift + F1	B.2.4-3
	Input bit of module 0	Shift + F2	B.3.1-2
	Output bit of module 0	Shift + F3	B.3.1-2
	Memory bit	Shift + F4	B.3.1-2
	System bit	Shift + F5	B.3.1-2
	Timer variable	Shift + F7	B.2.2-3
	Memory read	Shift + F8	B.2.1-10
	Wires block outputs	Shift + F9	B.2.2-2
	Return from a subroutine	Shift + F10	B.2.4-4
	Label	Alt + F1	B.2.4-3
	Input word	Alt + F2	B.3.1-1
	Output word	Alt + F3	B.3.1-1

G

A.2-6 List Editor instruction bar options (continued)

Button	Description	Keys	Reference
	Memory word	Alt + F4	B.3.1-1
	System word	Alt + F5	B.3.1-1
	Constant word	Alt + F6	B.3.1-1
	Counter	Alt + F7	B.2.2-4
	Move top of stack to accumulator	Alt + F8	B.2.1-10
	End of function block	Alt + F9	B.2.2-2
	End of program	Alt + F10	B.2.4-1

A.2-7 Data Editor Tools menu options

Button	Description	Keys	Quick Keys
None	Validate Program	Alt + T, V	None
	Insert	Alt + T, I	Ins
	Delete	Alt + T, D	Del
	Add Previous Instance	Alt + T, P	Ctrl + Up
	Add Next Instance	Alt + T, N	Ctrl + Down
	Force 1	Alt + T, 1	None
	Force 0	Alt + T, 0	None
	Clear Force	Alt + T, C	None
	Clear All Force	Alt + T, L	None
	Read Retained Values	Alt + T, R	None

G

A.2-7 Data Editor Tools menu options (continued)

Button	Description	Keys	Quick Keys
	Write Retained Values	Alt + T, W	None
	Write Data Value	Alt + T, T	None
None	Open Data Page	Alt + T, O	None
None	Save Data Page	Alt + T, S	None
None	Save Data Page As	Alt + T, A	None
	Help	None	None
	Toggle Number Format	None	None
	Toggle Animation	None	None

A.2-8 Edit menu options

Edit menu options	Keys	Quick keys
Undo	Alt + E, U	Ctrl + Z
Cut	Alt + E, C	Ctrl + X
Copy	Alt + E, O	Ctrl + C
Paste	Alt + E, P	Ctrl + V
Find	Alt + E, F	Ctrl + F
Replace	Alt + E, R	Ctrl + R

A.3 PL7-07 PC programming software variables

Variable Type	Description	Read/Write/Force
Counters	(%C0-%C15)	
%Ci.P	Preset value	R,W (2)
%Ci.V	Current value	R,W
%Ci.E	Underflow output (empty)	R
%Ci.D	Preset output reached	R
%Ci.F	Overflow output (full)	R
Drum Controllers		(%DR0-%DR3)
%DRi.S	Current step number	R
%DRi.F	Full (last step reached)	R
Fast Counter	%FC	
%FC.P	Preset value	R,W (2)
%FC.V	Current value	R,W
%FC.S0	Threshold value S0	R,W
%FC.S1	Threshold value S1	R,W
%FC.TH0	Threshold bit 0	R
%FC.TH1	Threshold bit 1	R
%FC.F	Overflow output	R
Grafcet Step Bits	(%X1-%X62)	R
Input Bits	(%I0.0-%I0.13 and %I1.0-%I1.13)	R,F (3)
Input Words	(%IWj.0-%IWj.1 where j=0,1,2,3,4)	R,W

A.3 PL7-07 PC programming software variables (continued)

Variable Type	Description	Read/Write/Force
Constant Words	(%KW0-%KW63)	R
LIFO/FIFO Registers	(%R0-%R3)	
%Ri.O	Output word	R
%Ri.I I	Input word	R,W
%Ri.E Register empty	R	
%Ri.F Register full	R	
Memory Bits	(%M0-%M127)	R,W
Memory Words	(%MW0-%MW255)	R,W
Message	(%MSG)	
%MSG.E	Communication error output	R
%MSG.D	Communication done output	R
Pulse Generator	(%PLS)	
%PLS.P	Preset value	R,W (2)
%PLS.N	Number of pulses	R,W
%PLS.Q	Current pulse generation output	R
%PLS.D	Pulse generation done output	R
Pulse Width Modulation	(%PWM)	
%PWM.P	Preset period	R,W
%PWM.R	Ratio of the period	R,W
Output Bits	(%Q0.0-%Q0.9 and %Q1.0-%Q1.9)	R,W,F (3)

A.3 PL7-07 PC programming software variables (continued)

Variable Type	Description	Read/Write/Force
Output Words	(%QWj.0-%QWj.1 where j=o,1,2,3,4)	R,W
System Bits	(%S0-%S127)	R,W (1)
Shift Bit Register	(%SBRi.0-%SBRi.15, where i=0-7)	R
Step Counter	(%SCi.0-%SCi.255, where i=0-7)	R
System Words	(%SW0-%SW127)	R,W (1)
Timers	(%TM0-%TM31)	
%TMi.V	Current value	R
%TMi.P	Preset value	R,W (2)
%TMi.Q	Timer done	R

(1) Certain bits and system words cannot be written. No specific messages warn the user in the List/Ladder editor. The Write on these bits or words has no effect in the PLC.

(2) These variables can be written if the Adjust Option is selected in the configuration. If the Adjust Option is not selected, the Write access in the Data editor will display a message, "PLC operation not available".

(3) Forcing the Run/Stop input or PLC status (security) output is authorized by the programming tools, the input or output is signaled as being forced but the PLC does not take the forcing into account.

Note: For all the variables that are not allowed to Write, a message warns the user in the Data editor.

A.4 Security features

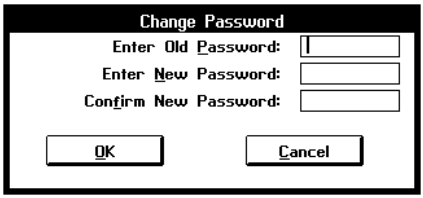
The security features for the PL7-07 PC programming software includes password protection to prevent unauthorized changes to an application and the ability to change security levels. When you define a password, you restrict access to an application by creating the following two levels of security:

• **Supervisor Level**—allows you to modify any part of the application.

• **Operator Level**—allows you to modify symbols and data pages, but does not allow you to modify the application program and configuration data. An application that is password-protected defaults to the operator level each time it is opened.

A.4-1 Entering a password for the first time

1. Open the application to be password protected.
2. Select **Security** from the File menu. Select **Change Password** from the Security sub-menu. The Change Password dialog box appears.



3. Do not enter any information in the **Enter Old Password** field. Tab to the **Enter New Password** field and type a password from 1 to 8 characters.
4. In the **Confirm New Password** field, re-type the same password.
5. Select **Ok** to use the new password. The application title bar displays the words, "Supervisor Level".
   Select **Cancel** to close the dialog box without defining the password.
6. The application must be saved for the new password to take effect.

A.4-2 Changing a password

1. Select **Security** from the File menu. Select **Change Password** from the Security sub-menu. The Change Password dialog box appears.
2. In the **Enter Old Password** field, enter the old password. Tab to the **Enter New Password** field and type a new password from 1 to 8 characters.
3. In the **Confirm New Password** field, re-type the same password.
4. Select **Ok** to use the new password. Select **Cancel** to close the dialog box without changing the password.
5. The application must be saved for the new password to take effect.

A.4-3 Changing from operator level to supervisor level

1. Select **Security** from the File menu. Select **Supervisor Level** from the Security sub-menu. The Security dialog box appears.

2. Type the password. Select **Ok** to change to the supervisor level. Select **Cancel** to return to the operator level.

A.4-4 Removing a password

1. To remove a password, select **Security** from the File menu. Select **Change Password** from the Security sub-menu.
2. At the Change Password dialog box, type in the current password in the **Enter Old Password** field.
3. Tab through the **Enter New Password** and **Confirm New Password** fields without entering any information.
4. Select **Ok** to remove the password. Select **Cancel** to return to the application with the password still in effect.

A.4-5 Password protection for files transferred to an FTX 117 terminal

The FTX 117 terminal only has keys for entering the alphanumeric characters 0 through 9 and A through F. If a file is transferred from a PC to the FTX 117 terminal with a password that has characters other than 0 through 9 and A through F, then the password will not be able to be entered. Changes to the file will not be possible from the FTX terminal.

A.5 Instruction scan time and memory usage

Times is expressed in µs.
Size is expressed in bytes.

Boolean instructions

Instructions		With bit operands		
	%I,%Q		%M, %S, %X, 0/1, %MWi:Xi... (1)	
	Time	Size	Time	Size
LD, LDN	0.2	2	0.4	4
LDR, LDF	0.5	4	-	-
AND, ANDN	0.2	2	0.6	4
ANDR,ANDF	0.8	4	-	-
AND(, AND(N	6.3	8	7	10
AND(R,AND(F	7	10	-	-
OR, ORN	0.4	2	0.7	4
ORR, ORF	0.8	4	-	-
OR(, OR(N	6.3	8	6.6	10
OR(R, OR(F	6.7	10	-	-
XOR,XORN	0.2	2	0.6	4
XORR,XORF	0.8	4	-	-
ST, STN (2)	0.2 (0.8)	2	0.9 (1.5)	4
S,R (2)	0.8 (1.2)	4	1.5 (1.8)	6
N	0.6	4		
)	15	8		
MPS	0.6	4		
MRD	0.2	2		
MPP	0.2	2		

These times must be multiplied by 3 when the instructions are written after program line 099.

(1) For word extract bits %MW16 to %MW255 and for all other types (%KWi:Xj,%SWi:Xj) these times are multiplied by 1.5 and the sizes are increased by 2 bytes.
(2) The times given in parentheses are the scan times for the instructions when the application has been initialized to use MCS/MCR instructions.
Summary **:** the choice of whether to use master relay instructions or not is made when the application memory is cleared (see Part C, Section 4.6 of the FTX 117 manual, or Part C, Section 5.21 of the PL7-07 manual).

Instructions on function blocks (in reversible programming)

Instruction	Scan time (in µs)	Memory usage (in bytes)
BLK %TMi	8	4
BLK %Ci	8	4
BLK %Ri	8	4
BLK %SBRi	8	4
BLK %SCi	8	4
BLK %DRi		
BLK %FC	8	4
BLK %MSG	8	4
BLK %PMW	8	4
BLK %PLS	8	4
OUT_BLK	200	2
END_BLK	180	2
IN	1.2	4
R	0.6	4
CU	0. 7	4
CD	0.7	4
I	1	4
O	1	4
U		
S	0.7	4

Instructions on function blocks (in non-reversible programming)

Instruction	Scan time (in µs)	Memory usage (in bytes)
IN %TMi	48	4
CD %Ci (CU %Ci)	46	4
S %Ci	49	4
R %Ci	47	4
U %DRi		
R %DRi		
LD %SCi.j	9	6
CD %SCi (CU %SCi)	38	4
ST %SCi.j	10	6
R %SCi	36	4
BLK %PMW	42	4
BLK %PLS	53	4
CD%SBRi(CU%SBRi)	39	4
R %SBRi	37	4
I %Ri (O %Ri)	49	4
R %Ri	48	4
IN %PWM	36	4
IN %PLS	46	4
S %PLS	42	4
R %PLS	58	4
IN %FC	43	4
S %FC	69	4
READ	9.8	6
EXCH	160 - 700	8
R %MSG	25	4

Numerical instructions

Instruction	Scan time (in µs)	Memory usage (in bytes)
:=	29.5	10
+	34	12
-	38	12
*	49	12
/	48	12
REM	49	12
INC	28	6
DEC	28	6
AND	37	12
OR	37	12
XOR	37	12
NOT	29	8
SHL	34	10
SHR	34	10
ROL	35	10
ROR	35	10
BTI	40	8
ITB	40	8
SQRT	80	8

Comparison instructions

Instruction	Scan time (in µs)	Memory usage (in bytes)
LD[word1 comp word2]	18	8
AND[word1 comp word2]	19	10
AND([word1 comp word2]	24	14
OR[word1 comp word2]	21	10
OR([word1 comp word2]	25	14

comp : comparison operations =>,<=,<>,=,>=,>

Grafcet instructions

Instruction	Scan time (in µs)	Memory usage (in bytes)
=*= i		4
#		4
#i		4
#Di		6
=*=POST		4
-*- i		6

Instructions on programs

Instruction	Scan time (µs)	Memory usage (in bytes)
END	0.4	2
ENDC, ENDCN	0.6	4
SR	14	4
RET	2	6
NOP	0.4	2
JMP	7.8	4
JMPC, JMPCN	8	6
%Ln :	0.6	4
%SRn :	2	4
MCR	0.5	2
MCS	2.5	12

A.6 Importing and Exporting ASCII program files and symbols files

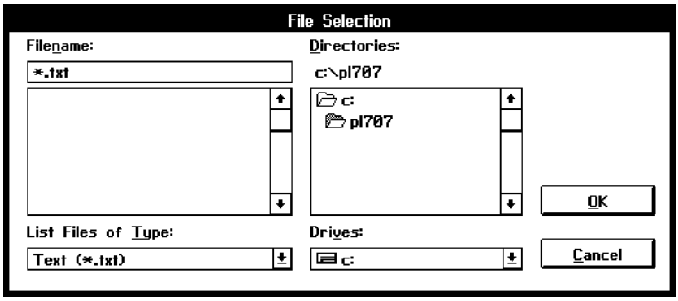
The PL7-07 PC programming software allows you to import and export ASCII program files and symbols files.

A.6-1 ASCII program files

List programs can be developed by external ASCII text editors. ASCII format program import and export allows you to transfer these files between the PL7-07 software and the ASCII text editor.

To import an ASCII program file:

1. With a new or existing application open, select **Import** from the File menu. Select **ASCII Program** from the Import sub-menu. The File Selection dialog box appears.



2. In the **Drives** field, select the drive where the text file is located.
3. In the **Directories** field, select the directory containing the text file to be imported.
4. In the **List Files of Type** field, select either Text (*.txt) or All (*.*) to display the files in the selected directory.
5. In the **Filename** field, select the file name of the text file to be imported.
6. Select **Ok** to import the file. Select **Cancel** to close the dialog box without importing the file.

To export an ASCII program file:

1. With an application open, select **Export** from the File menu. Select **ASCII Program** from the Export sub-menu. The File Selection dialog box is displayed.
2. In the **List Files of Type** field, select either Text (*.txt) or All (*.*). The .txt file extension identifies text editor files.
3. In the **Drives** field, select the drive where you want to store the file.
4. In the **Directories** field, select the directory where you want to save the file.
5. In the **Filename** field, over type the asterisk (*) with a standard DOS file name.

   If the file name does not comply with DOS file naming conventions, an "Invalid Filename" message is displayed.

If you type a file name that already exists in the selected directory, an error message appears: "The file selected already exists. Overwrite?" Select **Ok** or **Cancel**.

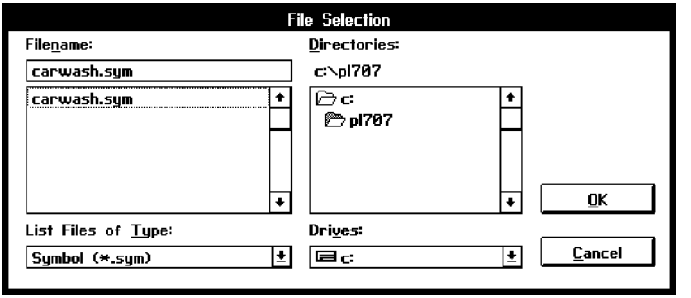
6. Select **Ok** to export the file. Select **Cancel** to close the dialog box without exporting the file.

A.6-2 Symbols files

Symbols files can be created for common applications. Symbols import and export allows you to transfer the symbols files between applications.

To import a symbols file:

1. With a new or existing application open, select **Import** from the File menu. Select **Symbols** from the Import sub-menu. The File Selection dialog box appears.



2. In the **Drives** field, select the drive where the symbols file is located.
3. In the **Directories** field, select the directory containing the symbols file to be imported.
4. In the **List Files of Type** field, select either Symbol (*.sym) or All (*.*) to display the files in the selected directory.
5. In the **Filename** field, select the file name of the symbols file to be imported.
6. Select **Ok** to import the file. Select **Cancel** to close the dialog box without importing the file.
7. If **Ok** is selected in step 6, a message appears in the status bar.

G

To export a symbols file:

1. With an application open, select **Export** from the File menu. Select **Symbols** from the Export sub-menu. The File Selection dialog box appears.
2. In the **List Files of Type** field, select either Symbol (*.sym) or All (*.*). The .sym file extension identifies symbols files.
3. In the **Drives** field, select the drive where you want to store the file.
4. In the **Directories** field, select the directory where you want to save the file.
5. In the **Filename** field, over type the asterisk (*) with a standard DOS file name.

   If the file name does not comply with DOS file naming conventions, an "Invalid Filename" message appears.

   If you type a file name that already exists in the selected directory, an error message appears: "The file selected already exists. Overwrite?" Select **Ok** or **Cancel**.
6. Select **Ok** to export the file. Select **Cancel** to close the dialog box without exporting the file.
7. If **Ok** is selected in step 6, a message appears in the status bar.

A.7 Transferring applications between a PC and the FTX 117 terminal

A.7-1 Application files

An application can be transferred from a PC to a FTX 117 terminal by the following ways:

- transfer the program from the PC to the Nano-PLC and then transfer from the PLC to the FTX 117 terminal.
- save the program in the PC to a binary file (.app extension), and transfer the file to the FTX 117 terminal via a PC memory card (T FTX REM 3216, T FTX RSM 3216, OR T FTX RSM 12816). The memory card must be formatted on the FTX 117 terminal.

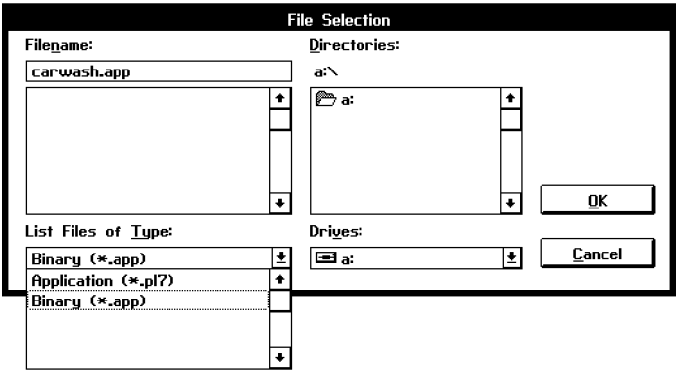
Note : When a program is saved and transferred to the FTX 117 terminal (as a binary file), symbols and comments are not saved or transferred.

Section 12, Transferring an application, details how to transfer from the PC to the PLC. The TSX 07 Nano PLC FTX 117 Terminal manual details how to transfer from the PLC to the FTX 117 terminal.

To transfer an application from a PC to a FTX terminal via a memory card requires the application to be saved as a binary file (.app extension). When the application is saved as a binary file, the symbols and comments are not saved. Therefore, if the application is later transferred back to the PC, the symbols and comments will not appear. Additionally, the word "DEFAULT" replaces the file name in the title bar. To re-name the file, select **Save As** from the File menu and save the file with the desired file name. For more information on Save As, see Section11.3.

To transfer an application from the PC to the FTX 117 terminal via a FTX 117 formattted memory card:

1. With the application open, select **Save As** from the File menu. A File Selection dialog box appears.



2. In the **List Files of Type** field, open the selection box and select the Binary (*.app) file type.
3. In the **Drives** field, select the memory card drive.

G

4. In the **Directories** field, select the directory where you want to save the file.
5. In the **Filename** field, over type the asterisk (*) with a standard DOS file name.

   If the file name does not comply with DOS file naming conventions, an "Invalid Filename" message is displayed.

   If you type a file name that already exists in the selected directory, an error message appears: "The file selected already exists. Overwrite?" Select **Ok** or **Cancel**.
6. Select **Ok** to save the binary file. Select **Cancel** to close the dialog box without saving the file.

To transfer an application from the FTX 117 terminal to the PC via a memory card:

1. Select **Open** from the File menu. A File Selection dialog box appears.
2. In the **List Files of Type** field, open the selection box and select the Binary (*app) file type.
3. In the **Drives** field, select the memory card drive.
4. In the **Directories** field, select the directory that contains the file to be transferred.
5. In the **Filename** field, select the file to be opened.
6. Select **Ok** to open the binary file. Select **Cancel** to exit the dialog box without opening the file.

The application transferred is now available on the PL7-07 software screen.

A.7-2 Data page files

Data page files (.dat extension) in the PC have the same file format as data table files in the FTX 117 terminal. However, a data table in the FTX 117 terminal contains 16 data objects whereas a data page contains 64 data objects. The PL7-07 PC programming software data page can hold up to 4 FTX 117 data pages.

A data page file is transferred to the FTX 117 terminal via a FTX 117 formatted memory card. Use Save Data Page As (Section14.4-16) and save the file to a memory card.

A.8 Troubleshooting installation problems

A.8-1 Installing PL7-07 PC programming software on a 5.25" drive

PL7-07 PC programming software is shipped on 3.5" diskettes. You can install the software on a computer with a 5.25" disk drive, if you have another IBM-compatible computer with both a 3.5" and 5.25" disk drive. You also need three to six formatted 5.25" diskettes, depending on the language selected.

1. On the computer with both the 3.5" and 5.25" disk drive, install installation **Disk #1** in the 3.5" floppy disk drive.
2. With the DOS prompt at **c:**, select the drive letter for the 3.5" floppy drive, this is typically **a:**.
3. At the **a:** prompt, type **install** and press <enter>.
4. Select the language for installing the program. The PL7-07 PC programming software can be installed in one of five languages—English, French, German, Italian, and Spanish.

   The PL7-07 PC programming software can operate in only one language at a time. If support for multiple languages is required, create a new directory with a different name and re-install the software in the new directory. If you re-install the software in the same directory as the original installation, the newly-installed version will overwrite the old one.
5. Select the disk drive where the program is to be installed.
6. Enter the disk destination subdirectory. The default subdirectory is \PL707.
7. Select the COM port to be used by the computer for communicating with the PLC.
8. The program files are automatically decompressed and written to the subdirectory specified.
9. When prompted by the installation program, insert installation **Disk #2**.
10. The installation program prompts you with the following:

    May I create/modify your AUTOEXEC.BAT file if needed (Y/N)? Select **No**.

    May I create/modify your CONFIG.SYS file if needed (Y/N)? Select **No**.
11. When prompted by the installation program, re-insert installation **Disk #1**.
12. After the installation is finished, the following information is displayed:

    Make sure the changes were made to CONFIG.SYS:

    FILES=30 (or higher)

    DEVICE=C:\PL707\DUNTLW.EXE PROFILE=C:\PL707\DUNTLW.001

and AUTOEXEC.BAT:

PATH=....;C:\PL707

Record this information. This information will be used to update the CONFIG.SYS and AUTOEXEC.BAT files in the destination computer with the 5.25" disk drive.

COM1 is the default communications port. To change the COM port setting, edit DUNTLW.001 in directory C:\PL707. In the Basic Parameters section, change the COM setting in the line PORT=COM1:0,8,1 to the desired COM port.

13. Remove the 3.5" installation diskette and insert a blank formatted 5.25" diskette in the appropriate drive.
14. Type **cd** to return to the root directory.
15. To transfer the installed software to the 5.25" diskettes, type **backup C:\PL707 b:/s** and press <enter>.

    Backup is a DOS command. For more information about this command, refer to your DOS instruction manual.
16. You are prompted for additional diskettes, as needed. Label or keep the diskettes in the order in which they are backed up.
17. After the backup is complete, take the diskettes to the destination computer and insert the first diskette in the 5.25" disk drive.
18. From the **c:** prompt, type **restore a: c:\ /s**. Follow the instructions displayed until all information is restored. This assumes that **a** is the drive designation for the 5.25" disk drive.
19. Add the information recorded in step 12 to the CONFIG.SYS and AUTOEXEC.BAT files.
20. Reboot the computer.

A.8-2 Software installation problems

If you encounter a problem during or after installing the PL7-07 PC programming software, check the following conditions.

1. Check that these PL7-07 PC programming software files are installed:

DIALOG.ZNC	ERRORS.MDX	PRINTERS.DBF
DUNTLW 001	HELP.ZNC	PRINTERS.DBT
DUNTLW.386	PAGES.DBF	PRINTERS.MDX
DUNTLW.EXE	PAGES.MDX	RESWORD.TBL
ERRORS.DBF	PL707.EXE	PL707.ICO
ERRORS.DBT	PL707.INI	

2. Do not install the PL7-07 PC programming software deep in the DOS directory tree structure, such as 10 levels down from the root directory.
3. If you encounter either memory or communications errors while running the PL7-07 software, you may need to reconfigure your memory manager or add lines to your CONFIG.SYS file. Use the following guidelines for reconfiguring EMM386.EXE, QEMM, and 386MAX.

A.8-2.1 EMM386.EXE

EMM386.EXE is part of MS-DOS, versions 5.0 and later. EMM386.EXE supports VCPI, but does not support DPMI.

You do not need to have EMM386.EXE installed. However, if you encounter EMM386.EXE error messages when running PL7-07 software and EMM386.EXE is installed, add the following line to your CONFIG.SYS file:

DEVICE=EMM386.EXE FRAME=NONE

The FRAME=NONE option leaves the EMS and VCPI interfaces active, but prevents EMM386.EXE from allocating any page frames.

Regardless of the version you are using, you must be careful when selecting EMM386 configuration options. Some options conflict with one another and choosing them can cause EMM386.EXE to conflict with Phar Lap's 286 | DOS-Extender.

One set of conflicting options is the NOEMS/FRAME=xxxx/RAM/Mx group. Each of these EMM386.EXE switches affects the creation of an EMS page frame. Do not use the options together, but rather, select only one option. Specifying FRAME=NONE is satisfactory for most users.

A.8-2.2 QEMM

QEMM, by Quarterdeck Office systems, supports VCPI, but does not support DPMI. We suggest that you add the following line to your CONFIG.SYS file:

DEVICE=QEMM.SYS FRAME=NONE

The FRAME=NONE option leaves the EMS and VCPI interfaces active, but prevents QEMM from allocating any page frames.

Do not specify the QEMM configuration option NOEMS, if you want to run Phar Lap's 286 | DOS-Extender. This option disables the VCPI interface, so that Phar Lap's 286 | DOS-Extender cannot communicate, resulting in the following error message:

Fatal Error 286.1020: This program requires VCPI or DPMI in V86 mode.

A.8-2.3 386MAX

386MAX, by Qualitas, supports both VCPI and DPMI. If you encounter an error message, we suggest you add the following line to your CONFIG.SYS file:

DEVICE=C:\386MAX\386MAX.SYS PROFILE=C:\386MAX\386MAX.PRO

where 386MAX.PRO is a profile file.

You can use either the NOFRAME or EMS=0 option without significantly affecting the performance of Phar Lap's 286 | DOS-Extender. The NOFRAME option keeps the EMS and VCPI interfaces active, but prevents 386MAX from allocating a 64 KB page frame for EMS. The EMS=0 option also keeps the EMS and VCPI interfaces active, while making all remaining memory available as extended (XMS) memory.

A.8-3 COM port problems during installation

In order for the PL7-07 PC programming software in your PC to communicate with the PLC, you must have serial communications capability between the PC and the PLC.

When the PL7-07 PC programming software (DOS or Windows 3.1/95/98) or the XWAY communication tool (Windows NT) is first installed on your computer, the software checks to see which serial COM ports (COM1 - COM4) are available. During the installation, a listing of the available COM ports is displayed and you are asked to select the desired COM port. If the word "None" is the only thing displayed, then no serial COM ports are available. The following are the likely problems and the suggested actions to take :

- There is no serial communications card in your computer. You need to install a serial communications card and assign a COM port designation.

or

- All of the existing COM parts are being used. The following choices are recommended :
- disable a device assigned to the COM port you would like to use for the PLC and assign the PLC the COM port.
  If installing PL7-07 under DOS or Windows 3.1/95/98, in the Basic Parameters section of the DUNTLW.001 file, ensure PORT=COM?:O,8,1 is set to the desired COM port, where ? is the COM port number.
  If installing PL7-07 under Windows NT, install the UNI-TELWAY driver.
- share the COM port with another device. However, the PLC and the other device cannot be operated at the same time. For example, if you share a COM port with a modem, the PLC cannot be connected to the PC when the modem is being used.
  If installing PL7-07 under DOS or Windows 3.1/95/98, in the Advance Parameters section of the DUNTLW.001 file, ensure COMSHARE=YES.
  If installing PL7-07 under Windows NT, the system sets this parameter automatically.
- install another serial communications card and assign a COM port designation.

A.8-4 Communication error when connecting the PC to the PLC

If a communication error is detected when you attempt to connect the PC to the PLC, an Error dialog box appears. Select **Ok** to acknowledge the Error dialog box.

1. Check to ensure the cable is properly connected between the PC and the PLC and that power is available to the PLC.
2. In the CONFIG.SYS file, ensure the following line is present:

   DEVICE=C:\PL707\DUNTLW.EXE PROFILE=C:\PL707\DUNTLW.001
3. In the CONFIG.SYS file, ensure FILES=30 (or higher)
4. Ensure there is no IRQ (interrupt) conflict between the PLC and any other device.
5. In the DUNTLW.001 file located in the c:\PL707 directory, ensure the proper port is designated in the line PORT=COM?:O,8,1, where ? is the COM port number.
6. If your computer uses EMM386, it may be necessary to add REM at the beginning of the line in the CONFIG.SYS file that designates EMM386 as a device. The REM prevents the line from being read.

A.8-5 Video Display problems

(Software version V1.0 only)

If you are experiencing video display problems such that the characters in the dialog boxes or the List Editor seem to be scrambled, it may be because your PC is using a different font than the standard IBM character font. You may have to adjust your CONFIG.SYS and AUTOEXEC.BAT files to use what is referred to by some PC manufacturers as the "Material Font" in order to fix the problem. This is accomplished by changing to the DOS code page.

For instance, changing the code page for DOS 6.2 in some PCs would be something like the following example.

Using a text editor, edit the CONFIG.SYS file to include this statement :

DEVICE = C:\DOS\DISPLAY.SYS CON: = (EGA,437,1)

Again using a text editor, edit the AUTOEXEC.BAT file to include the following statements :

C:\DOS\NLSFUNC
MODE CON CP PREPARE = ((850) C:\DOS\EGA.CPI)
CHCP 850

G

For more information on changing the code page, consult the DOS documentation that came with your PC.

A.9 Running PL7-07 in Windows

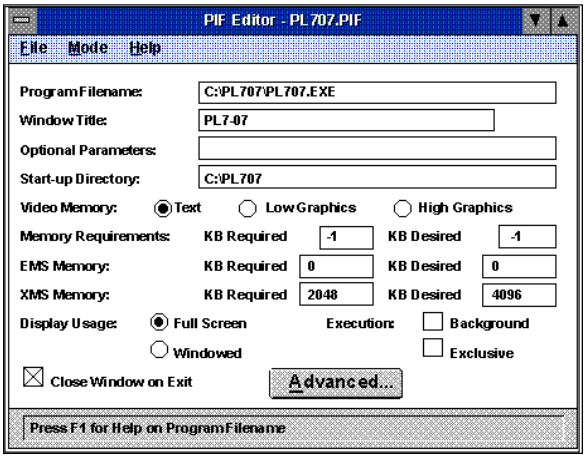
PL7-07 PC Programming Software for the TSX Nano runs in DOS, and therefore can be run under Windows within the same constraints as other graphical DOS applications. Following the suggestions listed below should allow successful operation under Windows.

A.9-1 Running PL7-07 under Windows 3.1

You should be running Windows 3.1 on a PC that has at least 4 Mb of memory. Preferably, the PC should have 8 Mb of memory if other Windows applications, drivers, and the like are open simultaneously. Further, faster CPUs and video cards in the PC will only enhance performance of PL7-07.

A PIF file should be created for PL7-07 using Windows 3.1's PIF File Editor. The PIF file tells Windows what kind of application PL7-07 is, and determines the PC resources that Windows 3.1 will give to PL7-07. You can then "run" the PIF file while in Windows 3.1 which in turn will set up Windows for PL7-07, and then automatically run PL7-07. For more information, see the Windows 3.1 documentation on creating and editing PIF files.

The following parameters should be used when creating a PIF File for PL7-07:



The Program Filename must include the DOS directory path to PL707.EXE. The path in the above example is the default path created at installation time for PL707. If you didnot choose to accept the default directory during installation of PL7-07, then enter the directory path you had provided . The same directory should be used as the Start up directory.

Video memory should be set to text. PL7-07 will configure the video for you. Enter -1 for the Memory Requirements. This will indicate to Windows 3.1 that PL7-07 should be given all the conventional PC memory possible. PL7-07 only makes use of Extended Memory. Therefore, set EMS Memory Required and Desired to zero, and set XMS Memory Required to at least 2Mb. Set XMS Memory Desired to 4Mb if running on a PC with at least 8Mb of RAM memory; otherwise set it to 2Mb.

Set the Display Usage to Full Screen. PL7-07 is a graphical environment under DOS, and Windows requires that it be run as a full screen, as opposed to a window. If you do not set this parameter in the PIF file to full screen, Windows 3.1 will provide an error message and suspend the running of PL7-07 until the window is turned into a full screen. This can be accomplished by pressing <alt><enter> together (holding done the Alt key and pressing the Enter key).

It should be noted that some VGA or SVGA video cards, especially older models, do not restore the full screen window properly after having either been minimized or turned into a window. Please be sure that you are using the latest video driver for your video card, available from the manufacturer of the video card. On some models, if the screen is restored improperly, selecting a PL7-07 menu item can sometimes clean up the display, but this is not always the case.

Leave the Execution parameters Background and Exclusive un-checked. It is also advisable to set the Close Window on Close Window checkbox. In this way, the termination of PL7-07 will automatically close the full screen window and return you to Windows 3.1's Program Manager.

Select the Advanced button to obtain the PIF file Advanced Options dialog box. The only parameters that need to be changed are the Display Options. For the best video performance, only the Text checkbox should be set in the Display Options box. Otherwise, the rest of the Advanced Options parameters can remain at their default values.

Installing the PL7-07 icon under Windows 3.1

Select or create the Modicon Telemecanique Group

To create a group:

1 In Program Manager, select the **File/New** command.
2 In the **New** dialog box, select **Group of programs** then confirm by clicking on **OK**.
3 Enter the name of the group then confirm again by clicking on **OK**.

Install the PL7-07 icon

1 In Program Manager, select the **File/New** command.
2 In the **New** dialog box, select **Program** then confirm by clicking on **OK**.

In the **Program items properties** dialog box:

3 Enter the name of the software "**PL7-07**".
4 Enter the command line **C:\PL707\PL707.EXE** (C being for example the disk drive where PL7-07 is installed).
5 Select **Change icon**.
The message "No icon available in the file" appears.
6 Confirm by clicking on **OK**.

In the **Change icon** dialog box:

7 Fill in the **Name** field by entering the command line **C:\PL707\PL707.ICO** (C being for example the disk drive where PL7-07 is installed) then confirm the various dialog boxes by clicking on **OK**.

A.9-2 Running PL7-07 under Windows 95/98/NT

The user does not need to set any particular parameters to run PL7-07 under Windows 95/98/NT.

Installing the PL7-07 icon under Windows 95/98/NT

Select or create the Modicon Telemecanique Group

To create a group :

1 In the **Start** menu, go to the **C:\Windows\Start menu\Programs** directory.
2 Select the **File/New/Folder** command.
3 Enter the name of the group then confirm again by clicking on **OK**.

Install the PL7-07 icon

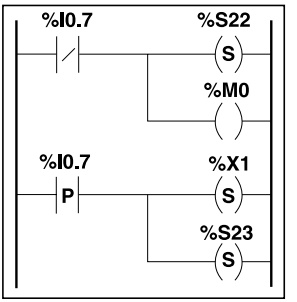
From the Modicon Telemecanique directory

1 Select the **File/New/Shortcut** command.
2 Enter the full path for the **PL707.EXE** program and click on **Next**.
3 Select a name for the shortcut and click on **Next**.
4 Select an icon for the shortcut and confirm by clicking on **Close**.

A.10 Example of prepositioning Grafcet steps

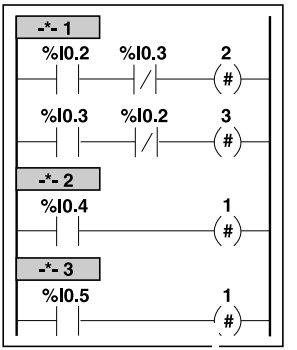
In this example, no initial step (=*=) is programmed, the chart is initialized on the rising edge of input %I0.7 in preprocessing part of the program.
The example given in SectionB2.3-2 is repeated below.

Preprocessing
In the preprocessing part of the program (area prior to the first Grafcet step), state 0 of input %I0.7 prompts a reset of the Grafcet chart (sets system bit %S22 to 1) which deactivates the active steps.
The rising edge of input %I0.7 pre-positions the chart (activation of step X1 associated with resetting system bit %S23 to 1).



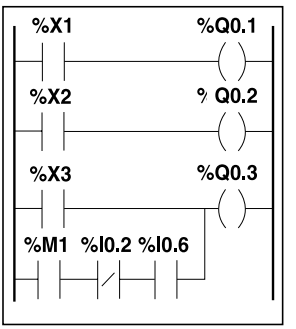

```
000  LDN    %I0.7
001  S      %S22
002  ST     %M0
003  LDR    %I0.7
004  S      %X1
005  S      %S23
```

Sequential processing
No initial step is declared; since step 1 has been initialized by input I0.7 in preprocessing.




```
006  -*-    1
007  LD     %I0.2
008  ANDN   %I0.3
009  #      2
010  LD     %I0.3
011  ANDN   %I0.2
012  #      3
013  -*-    2
014  LD     %I0.4
015  #      1
016  -*-    3
017  LD     %I0.5
018  #      1
```

Post-processing
(The Post-processing part of the program is the area following the =*= POST instruction).



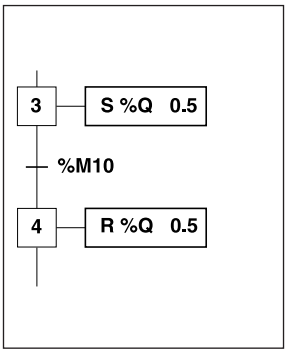

```
019  =*=    POST
020  LD     %X1
021  ST     %Q0.1
022  LD     %X2
023  ST     %Q0.2
024  LD     %X3
025  OR(    %M1
026  ANDN   %I0.2
027  AND    %I0.6
028  )
029  ST     %Q0.3
```

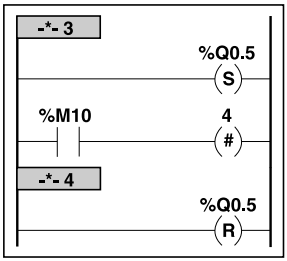
G

A.11 Actions associated with steps

The actions associated with steps can be programmed in two ways :

- By List sentences or ladder rungs associated with the step. In this case the List sentence or ladder rung is not scanned unless the step is active.
- In the Post-processing module by using the step bit %Xi (see chapter B 2.3-2) scanned on each cycle. **Therefore, it is preferable for the actions to be programmed in the Post-processing module.**






```
020  -*-     3
021  LD      1
022  S       %Q0.5
023  LD      %M10
024  #       4
025  -*-     4
026  LD      1
027  R       %Q0.5
028  •••
029  •••
```

The actions can be saved to logic conditions and are entered in the form of latched instructions or coils (SET). Each SET instruction or coil must be reset to zero by a RESET coil.

G

A

B

C

D

G

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

G