# Quantum
# Addressing Modes

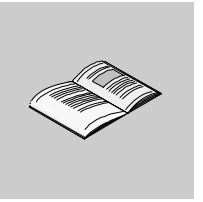
November 2007

31008084 00

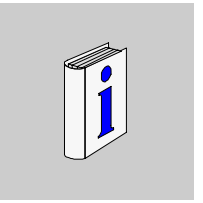
Merlin Gerin
Square D
Telemecanique

Schneider Electric

# Table of Contents



# About the Book



## At a Glance

**User Comments** We welcome your comments about this document. You can reach us by e-mail at techpub@schneider-electric.com

# Symax I/O Modules Configuration

# 1

## Configuration of Symax I/O Modules

**Introduction**

To configure a Symax I/O module under Unity you first have to configure a RIO drop which then contains your I/O modules. The following description gives you step by step instructions on how to implement and configure Symax I/O modules into a Quantum system working with Unity.

**Add a RIO Bus**

To add a RIO Bus to a Quantum system under Unity you have to perform the following steps:

| Step | Action | Comment |
|---|---|---|
| 1 | From the Project Browser "Configuration" tree open the local Quantum rack | The graphical representation of the local Quantum rack is opened. |
| 2 | Double-click an empty slot in the rack where you want to place your RIO head. | The "New Device" dialog window opens. |
| 3 | Open the "Communication" tree and double-click on the 140 CRP 93X 00 module. | The RIO head module is added to the local Quantum rack.<br>In the Project Browser "Configuration" tree the "RIO bus" is automatically added. |

### Add a Symax I/O Drop

To add a Symax I/O Drop to a Quantum system under Unity you have to perform the following steps:

| Step | Action | Comment |
|---|---|---|
| 1 | From the Project Browser "Configuration" double-click on the "RIO bus" | The graphical representation of the RIO bus is opened. |
| 2 | Double-click an empty node of the RIO bus | The 'New Device' dialog window opens. |
| 3 | In the "New Device" dialog window select the appropriate rack from the "Symax IO Drop" tree and an "Drop end communicator" module.<br>Click on the OK Button. | A new Symax I/O rack containing a communication module is added to your RIO bus. |

### Add a Symax I/O Module

To add a new I/O module to your RIO Drop you have to perform the following steps:

| Step | Action | Comment |
|---|---|---|
| 1 | From the "RIO Bus" configuration window double-click an empty slot in the rack where you want to place your I/O module. | The "New Device" dialog window opens. |
| 2 | From the "New Device" window open the list of analog or discrete modules and double-click on the appropriate module. | The I/O module is added to the rack. |

## Module Configuration

To configure an I/O module double-click on the module. The configuration window opens and you can enter the following parameters:

| Parametername | Description |
|---|---|
| MAPPING | You can define whether the access to the module is as Bits (%I-1x, %M-0x) or Words (%IW-3x, %MW-4x) |
| INPUTS STARTING ADDRESS | You have to enter the starting address of the modules input data in the address type defined by the MAPPING parameter. |
| INPUTS ENDING ADDRESS | This parameter is automatically calculated by the system |
| OUTPUTS STARTING ADDRESS | You have to enter the starting address of the modules output data in the address type defined by the MAPPING parameter. |
| OUTPUTS ENDING ADDRESS | This parameter is automatically calculated by the system |
| OUTPUT TYPE | You can define whether the modules data is interpreted as BINARY or as BCD value. |

# Quantum Addressing Modes

# 2

## Overview

**Purpose**

This chapter provides information on the three different modes Unity Pro allows to address the I/O data from a Quantum I/O module:

- Flat Addressing
- Topological Addressing
- IODDT Addressing

**Note:** The different addressing modes refer to the same physical location in the PLC memory for a given data point.

While Flat Addressing and Topological Addressing are available for all Quantum I/O modules, IODDTs are only provided for modules that deliver information in addition to the I/O values (e.g. errors or warnings).

Also provided is information about I/O module status bytes and bit order.

**What's in this Chapter?**

This chapter contains the following topics:

## Flat Addressing—800 Series I/O Modules

**Introduction**

800 series I/O modules follow a system of flat address mapping in Unity Pro. To work properly. each module requires a determinate number of bits and/or words. The IEC addressing system is equivalent to the 984LL register addressing. Use the following assignments:

- 0x is now %Mx
- 1x is now %Ix
- 3x is now %IWx
- 4x is now %MWx

The following table shows the relationship between 984LL notation and IEC notation.

| Outputs and Inputs | 984LL Notation Register Addresses | IEC Notation | | |
|---|---|---|---|---|
| | | System Bits and Words | Memory Addresses | I/O Addresses |
| output | 0x | System Bit | %Mx | %Qx |
| input | 1x | System Bit | %Ix | %Ix |
| input | 3x | System Word | %IWx | %IWx |
| output | 4x | System Word | %MWx | %QWx |

To access the I/O data of a module,

| Step | Action |
|---|---|
| 1 | Enter the address range in the configuration screen. |

**Examples**

The following examples show the relationship between 984LL register addressing and IEC addressing:

000001 is now %M1

100101 is now %I101

301024 is now %IW1024

400010 is now %MW10

## Topological Addressing—800 Series I/O Modules with Unity

### Accessing I/O Data Values

Use topological addressing to access I/O data items. Identify the topological location of the module within an 800 series I/O module with Unity Pro using the following notation:

```
%<Exchangetype><Objecttype>[\b.e\]r.m.c[.rank]
```

where:

- **b** = bus
- **e** = equipment (drop)
- **r** = rack
- **m** = module slot
- **c** = channel

**Note:** When addressing,
1. The [\b.e\] defaults to \1.1\ in a local rack and does not need to be specified.
2. The rank is an index used to identify different properties of an object with the same data type (value, warning level, error level).
3. The rank numbering is zero-based, and if the rank is zero, omit the entry.

For detailed information on I/O variables, please refer to the *Unity Pro Reference Manual*.

### Reading Values: An Example

| To read | Action |
|---|---|
| input value (rank = 0) from channel 7 of an analog module located in slot 6 of a local rack: | Enter `%IW1.6.7[.0]` |
| input value (rank = 0) from channel 7 of an analog module located in drop 3 of RIO bus 2: | Enter `%IW\2.3\1.6.7[.0]` |
| 'out of range' value (rank = 1) from channel 7 of an analog module located in slot 6 of a local rack: | Enter `%I1.6.7.1[.0]` |

## Addressing Example

**Analog Module**

The following example compares the 2 possible addressing modes. An 8-channel analog input module 8030 RIM 123 with the following configuration data is used:

- mounted in slot 5 of the RIO rack #1 located at drop 4 on bus 2
- starting input address is 201 (input word %IW201)
- end input address is 208 (input word %IW208)

To access the I/O data from the module you can use the following syntax:

| Module data | Flat addressing | Topological addressing | Concept addressing |
|---|---|---|---|
| Channel 3 | %IW203 | %IW\2.4\1.5.3 | 300203 |

For comparison, the register addressing as used with Concept is added in the last column.

**Discrete Module**

The following example compares the 2 possible addressing modes. An 32-channel discrete output module 8030 ROM 441 with the following configuration data is used:

- mounted in slot 4 of the RIO rack #1 located at drop 4 on bus 2
- starting output address is 101 (output word %MW101)
- end output address is 102 (output word %MW102)

To access the I/O data from the module you can use the following syntax:

| Module data | Flat addressing | Topological addressing | Concept addressing |
|---|---|---|---|
| Output 5 | %MW101.11 | %QW\2.4\1.4.1.1.11 | 300101 |
| Output17 | %MW102.15 | %QW\2.4\1.4.1.2.15 | 300102 |

For comparison, the register addressing as used with Concept is added in the last column. As Concept does not support direct addressing of a bit in a word, the bit extraction has to be performed in the user program.

The same configuration as before but data mapped into bits:

- mounted in slot 4 of the RIO rack #1located at drop 4 on bus 2
- starting output address is 1 (output %M1)
- end output address is 32 (output %M32)

To access the I/O data from the module you can use the following syntax:

| Module data | Flat addressing | Topological addressing | Concept addressing |
|---|---|---|---|
| Output 5 | %M5 | %Q\2.4\1.4.5 | 000005 |
| Output17 | %M17 | %Q\2.4\1.4.17 | 000017 |

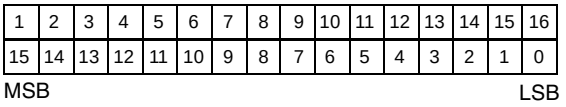
For comparison, the register addressing as used with Concept is added in the last column.

## Discrete I/O Bit Numbering

**Introduction**

The numbering of channels of an I/O module usually starts with 1 and counts up to the maximum number of supported channels. The software however starts numbering with a 0 for the least significant bit in a word (LSB). Additional the Quantum I/O modules have their lowest channel mapped to the most significant bit (MSB).

The following figure shows the mapping of I/O channels related to the bits in a word:.




**Word Addressing Versus Bit Addressing**

Mainly discrete I/O modules can be configured to deliver their I/O data either in word format or in bit format. This can be selected during configuration by selecting either `%IW` (`%MW`) or `%I` (`%M`). If you need to access a single bit from an I/O module configured to use an I/O word, you can use the syntax `%word.bit`. The following table gives you the connection between I/O point number and the belonging I/O address in bit and word addressing.

The table shows a 32-point input module in the main rack, slot 4 configured with starting address `%I1` or `%IW1`:

| I/O channel | Bit address (flat addressing) | Bit address (topological addressing) | Bit address extracted from word (flat addressing) | Bit address extracted from word (topological addressing) |
|---|---|---|---|---|
| 1 | %I1 | %I1.4.1[.0] | %IW1.15 | %IW1.4.1.1.15 |
| 2 | %I2 | %I1.4.2[.0] | %IW1.14 | %IW1.4.1.1.14 |
| 3 | %I3 | %I1.4.3[.0] | %IW1.13 | %IW1.4.1.1.13 |
| | | • • • | | |
| 15 | %I15 | %I1.4.15[.0] | %IW1.1 | %IW1.4.1.1.1 |
| 16 | %I16 | %I1.4.16[.0] | %IW1.0 | %IW1.4.1.1.0 |
| 17 | %I17 | %I1.4.17[.0] | %IW2.15 | %IW1.4.1.2.15 |
| 18 | %I18 | %I1.4.18[.0] | %IW2.14 | %IW1.4.1.2.14 |
| | | • • • | | |
| 31 | %I31 | %I1.4.31[.0] | %IW2.1 | %IW1.4.1.2.1 |
| 32 | %I32 | %I1.4.32[.0] | %IW2.0 | %IW1.4.1.2.0 |

# Index