# DU350 / DU351
# User Manual

Rev. E 07/2016
Doc. Code: MU213300

altus

According to legislation in force in Brazil, the Consumer Defense Code, we are giving the following information to clients who use our products, regarding safety of personal and premises.

The industrial automation equipment, built by ALTUS is strong and reliable due to the stringent quality control it is subjected to. However the electronic industrial control equipment (programmable controllers, numerical commands, etc.) can cause damage to the machines or processes controlled by them when there are defective components and programming or installation errors. This can even put human lives at risk.

The user should consider the possible consequences of the defects and should provide additional external installations for security so that, if necessary, the security of the system can be maintained especially during the initial installation and testing.

Equipment manufactured by Altus does not directly expose the environment to hazards, since they do not issue any kind of pollutant during their use. However, concerning the disposal of equipment, it is important to point out that any built-in electronics contain materials which are harmful to nature when improperly discarded. Therefore, it is recommended that whenever discarding this type of product, it should be forwarded to recycling plants, which guarantee proper waste management.

It is essential to completely read the manuals and/or about the technical characteristics of the product before its installation or use.

ALTUS warrants its equipment as described in General Conditions of Supply, attached to the commercial proposals.

ALTUS guarantees that their equipment works in accordance with the clear instructions contained in their manuals and/or the technical characteristics, not guaranteeing the success of any particular type of application of the equipment.

ALTUS does not acknowledge any other guarantee, direct or implied, principally when it is dealing with supply of other companies.

Requests for additional information about the supply and/or characteristics of the equipment and ALTUS services should be put in writing. The address for ALTUS can be found on the back cover. ALTUS is not responsible for supplying information about their equipment without formal registration.

# Table of Contents

# 1. Introduction

The Duo series of programmable controllers (PLC) with built-in Human Machine Interface (HMI) is the ideal solution for applications which require control and supervision in a single product and environment. This solution was designed through a hardware architecture based on 32-bit processors and high performance. The main product features are the speed of application processing, the high density of I/O integrated points, the connectivity with other elements of the environment through two serial ports and the requirements for axes positioning using fast inputs and outputs.

The programming of the product is performed through a single tool, using for that six PLC programming languages (five of them described in IEC 61131-3: LD, ST, IL, FBD and SFC) plus an additional language (CFC). MasterTool IEC has an important simulation feature that allows the user to test the application without the use of equipment, thus allowing for greater agility in the program development.

In addition to an HMI that supports text and graphics, alphanumeric keypad, 7 function keys, the innovation of the product is also found in its presentation. The ID Duo allows the client to customize the brand design in accordance with the application and the company branding.

The DU350 and DU351 models differ in the type of digital outputs available in each one. While the DU350 comprises 14 normal outputs and 2 configurable transistor fast outputs, the DU351 has 14 relay normal outputs and two configurable transistor fast outputs.


**Figure 1-1. DU350/DU351 Controller**

The main product features are:

- MasterTool IEC programmer with 6 programming languages, among which 5 are described by IEC 61131-3 standard, plus one additional language
- 20 insulated digital inputs, among which 6 are fast points configurable for 2 two-way

counters, 4 one-way counters or for positioning functions
- 14 insulated digital outputs (relay or transistor)
- 2 insulated digital transistor outputs configurable as 2 fast output points – PTO, PWM/VFO – or for positioning functions
- 4 analog inputs – configurable as 0 to 10 V, 0 to 20 mA or 4 to 20 mA
- 2 analog outputs – configurable as 0 to 10 V or 0 to 20 mA
- Graphics LCD display with a 128 x 64 dot matrix –configurable by MasterTool IEC programmer
- Membrane keyboard with 25 keys
- Real-time clock
- One RS-232 serial port for programming and MODBUS Master and Slave and generic protocol
- One RS-485 serial port for MODBUS Master and Slave and generic protocol
- 24Vdc insulated power supply
- 256 kbytes application memory
- Capacity up to 1250 bytes of retentive memory without battery

## Documents Related to this Manual

To get additional information about Duo Series, other documents (manuals and technical characteristics) may be consulted in addition to this one. The last revision of these documents is available at www.altus.com.br.

Each product has a document entitled Technical Characteristics (CE), where are the characteristics of the product concerned. Additionally the product may have Utilization Manuals (the manuals codes are cited in CT).

For example, the PO2022 module has all the information about characteristics of use and purchase, in its CE. On the other side, the PO5063 has, in addition to the CT, a Utilization Manual.

The following documents are suggested as source of additional information:

- Technical Characteristics– CE113100
- MasterTool IEC Utilization Manual – MU299606

## Visual Inspection

Before the installation, a careful visual inspection of the product is recommended to verify whether there has been any damage during the freight. Check if the CD-ROM is in perfect condition. In case of defects, please inform the company or the nearer ALTUS distributor.

**WARNING:**
**Before unpacking the modules, it is important to discharge any static potential accumulated in the body. To do this, touch (with bare hands) a grounded metal surface before handling the modules. This procedure ensures that the levels of static electricity supported by the module will not be exceeded.**

It is important to register the serial number of the equipment received, as well as the revision of software if applicable. This information will be necessary in case there is need to contact Altus Technical Support.

## Technical Support

To contact Altus Technical Support in São Leopoldo/RS please call +55 (51) 3589-9500. For information on Altus Technical Support centers in other localities, go to our site (www.altus.com.br) or send an e-mail to altus@altus.com.br.

If the software is already installed, please have the following information from our assistance upon request:

- The models of the used equipment and the configuration of installed system
- The CPU serial number
- The revision of the equipment indicated in the tag in the back of the product
- The executive software version described in the special screen INFORMATION
- The application program content obtained through MasterTool IEC Programmer
- The version of the used programmer

## Warning Messages Used in this Manual

In this manual, the warning messages will be presented in the following formats and descriptions:

**DANGER:**
**Reports potential hazard that, if not detected, may be harmful to people, materials, environment and production.**

**WARNING:**
**Reports configuration, application or installation details that must be taken into consideration to avoid any instance that may cause system failure and consequent impact.**

ATTENTION:
Identifies configuration, application and installation details aimed at achieving maximum operational performance of the system.

# 2. Technical Description

This chapter presents the DU350 and DU351 controllers technical features.

## General Features

| | DU350, DU351 |
|---|---|
| **Number of digital input points** | 20 insulated digital input points:<br>14 general purpose digital inputs<br>6 fast digital inputs (fast inputs can be used as counters or general purpose inputs) |
| **Number of relay/transistor output points** | 16 insulated digital output points:<br>2 fast output points and 14 transistor output points -DU350<br>2 fast output points and 14 relay output points - DU351 |
| **Number of fast output points** | 2 fast output points: PTO, PWM, frequency or digital output |
| **Fast counters** | 6 fast points divided into 2 blocks configurable as uni/bidirectional (2 counters per block) |
| **Analog inputs** | 4 analog inputs:<br>0 to 10 V , 0 to 20 mA or 4 to 20 mA |
| **Analog outputs** | 2 analog outputs:<br>0 to 10 V or 0 to 20 mA |
| **Real Time Clock RTC** | Yes, retains data for 15 days without power supply.<br>Resolution of 1 second and maximum variance of 2 seconds per day |
| **Display** | Graphical monochrome LCD display, with a dot-matrix of 128 x 64 pixels, *backlight* and contrast control. |
| **Keyboard** | Membrane keyboard with 25 keys |
| **MODBUS protocol** | Master and Slave RS-232 and RS-485 |
| **Field application download** | Yes, through COM 1, RS-232 |
| **Online programming** | No |
| **RS-232 interface** | Yes, one port with modem signals: TXD, RXD, RTS, CTS, DTR, DSR, DCD |
| **RS-485 interface** | Yes, non-insulated |
| **Watchdog circuit** | Yes |
| **External power supply voltage** | 19 to 30 Vdc |
| **24 Vdc input current** | 350 mA |
| **Power consumption** | 8.4 W |
| **Maximum power supply interruption** | 10 ms @ 24 Vdc |
| **Power supply insulation** | Yes |
| **Initialization time** | 10 s |
| **Related standards** | IEC 61131-3 2003 |
| **Weight** | 600 g |
| **Operation temperature** | 0 to 60°C |
| **Storage temperature** | -20 to 75°C |
| **Front panel protection** | IP 54 |
| **Back panel protection** | IP 20 |
| **Dimensions** | 180.1 x 144.1 x 51 mm |

**Table 2-1. General Features**

### Notes

**RTC:** In environments with temperature of 25 °C. The retaining time may vary between 10 and 20 days throughout the temperature range of the product operation.

# Digital Inputs

| | DU350, DU351 |
|---|---|
| **Number of inputs** | 20 digital inputs divided into 3 insulation groups:<br>I00..I08 - 9 inputs – Group 0<br>I10..I18 - 9 inputs – Group 1<br>I20..I21 - 2 inputs – Group 2 |
| **Input voltage** | 14 to 30 Vdc (common reference) for state 1<br>0 to 5 Vdc (common reference) for state 0 |
| **Input current** | 5 mA (24 Vdc common reference) – general purpose inputs<br>15 mA (24 Vdc common reference) – fast inputs |
| **Input type** | "sink" type 1 |
| **Input impedance** | 4.3 KΩ - general purpose inputs<br>1.5KΩ - fast inputs |
| **Insulation** | 2000 Vac for one minute between each input group<br>2000 Vac for one minute between input group and logic circuit |
| **Terminal block configuration** | The digital inputs are divided in 3 connectors (insulation groups) insulated among themselves and insulated from the logic circuit. Each connector is composed of a terminal block for each input and a terminal block to the reference voltage.<br><br>I00 to I08 – input 0 to 8 of the insulation group 0<br>I10 to I18 – input 0 to 8 of the insulation group 1<br>I20 to I21 – input 0 to 1 of the insulation group 2<br>C0 – common of the insulation group 0<br>C1 – common of the insulation group 1<br>C2 – common of the insulation group 2<br><br>Inputs I00 to I02 and I10 to I12, are fast inputs. Inputs I00 to I02, belong to Block 0 fast inputs; inputs I10 to I02 belong to the Block 1 fast inputs. Fast inputs can be used as general purpose inputs. |
| **Response time** | 0.5 ms – general purpose inputs<br>10 μs – fast inputs |
| **Status indication** | Can be accessed at product standard screens |

**Table 2-2. Digital Inputs Features**

### Notes

**Response time:** the maximum response time for general purpose digital inputs will be the response time plus the maximum cycle.

# Transistor Digital Outputs (DU350)

| | DU350 |
|---|---|
| **Number of common outputs** | 14 transistor digital outputs divided into 2 insulation groups:<br>Q02 to Q07 – 6 outputs – Group 0<br>Q10 to Q17 – 8 outputs – Group 1 |
| **Maximum current per point** | 0.5 A |
| **Output type** | Transistor “source” type |
| **Switching time** | 600 µs |
| **Maximum switching frequency with load** | 250 Hz, with minimum external load of 12500 Ω |
| **Status indication** | Can be accessed at product standard screens |
| **Protection** | TVS diode in all transistor outputs |
| **Operation voltage** | 10 to 30 Vdc |
| **Insulation** | 2000 Vac for one minute between each output transistor group<br>2000 Vac for one minute between output transistor group and logic circuit |
| **Output impedance** | 500 mΩ |
| **Terminal block configuration** | Transistor digital outputs are divided into 2 connectors (insulation groups). Each connector is made up of a terminal block for each output, a terminal block to the common contact (supply) and a 0 V reference terminal block.<br><br>Q02 to Q07 – transistor output 2 to 7 of insulation group 0<br>Q10 to Q17 – transistor output 0 to 7 of insulation group 1<br>C5 – reference (0 V) insulation group 0 (shared with fast outputs)<br>C6 – power supply of insulation group 0 (shared with fast outputs) (maximum 30 Vdc)<br>C7 – reference (0 V) insulation group 1<br>C8 – power supply of insulation group 1 (maximum 30 Vdc) |

**Table 2-3. Transistor Outputs Features**

### *Notes*

**Maximum current per point:** the transistor output has no overload protection. If output protection is necessary, an external fuse must be used with the product.

**Terminal block configuration:** the insulation group 0 has two fast transistor outputs (Q00 and Q01).

# Relay Digital Outputs (DU351)

| | DU351 |
|---|---|
| **Number of outputs** | 14 relay digital outputs divided into 2 insulation groups:<br>Q02 to Q07 – 6 outputs – Group 0<br>Q10 to Q17 – 8 outputs – Group 1 |
| **Maximum current per point** | 1 A |
| **Output type** | Normally open relay |
| **Minimum load** | 5 mA |
| **Expected useful life** | $10x10^4$ operations with rated load |
| **Maximum commutation time** | 10 ms |
| **Maximum switching frequency** | 0.5 Hz with maximum rated load |
| **Status indication** | Can be accessed at product standard screens |
| **Maximum voltage (C6,C8)** | 30 Vdc insulation group 0<br>30 Vdc insulation group 1<br>240 Vac insulation group 1 |
| **Insulation** | 2000 Vac per one minute between each output group<br>2000 Vac per one minute between output group and logic circuit |
| **Resistance contact** | < 250 mΩ |
| **Terminal block configuration** | The relay digital output is divided into 2 connectors (insulation groups). Each connector is made up of a terminal block for each output, a terminal block to the common contact to all relays of the same connector and a 0 V terminal block (only used in transistor outputs).<br><br>Q02 to Q07 – output relay 2 to 7 of insulation group 0<br>Q10 to Q17 – output relay 0 to 7 of insulation group 1<br>C5 – not used for relays output<br>C6 – common relays of insulation group 0, and used for supply fast outputs. In the output option type SINK (0 Vdc in Pin C6) fast outputs Q00 and Q01 cannot be used.<br>The relays of insulation group 0 cannot trigger AC loads<br>The use of alternated voltage in PIN C6 may cause irreparable damage to the product.<br>C7 – Pin not used for relay outputs<br>C8 – Pin connected to the common of relays of the insulation group 1 |

**Table 2-4. Relay Outputs Features**

***Notes***

**Maximum current per point:** the relay output has no overload protection. If output protection is necessary, an external fuse must be used with the product

**Terminal block configuration:** the insulation group 0 has two fast transistor outputs (Q00 and Q01).

ATTENTION:
The use of alternated voltage in pin C6 may cause irreparable damage to the product.

# Fast Digital Outputs

| | DU350, DU351 | |
|---|---|---|
| **Number of outputs** | 2 fast:<br>Q00 and Q01 | |
| **Maximum current per point** | 0.5 A | |
| **Output type** | Transistor “source” type | |
| **Maximum frequency of pulse generation** | 50 kHz | |
| **Minimum pulse bandwidth @ 24 Vdc** | **MINIMUM EXTERNAL LOAD** | **MINIMUM PULSE TIME** |
| | No load | 20 us |
| | 1000 Ω | 4 us |
| | 50 Ω | 2 us |
| **Status indication** | Through reserved fixed operands | |
| **Protection** | TVS diode in all transistor outputs | |
| **Operation voltage** | 10 to 30 Vdc | |
| **Insulation** | 2000 Vac per one minute between each output group<br>2000 Vac per one minute between output group and logic circuit | |
| **Output impedance** | 700 mΩ | |
| **Terminal block configuration** | Fast outputs are located in the connector of digital output insulation group 0. The connector is made up of a terminal block for each output, a terminal block to the common contact (power supply) and a 0 V terminal block reference.<br><br>Q00 to Q01 – fast output 0 to 1 of insulation group 0<br>C5 – 0V insulation group 0 (shared with the transistor outputs Q02 to Q07 (DU350)<br>C6 – power supply, insulation group 0 (shared with the relay outputs Q02 to Q07 (DU351) / transistor (DU350)). Maximum voltage 30 Vdc | |
| **Output modes** | PTO, VFO and PWM | |
| **Maximum number of used outputs** | 2 | |
| **Functions performed via software** | **PTO** | **VFO/PWM** |
| | Writing of the value (number of pulses) to be generated<br>Writing of the number of pulses to be generated in the acceleration / deceleration<br>Start / end of the outputs operations<br>Fast outputs diagnostics<br>Monitoring of the current state of fast outputs | Writing of the frequency value to be generated in Hz (1 Hz to 50 kHz)<br>Writing of the outputs Duty-Cycle (0 to 100 %)<br>Start / end of outputs operations<br>Fast outputs diagnostics |
| **Access way to fast outputs registers** | In reserved fixed operands | |

**Table 2-5. Fast Outputs Features**

***Notes***

**Maximum current per point:** the transistor fast output has no overload protection. If output protection is necessary, an external fuse must be used with the product.

**Functions performed via software:** variation of 1 in 1 Hz or 1 in 1 % for frequency settings and duty cycle respectively.

| |
|---|
| ATTENTION:<br>During the product power on, fast outputs may change the output state for approximately 100 us. |

# Analog Inputs

| | DU350, DU351 |
|---|---|
| **Number of inputs** | 4 non insulated (of logic circuit) analog inputs |
| **Input type** | Voltage: 0 to 10 Vdc<br>Current: 0 to 20 mA, 4 to 20 mA |
| **Conversion resolution** | 12 bits |
| **Terminal block configuration** | AV0 – voltage input channel 0<br>AI0 – current input channel 0<br>C9 – common for inputs 0 and 1<br>AV1 – voltage input channel 1<br>AI1 – current input channel 1<br>AV2 – voltage input channel 2<br>AI2 – current input channel 2<br>C10 – common for inputs 2 and 3<br>AV3 – voltage input channel 3<br>AI3 – current input channel 3 |
| **Configurable parameters** | Inputs type for each point, voltage or current<br>Scale limit for each channel, maximum 30000<br>First order filtering with preset time constants |
| **Protections** | TVS diode in all analog inputs |
| **Updating Time** | 60 ms |

**Table 2-6. Analog Inputs Features**

***Notes***

**Updating time:** necessary time for the AD to make available a new value of a channel to CPU.

## Voltage Mode

| | DU350, DU351 | | |
|---|---|---|---|
| **Precision** | ± 0.3% of scale limit @ 25°C<br>± 0.015% / ºC of the scale limit | | |
| **Crosstalk DC at 100 Hz** | - 30 dB | | |
| **Input impedance** | 1.1 MΩ | | |
| **Maximum / minimum voltage without damage** | 12 Vdc / -0.3 Vdc | | |
| **Filtering** | Configurable time constant<br>90 ms, 140 ms, 1 s or 15 s | | |
| **Scale** | Range | Counting | Sensitivity |
| | 0 - 10 V | 0 - 30.000 | 2.52 mV |
| **Scale clearance** | 3 % | | |

**Table 2-7. Voltage Mode Inputs Features**

***Notes***

**Filtering:** the time constant value may vary by 10% of its nominal value. The maximum deviation of the constant time value is equal to the sampling rate; e.g., by selecting the time constant of 140 ms, the maximum time required for the voltage value with filter to have 63% of the input value is of 140 ms * 110% + 60 ms = 214 ms

**Counting:** the scale limit may be modified by software, the maximum value is 30000.

**Scale clearance:** defines the percentage over limit scale that can be read by analog inputs. This characteristic may be used to compensate for possible calibration errors of a given sensor which is employed.

### Current Mode

| | DU350, DU51 | | |
|---|---|---|---|
| **Precision** | ± 0.3% of scale limit @ 25°C<br>± 0.015% / ºC of the scale limit | | |
| **Crosstalk DC @ 100 Hz** | - 30 Db | | |
| **Input impedance** | 124.5 Ω | | |
| **Maximum/minimum continuous current without damage** | 25 mA / -2 mA | | |
| **Filtering** | Configurable time constant:<br>2 ms, 90 ms, 1 s or 15 s | | |
| **Scale** | Range | Counting | Sensitivity |
| | 0 - 20 mA | 0 - 30.000 | 5,1 μA |
| | 4 - 20 mA | 0 - 30.000 | 5,1 μA |
| **Diagnostic** | Current below 3.8 mA (valid only with 4 - 20 mA scale; used to indicate an open circuit) | | |
| **Scale clearance** | 4 % | | |

**Table 2-8. Current Mode Inputs Features**

***Notes***

**Filtering:** the time constant value may vary by 10% of its nominal value. The maximum deviation of the constant time value is equal to the sampling rate; e.g., by selecting the time constant of 1 s, the maximum time required for the current value with filter to have 63% of the input value is of 1 s * 110% + 60 ms = 1.16 s.

**Counting:** the scale limit may be modified by software, the maximum value is 30000.

**Scale clearance:** defines the percentage over limit scale that can be read by analog inputs. This characteristic may be used to compensate for possible calibration errors of a given sensor which is employed.

## Analog Outputs

| | DU350, DU351 |
|---|---|
| **Number of inputs** | 2 non insulated (of logical circuit) analog outputs |
| **Output type** | - Voltage: 0 to 10 Vdc<br>- Current: 0 to 20 mA |
| **Conversion resolution** | 12 bits |
| **Terminal block configuration** | C3 – common for AO0 output.<br>AO0 – analog output 0 (configurable by software as voltage or current)<br>C4 – common for AO1 output<br>AO1 – analog output 1 (configurable by software as voltage or current)) |
| **Protection** | TVS diode in all analog outputs |
| **Configurable parameters** | Type of signal in each channel (voltage or current)<br>Scale limit per channel, maximum 30000 |

**Table 2-9. Analog Outputs Features**

## Voltage Mode

| | DU350, DU351 | | |
|---|---|---|---|
| **Precision** | ± 0.3% of scale limit @ 25°C<br>± 0.015% / ºC of scale limit | | |
| **Crosstalk DC @ 100 Hz** | - 30dB | | |
| **Output impedance** | 22 Ω | | |
| **Minimum load impedance** | 600 Ω | | |
| **Short circuit protection** | Yes | | |
| **Updating time** | 1 ms | | |
| **Stabilization time** | 4 ms | | |
| **Scale** | Range | Counting | Resolution |
| | 0 - 10 V | 0 - 30.000 | 2.59 mV |
| **Scale clearance** | 4 % | | |
| **Over load indication** | Yes (typically loads with impedance lower than 500 Ω) | | |

**Table 2-10. Voltage Mode Outputs Features**

### *Notes*

**Updating time:** maximum time between the end of cycle and the output updating.

**Stabilization time:** maximum time for output signal stabilization with a resistive load greater than or equal to 600 Ω.

**Counting:** the scale limit may be modified by software, the maximum value is 30000.

**Scale clearance:** scale clearance allows DA to reaches voltage values out of range to compensate for eventual offset errors of the device to be controlled by the analog output.

## Current Mode

| | DU350, DU351 | | |
|---|---|---|---|
| **Precision** | ± 0.3% of scale limit @ 25°C<br>± 0.015% / ºC of the scale limit | | |
| **Crosstalk DC at 100 Hz** | - 30dB | | |
| **Maximum load impedance** | 600 Ω | | |
| **Updating time** | 1 ms | | |
| **Stabilization time** | 4 ms | | |
| **Scale** | Range | Counting | Minimum Resolution |
| | 0 - 20 mA | 0 - 30.000 | 5.2 µA |
| **Scale clearance** | 4 % | | |
| **Open loop indication** | Yes (typically loads with impedance greater than 650 Ω) | | |

**Table 2-11. Current Mode Outputs Features**

### *Notes*

**Updating time:** maximum time between the end of a cycle and the output updating.

**Stabilization time:** maximum time for output signal stabilization with a resistive load higher than or equal to 600 Ω.

**Counting:** the scale limit may be modified by software, the maximum value is 30000.

**Scale clearance:** scale clearance allows DA to reach voltage values above range to compensate for eventual offset errors of the devices to be controlled by the analog outputs.

# Fast Counters

| | DU350, DU351 |
|---|---|
| **Number of counters** | Up to 4 (32-bit) configurable fast counters |
| **Counting modes** | **Unidirectional**<br>Increments or decrements<br>**Bidirectional**<br>A: Increments B: Decrements<br>A: Counts B: Direction<br>Square with two countings per period (2x)<br>Square with four countings per period (4x) |
| **Data format** | 32-bit unsigned integer |
| **Operation limit** | 4.294.967.295 |
| **Maximum input frequency** | 40 kHz for Block 0 and 20 kHz for Block 1 |
| **Terminal block configuration** | Block 0<br>I00 – Input A (bidirectional) or Block 0 counter 0 (unidirectional mode)<br>I01 – Input B (bidirectional) or Block 0 counter 1 (unidirectional mode)<br>I02 – Configurable input block 0<br>Block 1<br>I10 – Input A (bidirectional) or Block 1 counter 2 (unidirectional mode)<br>I11 – Input B (bidirectional) or Block 1 counter 3 (unidirectional mode)<br>I12 – Configurable input block 1 |
| **Configurable parameters** | Counting mode<br>Inputs function<br>Outputs function<br>Comparison registers |

**Table 2-12. Fast Counters Features**

# Local Serial Channel

| | **DU350, DU351** |
|---|---|
| **Physical layer** | RS-232 |
| **Protocol** | MODBUS RTU Master and Slave, communication with MasterTool IEC programmer and generic protocol |
| **Hardware Signs** | RTS, CTS, DCD, DTR and DSR |
| **Insulation with logical circuit** | Yes |
| **Connector** | RJ45 |

**Table 2-13. RS-232 Serial Channel Features**

# RS-485 Serial Channel

| | DU350, DU351 |
|---|---|
| **Physical layer** | RS-485 |
| **Protocol** | MODBUS RTU Master and Slave and generic protocol |
| **Internal termination** | No |
| **Insulation with logical circuit** | No |
| **Connector** | RJ45 |

**Table 2-14. RS-485 Serial Channel Features**

# Software Features

| | DU350, DU351 |
|---|---|
| **Programming language** | IL, ST, LD, SFC, FBD and CFC |
| **Online programming** | No |
| **Language conversion** | Conversion of available language codes |
| **Application memory** | 256 kbytes |
| **Average memory per IL instructions** | 1000 instructions each 7 kbytes |
| **Memory for source code download** | 256 kbytes |
| **I Data type memory** | 128 bytes |
| **Q Data type memory** | 128 bytes |
| **M Data type memory** | 6656 bytes |
| **Global memory** | 6656 bytes |
| **Retain** | 1250 bytes non-volatile memory |
| **Number of POUs** | 300 |
| **IHM programmer** | Included in the same application developer's interface |
| **Simulator** | PLC simulator for application tests without the equipment |
| **Advanced functions** | Available libraries for advanced control |

**Table 2-15. Software Features**

***Notes***

**I-type and Q- type data memory:** the "No address check" in "Device Settings" is not selected by default. In this case it is only possible to use the DU350/DU351 digital and analog inputs as well as outputs associated to addresses in the application. If this option is selected the entire range may be used.

**Global Memory:** in this area of 6656 bytes it is necessary to use one byte for control, so only 6655 bytes can be declared (the remaining byte is for control).

**Software version:** the data in the table refer to the available memory from DU350/DU351 firmware 1.10 version.

**Retain:** When a retain variable is assigned using an AT to related it with a direct address of %I, %Q or %M, the DU350 and DU351 controllers do not treat this variable as retain.

# Compatibility with Other Products

| Code | Description |
|---|---|
| **MT8200** | MasterTool IEC version 1.0 or above |

**Table 2-16. Compatibility**

***Notes***

**Features:** Some of the features described in this CT are only available in certain versions of the MasterTool IEC and DU350/ DU351 software. For more details see the specific features sections.

# Performance

## Memory Map

DU350 and DU351 controllers have the following storage areas available for the user application:

| Memory | Type | Size |
|---|---|---|
| **Non-volatile code** | Application code | 256 kbytes |
| **Volatile data** | Global Operands | 6656 bytes |
| | Memory Operands (%M) | 6656 bytes |
| | Input Operands (%I) | 128 bytes |
| | Output Operands (%Q) | 128 bytes |
| **Non Volatile data** | Retentive Operands | 1250 bytes |

**Table 2-17. Quantity of Operands**

Global operands are used in declaration of operands without a user-defined address. For each task added to the project, the system uses up to 35 bytes of global operands memory. In case of external library usage (Standard.lib, SysLibTargetVisu .lib, ...) the system allocates the required memory for the libraries in Global Operands area.

## Application Times

The required time for distinct instructions to be executed by DU350 and DU351 controllers is shown in the following table. All measurements were done in LADDER language:

| Instruction | Operands | Instruction times (us) |
|---|---|---|
| **Overhead** | indifferent | 26 us |
| **1000 Lines with 10 Contacts** | indifferent | 94.4 us |
| **1000 Function Calls** | indifferent | 864 us |
| **1000 MUL Instructions** | BYTE | 404 us |
| | WORD | 440 us |
| | REAL | 1610 us |
| **1000 ADD Instructions** | BYTE | 388 us |
| | WORD | 416 us |
| | REAL | 1608 us |

**Table 2-18. Instruction Times**

The overhead is calculated by running a program without any instruction.

## Language Restrictions

DU350 and DU351 controllers do not support 64-bit type operands. They are: LWORD, LREAL, LINT, LREAL and ULINT.

ATTENTION:
LREAL may be handled as REAL by checking this option in the MasterTool IEC programmer. However, this option will only replace these types of variables, causing them to be handled as REAL, i.e. 32-bit variables.

### Software Restrictions

DU350 and DU351 controllers do not support breakpoint usage (possibility of freezing the application currently running for monitoring and debugging) in the “logged-in” mode. The restriction covers the following functions: Toggle Breakpoint, Breakpoint Dialog, Step Over, Step In and Single Cycle (Online menu).

### Variable Types Restriction

DU350 an DU351 controllers do not support PERSISTENT type variables usage.

When a retain variable is assigned using an AT to related it with a direct address of %I, %Q or %M, the DU350 and DU351 controllers do not treat this variable as retain.

### Startup Times

DU350 and DU351 controllers have a startup time of 10 seconds.

### Analog Outputs Times

In DU350 and DU351 controllers, the updating interval of voltage and current analog outputs is equal to the program cycle time. In case the function AES_ANALOG_OUTPUT is employed, the outputs are updated at the time the function is called.

### Analog Inputs Times

In DU350 and DU351 controllers , the updating interval of voltage and current analog inputs is of 60 ms. In case the function AES_ANALOG_INPUT is employed, the inputs remain with an updating interval of 60 ms, but the AIx operands are updated with the last reading cycle value of the analog inputs.

### Cycle Time

The CPU cycle time depends on the application and may vary from 1 to 2000 ms.

**WARNING:**
**Communication with the MasterTool IEC programmer is affected by the cycle time. The bigger the cycle time, the slower data exchange with it. A cycle time which is too close to 2000 ms may cause a communication time-out in MasterTool IEC. In case the long cycle time is caused by a problem in the application logic, the following procedure is recommended: de-energize PLC, press ESC, power it up again, wait for the boot screen, and release the ESC key. Thus the CPU will not start the application, allowing the corrected application to be downloaded. If the application requires a very high cycle time (but less than 2000 ms), it is possible to increase the programming interface time-out. In order to do this, it is necessary to enter “Project” → “Options...” → “Desktop”, and change the values of the fields “Communication timeout [ms]” and “Communication timeout for download [ms]”. The MasterTool IEC default timeout is 2000 ms for both fields.**

### Fast Inputs Performance

Due to the increased frequency at the inputs, the fast inputs need more processing. The required processing varies according to the counter blocks setup. The table below describes the processing dedicated to fast counting peripherals, with different counter configurations as well as how much processing is left for implementation in each case.

| | Block 0 | | Block 1 | | Processing (%) | |
|---|---|---|---|---|---|---|
| | Counter 0 | Counter 1 | Counter 0 | Counter 1 | Counters | Application |
| **1** | Mode 0 - 40 kHz | | Mode 0 - 20 kHz | | 19.4% | 80.6% |
| **2** | Mode 0 - 40 kHz | | Mode 1 - 20 kHz | | 19.7% | 80.3% |
| **3** | Mode 0 - 40 kHz | | Mode 2 - 10 kHz | | 29.8% | 70.2% |
| **4** | Mode 0 - 40 kHz | | Mode 3 - 20 kHz | | 29.6% | 70.4% |
| **5** | Mode 0 - 40 kHz | | Unit - 20 kHz | Unit - 20 kHz | 30.3% | 69.7% |
| **6** | Mode 1 - 40 kHz | | Mode 0 - 20 kHz | | 29.1% | 70.9% |
| **7** | Mode 1 - 40 kHz | | Mode 1 - 20 kHz | | 29.1% | 70.9% |
| **8** | Mode 1 - 40 kHz | | Mode 2 - 10 kHz | | 37.3% | 62.7% |
| **9** | Mode 1 - 40 kHz | | Mode 3 - 20 kHz | | 37.9% | 62.1% |
| **10** | Mode 1 - 40 kHz | | Unit - 20 kHz | Unit - 20 kHz | 32.0% | 68.0% |
| **11** | Mode 2 - 20 kHz | | Mode 0 - 20 kHz | | 40.5% | 59.5% |
| **12** | Mode 2 - 20 kHz | | Mode 1 - 20 kHz | | 33.6% | 66.4% |
| **13** | Mode 2 - 20 kHz | | Mode 2 - 10 kHz | | 44.0% | 56.0% |
| **14** | Mode 2 - 20 kHz | | Mode 3 - 20 kHz | | 50.0% | 50.0% |
| **15** | Mode 2 - 20 kHz | | Unit - 20 kHz | Unit - 20 kHz | 44.3% | 55.7% |
| **16** | Mode 3 - 40 kHz | | Mode 0 - 20 kHz | | 37.5% | 62.5% |
| **17** | Mode 3 - 40 kHz | | Mode 1 - 20 kHz | | 45.1% | 54.9% |
| **18** | Mode 3 - 40 kHz | | Mode 2 - 10 kHz | | 49.4% | 50.6% |
| **19** | Mode 3 - 40 kHz | | Mode 3 - 20 kHz | | 47.9% | 52.1% |
| **20** | Mode 3 - 40 kHz | | Unit - 20 kHz | Unit - 20 kHz | 42.2% | 57.8% |
| **21** | Unit - 40 kHz | | Mode 0 - 20 kHz | | 33.3% | 66.7% |
| **22** | Unit - 40 kHz | | Mode 1 - 20 kHz | | 33.6% | 66.4% |
| **23** | Unit - 40 kHz | | Mode 2 - 10 kHz | | 35.7% | 64.3% |
| **24** | Unit - 40 kHz | | Mode 3 - 20 kHz | | 35.3% | 64.7% |
| **25** | Unit - 40 kHz | | Unit - 20 kHz | Unit - 20 kHz | 29.1% | 70.9% |

**Table 2-19. Free Processing for Application**

## Physical Dimensions

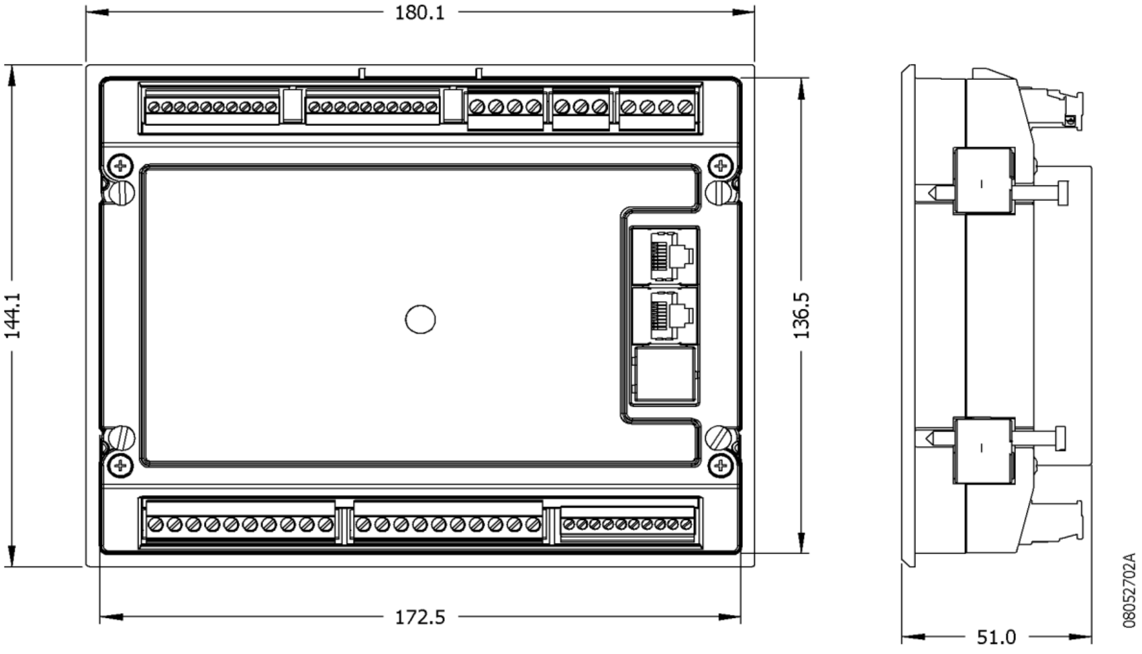


**Figure 2-1. DU350 and DU351 Dimensional**

# Ordering Information

## Included Items

The product package contains the following items:

- DU350 or DU351 modules
- 8 connectors
- installation guide

## Product Code

The following code should be used to purchase the product:

| Code | Denomination |
|---|---|
| **DU350** | PLC WITH HMI 20DI 16DOT 4AI 2AO |
| **DU351** | PLC WITH HMI 20DI 14DOR 2DOT 4AI 2AO |

**Table 2-20. Duo Series Models**

# Related Products

The following code should be used to purchase the product:

| Code | Denomination |
|---|---|
| **AL-1714** | Cable RJ45 – RJ45 RS-232 (PICCOLO) |
| **AL-1715** | Cable RJ45-CFDB9 |
| **PO8500** | Expansion cable 0.4 m |
| **PO8501** | Expansion cable 1.4 m |
| **PO8525** | RS-485 network splitter and termination |
| **FBs-USB-232M9** | USB-SERIAL Cable Conversor |

**Table 2-21. Related Products**

### *Notes*

**AL-1715:** this cable has a serial RJ45 connector and other DB9 RS-232C female IBM/PC standard. It should be used to communicate the module with MasterTool IEC software and for point-to-point RS-232 communication interface using the RTU MODBUS protocol.

**AL-1714**: this cable has a RJ45 connector at each end and is used for point-to-point RS-232 communication interface between two DU350 or two DU351, using the RTU MODBUS PROTOCOL.

**PO8500 and PO8501:** these cables have a RJ45 connector at each end. They are used to connect the RS-485 splitter PO8525 with the RS-485 serial port of DU350 or DU351.

**PO8525:** this module is used to split and terminate a RS-485 network. Each node in the network must have a PO8525. In the RJ45 connector of the PO8525 the RS-485 of the DU350 or DU351 communication interface must be connected to. The PO8525 that are located on the end of the network must be configured as termination, the remaining units as splitters.

**FBs-USB-232M9:** this module is used to connect the RS-232 communication interface of the DU350 and DU351 controllers to a standard USB interface of a micro-computer.

# 3. Configuration

DU350 and DU351 controllers are configured and programmed by the MasterTool IEC software. The established settings specify the behavior and use of peripherals as well as the special characteristics of the controllers. The programming stands for the application developed by the user, which is also called “application”.

## General

DU350 and DU351 controllers have some general settings which are carried out by writing into some special operands. These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable (simply by employing the name of the operand in any POU of the project).They are:

**CONTRAST** = contrast percentage [0 – 100%];

**BACKLIGHT** = backlight time [0 – 255s].

The names of the operands and how to access them are better described in the special operands list, located in the section “Maintenance” - “Diagnostics” - “Reserved Operands List”.

## Bus

DU350 and DU351 controllers have Inputs and Outputs described in its configuration tree as bus. In order to access the configuration tree, open the MasterTool IEC and click “PLC Configuration”, which is located in the “Resources” tab. Then, the module “PLC Configuration” must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be viewed and configured.

The “Bus” module may be accessed by clicking on “+” to expand its options.

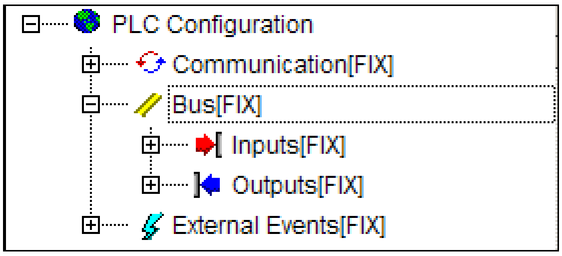


**Figure 3-1. Bus**

## Inputs

DU350 and DU351 controllers inputs are divided into:

- Digital Inputs
- Analog Inputs
- Fast Inputs

These options are shown in the following figure and will be described in the next chapter.

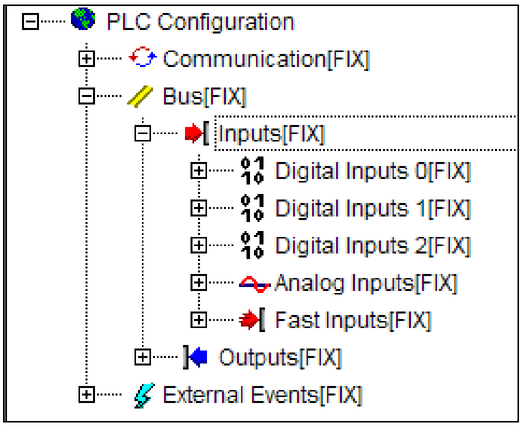


**Figure 3-2. Inputs**

## Digital Inputs 0/1/2

DU350 and DU351 controllers have 3 digital input blocks with a total of 20 digital inputs. Whenever there is a need to differentiate the fast inputs, the remaining inputs are defined in this manual as General Digital Inputs. The 3 blocks are divided into the following structure:

| | |
|---|---|
| **Number of inputs** | 20 digital inputs divided into insulation groups:<br>I00..I08 - 9 inputs – Group 0<br>I10..I18 - 9 inputs – Group 1<br>I20..I21 - 2 inputs – Group 2 |

**Table 3-1. Inputs Blocks Description**

The 3 inputs - I00, I01 and I02 – of Group 0, and the inputs I10, I11 and I12 of Group 1 may also be used as fast inputs. In case no fast input function is set up, all of the 20 inputs work as common digital inputs.

All digital inputs are previously assigned to the special operands described below:

| BLOCK 0 | |
|---|---|
| **I00** | Digital Input I00 |
| **I01** | Digital Input I01 |
| **I02** | Digital Input I02 |
| **I03** | Digital Input I03 |
| **I04** | Digital Input I04 |
| **I05** | Digital Input I05 |
| **I06** | Digital Input I06 |
| **I07** | Digital Input I07 |
| **I08** | Digital Input I08 |
| **BLOCK 1** | |
| **I10** | Digital Input I10 |
| **I11** | Digital Input I11 |
| **I12** | Digital Input I12 |
| **I13** | Digital Input I13 |
| **I14** | Digital Input I14 |
| **I15** | Digital Input I15 |
| **I16** | Digital Input I16 |
| **I17** | Digital Input I17 |
| **I18** | Digital Input I18 |
| **BLOCK 2** | |
| **I20** | Digital Input I20 |
| **I21** | Digital Input I21 |

**Table 3-2. Digital Inputs Blocks Distribution**

These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The names of the operands and how to access them are better described in the special operands list, located in the section "Maintenance" - "Diagnostics" - "Reserved Operands List".

## Analog Inputs

DU350 and DU351controllers have 4 analog inputs. Each channel may be set up in 3 ways and the inputs are independent between the channels:

- Voltage: 0 to 10 Vdc
- Current: 0 to 20 mA
- Current: 4 to 20 mA

The analog inputs are available from the pins described in the following table:

| | |
|---|---|
| **Terminal block configuration** | AV0 – voltage input channel 0<br>AI0 –current input channel 0<br>AV1 – voltage input channel 1<br>AI1 – current input channel 1<br>C9 – common for inputs 0 and 1<br><br>AV2 – voltage input channel 2<br>AI2 – current input channel 2<br>AV3 – voltage input channel 3<br>AI3 – current input channel 3<br>C10 – common for inputs 2 and 3 |

**Table 3-3. Analog Inputs Pins Description**

The inputs configuration is made through the configuration tree. In order to access the configuration tree, open the MasterTool IEC and click "PLC Configuration", which is located in the "Resources" tab. Then the module "PLC Configuration" must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be configured.

The analog input settings are located at the "Bus" module. By expanding this module, the "Inputs" and "Outputs" modules will appear. In order to set up the analog inputs, the "Inputs" module must be expanded, and then the "Analog inputs" subnode.

In the option "Channel type", the user selects the type of input that will be employed in that channel. The options are: "Voltage: 0 to 10 Vdc", "Current: 0 to 20 mA", "Current: 4 to 20 m" or "Disabled Channel" as previously described.

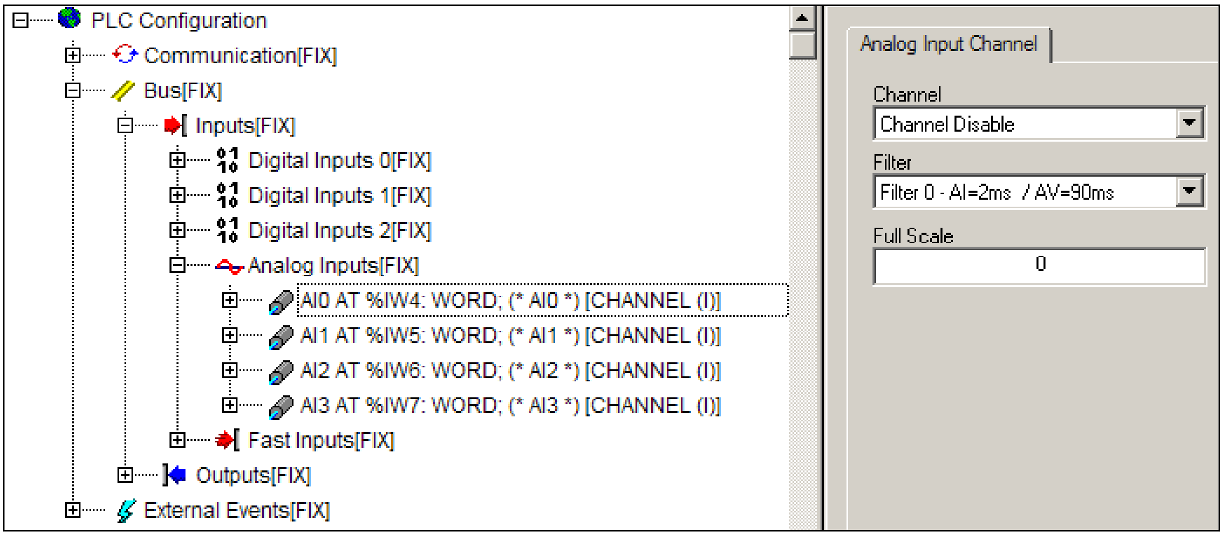


**Figure 3-3. Analog Inputs**

Following the input type configuration, the user must set up the filtering assigned to the respective analog input in the “Filtering” box, by choosing among the following filtering options: 2 ms, 90 ms, 1 s, 15 s (current inputs) or 90 ms, 140 ms, 1 s, 15 s (for voltage inputs). It is also necessary to set up the scale limit in the dialog box below, which may vary from 0 to 30000.

The analog inputs values are previously assigned to special operands. They are:

- **AI0** = Channel 0 value
- **AI1** = Channel 1 value
- **AI2** = Channel 2 value
- **AI3** = Channel 3 value

These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable . The names of the operands and how to access them are better described in the special operands list, located in the section “Maintenance” - “Diagnostics” - “Reserved Operands List”.

The analog inputs are updated every 60 ms, which means that the minimum time for the variance perception of a given input is of 60 ms. The stabilization time of the analog input signal depends on the selected filter. The update rate is independent from the selected filter as well as from the number of analog inputs used. It is recommended the use of the largest possible filter, according to the needs of either the control or monitoring system.

The corresponding value to the scale limit of analog inputs is configurable. This setting does not change the physical value of the scale limit, which is of 10 V for voltage analog inputs, and of 20 mA for current analog inputs, however, it changes the value which corresponds to the physical scale limit that will be read by the application. This function is useful to make the reading of analog inputs effortless for the user. For example: it might be interesting to have a configuration of a 10000 scale limit for a 0 - 10 V analog voltage input (in this case each reading unit corresponds to 1 mV). The sensitivity of analog inputs is fixed, therefore the increase in the value of the scale limit up to the peak value (30.000) does not lead to sensitivity improvement, although it may be useful for the reading data handling within a specific application. Employing a scale limit value lower than 4095 (12-bit) may cause a loss of resolution in the analog input.

## Fast Inputs

DU350 and DU351 controllers have two blocks of 32-bit counters (Block 0 and Block 1). Each block can operate as one bidirectional counter or as up to two unidirectional counters (Block 0: Counters 0 and 1; Block 1: Counters 2 and 3).

There are 6 terminal blocks of fast inputs, named as I00, I01, I02, I10, I11 and I12, used for counter handling. These terminal blocks may also be used as common digital inputs, in case the fast inputs are not used.

These 6 terminal blocks are configured as two counting blocks, called Block 0 and Block 1. Block 0 makes use of the terminal blocks I00 and I01 to perform the counting whereas the terminal block I02 for configurable input. Block 1, in turn, uses the terminal blocks I11 and I10 to perform the counting and the block I12 is for the configurable input.

Each one of the blocks can be configured independently so that the 2 counting terminal blocks perform the counting either in the bidirectional mode (using the 2 counting pins for only one counter) or in the unidirectional mode ( using 1 counting pin for each counter).

In case the block is used as a bidirectional counter, it will have only one counter. Only counter 0 is used for Block 0, while only counter 2 is used for Block 1.

The block configured as unidirectional block works with only one counter, and uses only one pulse terminal block, thus allowing the usage of the second terminal block as a common input.

Each of the blocks has an control input. This control input can be used to:

- Clear value
- Freeze
- Preset
- Hold

The control input can also be disabled to be used as a common digital input.

### *Fast Inputs Configuration*

The first step to configure the fast inputs is to open the configuration tree. In order to access the configuration tree, open the MasterTool IEC and click “PLC Configuration”, which is located in the “Resources” tab. Then, the module “PLC Configuration” must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be viewed and configured.

The fast input settings are located at the “Bus” module. By expanding this module, the “Inputs” and “Outputs” modules will appear. In order to set up the counters, the “Inputs” module must be expanded and then the “Fast inputs” subnode .The figure below illustrates this procedure:

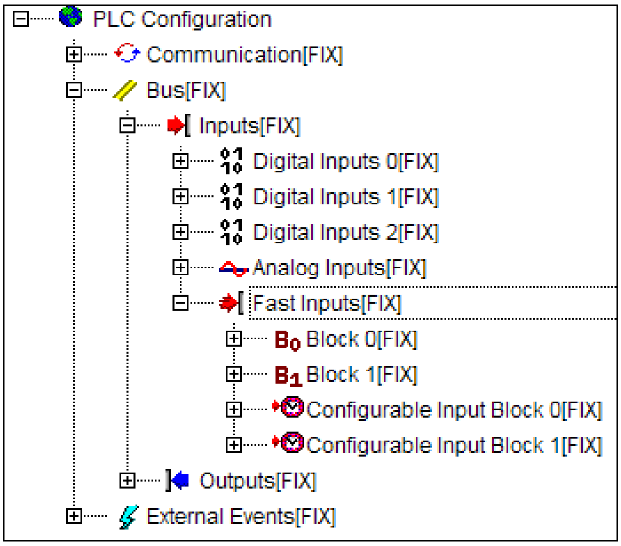


**Figure 3-4. Fast Inputs**

The counting Block 0 is used as an example, with the purpose of describing the setup processes, considering that the settings are also applied to counting Block 1.

The expansion of "Block 0" brings into view a module named "Disabled", due to the fact that no fast input configuration has been carried out in the controller. A click with the right button of the mouse on "Disabled", causes a pop-up to appear with the options "Replace element", "Calculate addresses" and "Copy". By selecting the "Replace element" option it is possible to visualize the "Bidirectional" and "Unidirectional" options for Block 0:

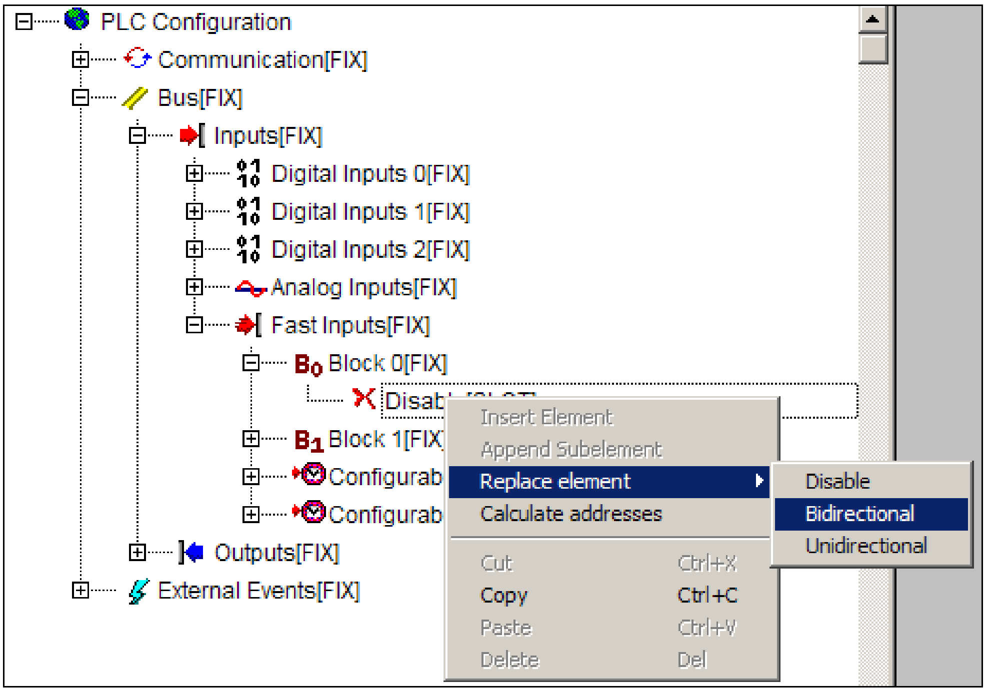


**Figure 3-5. Setting Up for Bidirectional Counter**

The registers that store the counting number of counters are previously assigned to special operands. These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The register that stores the counting number is called:

- **CNTx** = Counting value

Where "x" is the number of counter.

The names of the operands and how to access them are better described in the special operands list, located in the section "Maintenance" - "Diagnostics" - "Reserved Operands List".

Next, it is described the operation of each one of the configuration options.

*Bidirectional*

When Block 0 is selected as bidirectional, a window with the counting mode setting will appear on the right side of the screen. It enables the configuration of the bidirectional counter in the following modes:

**Mode 0: A increments, B decrements** – in this mode, a rising edge applied to channel A (I00 or I10) causes an increment of the counter value, while on channel B (I01 or I11) it causes a decrement in the counting value.

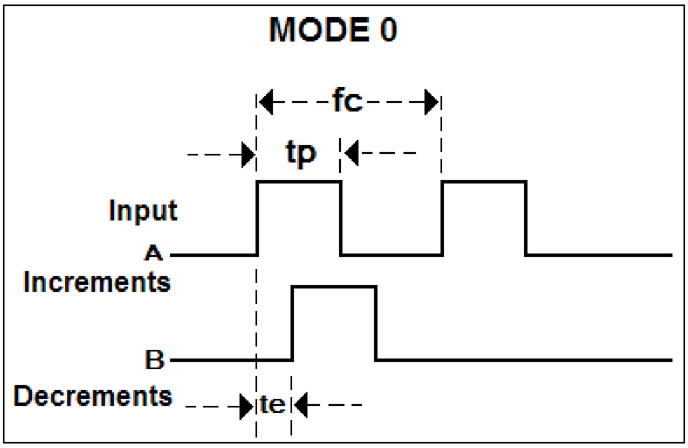


**Figure 3-6. Bidirectional Mode 0**

**cf** – Counting frequency

Block 0 cf ≤ 40 kHz

Block 1 cf ≤ 20 kHz

**pd** – Pulse duration pd > 10 μs

**te** – Minimum space te > 10 μs

**Mode 1: A counting, B direction** – in this mode, input A (I00 or I10) is responsible for counting, whereas the direction is determined by input B (I01 or I11).

If the input B signal is at logic level 0 (during a rising edge in input A), the counter is incremented. If B is at logic level 1 (during a rising edge in input A) the counter is decremented.

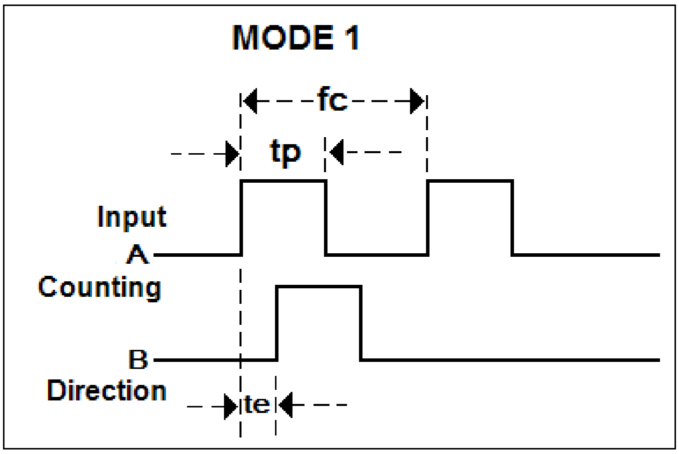


**Figure 3-7. Bidirectional Mode 1**

**fc** – Counting frequency

Block 0 cf ≤ 40 kHz

Block 1 cf ≤ 20 kHz

**pd** – Pulse duration pd > 10 µs

**te** – Minimum space te > 10 µs

**Mode 2 and Mode 3: Square 4x** and **Square 2x**– In these modes the counting units decode the input signals in squaring according to the usual standard provided by optical transducers.

The direction of counting is obtained from the phase relationship between signals (the counting is incremented if the pulse in counting input A is ahead in relation to the pulse in counting input B, and it is decremented if pulse B is ahead in relation to pulse A), while counting pulses are related to the status transitions.

In mode 2 (4 x squaring) 4 counting pulses are generated per period of input signals (rising and falling edges of both counting inputs), where a count is generated at each pulse. Therefore, the employment of an input signal of 20 kHz will cause the counter to be incremented (or decremented) at a frequency of 80 kHz.

In mode 3 (2x squaring) 2 counting pulses are generated per period of input signals (rising and falling edges of only one counting input; the second counting input is used to set the counting direction), where a count is generated at each pulse. Therefore, an input signal of 40 kHz will cause the counter to be incremented (or decremented) at a frequency of 80 kHz.

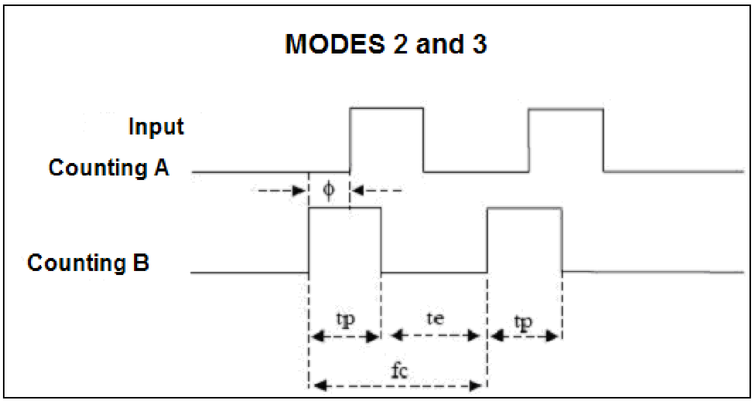


**Figure 3-8. Bidirectional Modes 2 and 3**

**fc** – Counting Frequency

**Mode 2**

Block 0 cf ≤ 20 kHz

Block 1 cf ≤ 10 kHz

**pd** – Pulse duration pd > 20 µs

**te** – Minimum space te > 20 µs

$\phi$ – Phase relation $\phi = 90 \pm 10°$

**Mode 3**

Block 0 cf ≤ 40 kHz

Block 1 cf ≤ 20 kHz

**pd** – Pulse duration pd > 10 µs

**te** – Minimum space te > 10 µs

$\phi$ – Phase relation $\phi = 90 \pm 10°$

The following figure points out the set up window of the bidirectional counting modes.

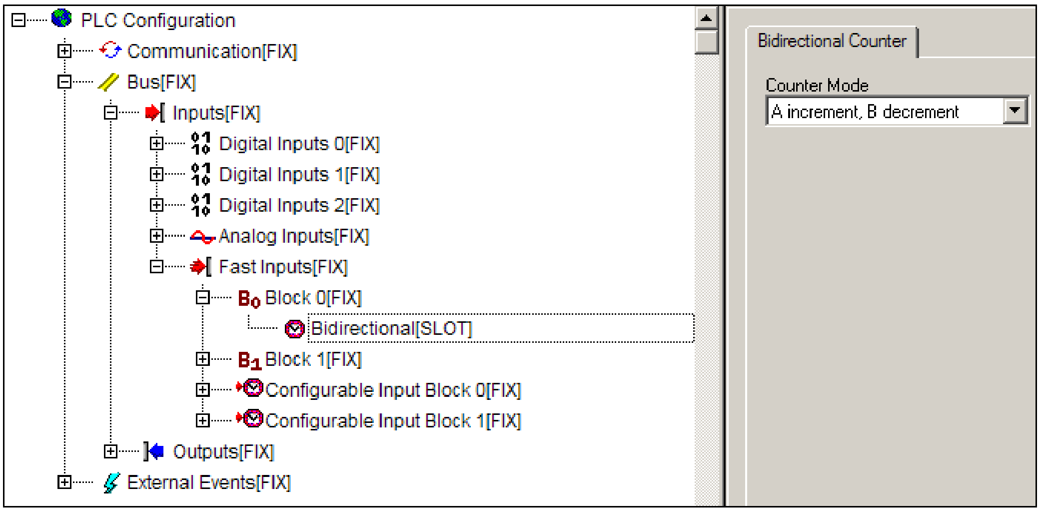


**Figure 3-9. Bidirectional Counter**

*Unidirectional*

In case the block is selected as unidirectional, two subnodes will be attached (the "Counter 0" module and the "Counter 1" module). By clicking on one of the counters, it is possible to configure the counting mode, in the field "Counting Mode" which is located on the right side of the screen. Unidirectional counters may be configured with the following counting modes:

- Counting Up mode
- Counting Down mode

If one of the block counters is not used, one of them may be disabled by clicking on it with the right button of the mouse, then selecting the option "Replace element", and clicking on "Disabled".

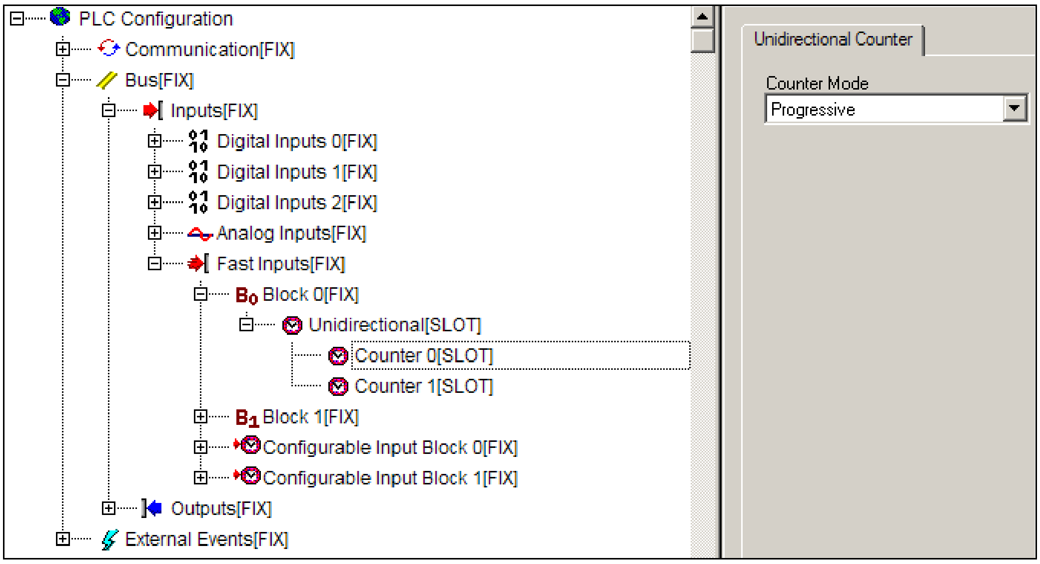


**Figure 3-10. Unidirectional Counter**

*Configurable Inputs*

After configuring the counting type, it is possible to configure the block configurable input to perform a specific command selected by the user. This setting is done in "Block x Configurable input" module, where "x" is the block number. By expanding this module, the configurable input is disabled; to enable it, simply select the function that the configurable input will have for the respective counter. This setting is done on the upper-right corner of the screen when selecting this counter. Configurable inputs cannot be used for Counters 1 and 3. If Block 0 is configured as bidirectional, the counter 0 is used. For Block 1the corresponding counter is Counter 2.

The configurable input of Block 0 is connected to the terminal block I02 and the configurable input of Block 1 is connected to the terminal block I12.

After defining which counter will be used, it is possible to select the command type which will be assigned to the configurable input. The command to be performed must be selected in the dialog box "Command". The configurable input may be set up as:

- **Clear** to load the counter with the value 0
- **Freeze** to pause the counter
- **Preset** to load the counter with the value which is in the operand CNTx_PRESET, where "x" corresponds to the counter index
- **Hold** to copy the counter value onto the operand CNTx_HOLD, where "x" corresponds to the counter index

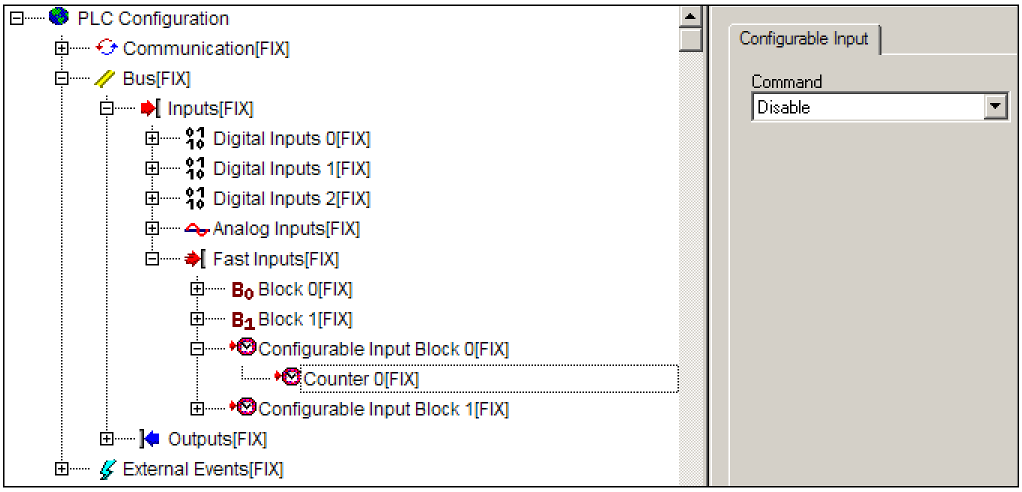


**Figure 3-11. Command Inputs**

The registers of the configurable inputs are previously assigned to special operands. These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The registers are called:

- **CNTx_PRESET** = Preset value
- **CNTx_HOLD** = Hold value
- **CNTx_CMP0** = Comparator 0 value
- **CNTx_CMP1** = Comparator 1 value

Where "x" is the number of counter.

In case the user does not want to use the configurable input terminal block, the commands may also be performed via software. In order to do this, employ the command byte, which does not require any configuration in the configuration tree.

The registers of the commands are previously assigned to special operands. These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The registers are called:

- **CNTx_CLR** = Clear the counting register
- **CNTx_STOP** = Disables the counter counting (freeze the counter value)
- **CNTx_LOAD** = Load PRESET
- **CNTx_AMG** = Counting sampling (HOLD)
- **CNTx_OVER** = Clear overflow and underflow status bits

Where "x" is the number of counter.

*Counters Diagnostics*

The diagnostics related to the counters are described in the chapter "Diagnostics"- "Fast Inputs".

*Counter Comparator Outputs*

The use of fast output comparators is described in the chapter: "Settings" - "Outputs" - "Fast Outputs" - "Counter Comparison Fast Outputs". The use of common comparator outputs is described in the comparison chapter: "Settings" - "Outputs" - "Counter Comparison Common Output".

# Outputs

DU350 and DU351 controllers outputs are divided into:

- Digital outputs (both can operate as counter comparator outputs)
- Analog outputs
- Fast outputs (both can be used as digital inputs, counter comparator outputs, PWM/VFO outputs or PTO outputs)

These options are shown in the following figure and are described in the course of this chapter.

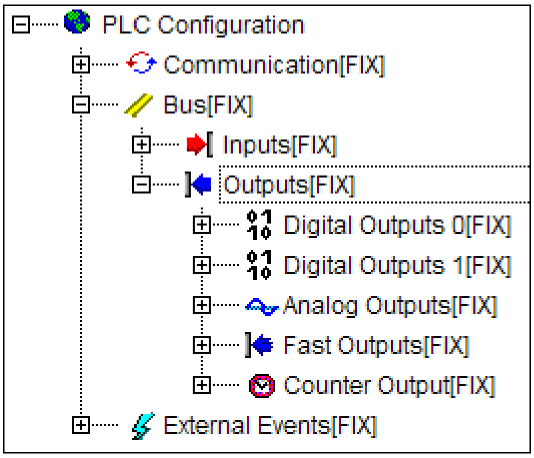


**Figure 3-12. Outputs**

## Digital Outputs

DU350 and DU351 controllers have 2 digital output blocks with a total of 16 digital outputs divided into 2 insulation groups. DU350 has 16 transistor outputs (2 fast and 14 normal). DU351 has 14 relay outputs and 2 transistor fast outputs. The outputs distribution can be seen in the table below.

| | DU350 |
|---|---|
| **Number of common outputs** | 16 transistor digital outputs divided into 2 insulation groups:<br>Q00 to Q07 – 8 outputs – Group 0<br>Q10 to Q17 – 8 outputs – Group 1 |

**Table 3-4. DU350 Common Outputs**

| | DU351 |
|---|---|
| **Number of outputs** | 2 transistor digital outputs:<br>Q00 to Q01 – 2 outputs – Group 0<br>14 relay digital outputs divided into 2 insulation groups:<br>Q02 to Q07 – 6 outputs – Group 0<br>Q10 to Q17 – 8 outputs – Group 1 |

**Table 3-5. DU351 Common Outputs**

The Q00 and Q01outputs can be used as normal outputs, fast outputs (PWM/VFO or PTO) or counter comparison outputs. The Q02 and Q03 outputs can be configured as normal outputs or counter comparator outputs. If such outputs are being used for a special function, it is not possible to use them as a common digital output.

All digital outputs are previously allocated to special operands. They are:

| BLOCK 0 | |
|---|---|
| Q00 | Digital Output Q00 |
| Q01 | Digital Output Q01 |
| Q02 | Digital Output Q02 |
| Q03 | Digital Output Q03 |
| Q04 | Digital Output Q04 |
| Q05 | Digital Output Q05 |
| Q06 | Digital Output Q06 |
| Q07 | Digital Output Q07 |
| BLOCK 1 | |
| Q10 | Digital Output Q10 |
| Q11 | Digital Output Q11 |
| Q12 | Digital Output Q12 |
| Q13 | Digital Output Q13 |
| Q14 | Digital Output Q14 |
| Q15 | Digital Output Q15 |
| Q16 | Digital Output Q16 |
| Q17 | Digital Output Q17 |

**Table 3-6. Digital Outputs Operands**

These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The names of the operands and how to access them are better described in the special operands list, located in the section "Diagnostics" - "Reserved Operands List".

## Analog Outputs

DU350 and DU351 controllers have 2 analog outputs. Each channel can be configured individually as an output of:

- Voltage: 0 to 10 Vdc
- Current: 0 to 20 mA

The analog outputs are available from the pins described in the following table.

| | |
|---|---|
| **Terminal block configuration** | C3 –common for channel 0 output<br>AO0 – analog output channel 0<br>C4 – common for channel 1 output<br>AO1 – analog output for channel 1 |

**Table 3-7. Analog Outputs Pins Description**

The analog outputs setup is made via the configuration tree. In order to access the configuration tree, open the MasterTool IEC and click "PLC Configuration", which is located in the "Resources" tab. Then the module "PLC Configuration" must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be configured.

The analog output settings are located at the "Bus" module. By expanding this module, the "Inputs" and "Outputs" modules will appear. In order to set up the analog outputs, the "Outputs" module must be expanded, and then the "Analog outputs" subnode.

In the option "Channel type", the user selects the output type to be used in that channel. The outputs may be set up as: current analog outputs (0 - 20 mA), voltage analog outputs (0 - 10 V) or as "Disabled Channel".

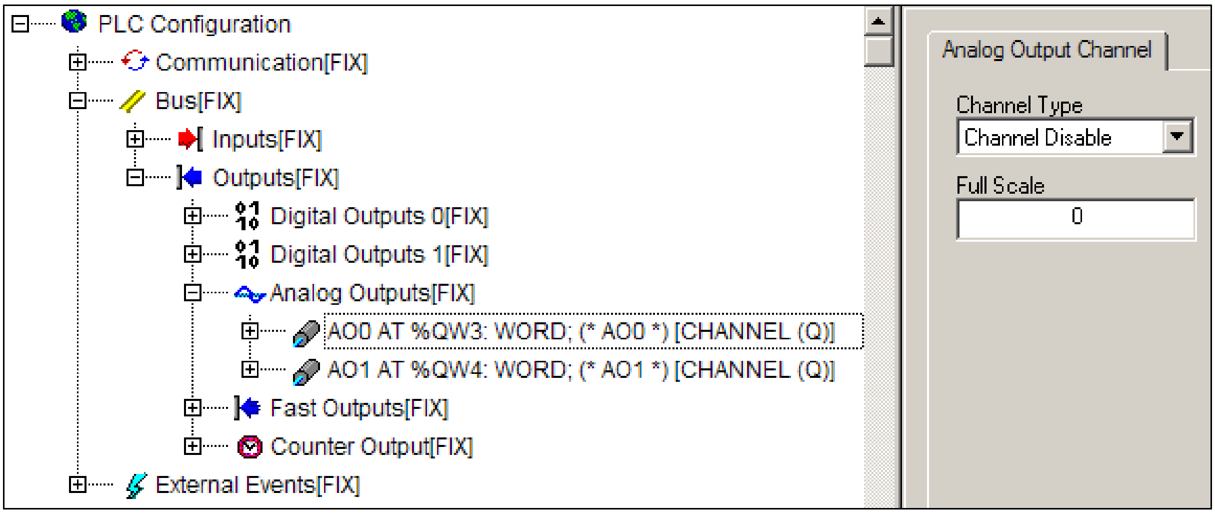


**Figure 3-13. Analog Outputs**

After the output type configuration, the user must set up the scale limit in the dialog box below, which may vary from 0 to 30000.

The analog outputs values are previously assigned to special operands. They are:

- **AO0** = Channel 0 value
- **AO1** = Channel 1 value

These special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable. The names of the operands, and how to access them are better described in the special operands list, located in the section "Diagnostics" - "Reserved Operands List".

The corresponding value to the scale limit of analog outputs is configurable. This setting does not change the physical value of the scale limit, which is of 10 V for voltage analog inputs, and of 20 mA for current analog inputs, however, it changes the value which corresponds to the physical scale limit that will be read by the application. This function is useful to make the reading of analog inputs effortless for the user. For example: it might be interesting to have a configuration of a 100 scale limit for a 0 V 10 V analog voltage input (in this case each reading unit corresponds to 1 mV). The sensitivity of analog inputs is fixed, therefore the increase in the value of the scale limit up to the peak value (30.000) does not lead to sensitivity improvement, although it may be useful for the reading data handling within a specific application. Employing a scale limit value less than 4095 (12-bit) may cause a loss of resolution in the analog input.

## Fast Outputs

DU350 and DU351 controllers have 2 (two) fast outputs. They are located in the connector of digital output insulation group 0.

The fast outputs are found in the pins described in the table below:

| | |
|---|---|
| **Terminal block configuration** | Q00 to Q01 – fast output 0 to 1 of insulation group 0<br>C5 – Reference (0 V), insulation group 0 (shared with the Q02 to Q07 transistor outputs(DU350))<br>C6 – Power supply, insulation group 0 ( shared with the Q02 to Q07 relay outputs (DU351)/transistor(DU350)). Maximum voltage 30 Vdc. |

**Table 3-8. Fast Outputs Pins Description**

Q00 and Q01 fast outputs can be configured as:

- PTO ("Pulse Train Output")
- VFO ("Variable Frequency Output")

- PWM ("Pulse Width Modulation" )
- Counter comparison fast outputs

In case none of the special settings described above is used, the Q00 and Q01 fast outputs may be utilized as a transistor normal output.

**WARNING:**
**A shielded cable is recommended for fast outputs, in the event of lengths greater than 1 meter or when other cables are close to the fast output cable.**

The configuration of the fast outputs is made through the configuration tree. In order to access the configuration tree, open the MasterTool IEC and click "PLC Configuration", which is located in the "Resources" tab. Then the module "PLC Configuration" must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be configured.

Fast outputs settings are located on the "Bus" module. By expanding this module, the "Inputs" and "Outputs" modules will appear. In order to set up the fast outputs, the "Outputs" module must be expanded, and then the "Fast outputs" subnode.

The 2 fast outputs are shown, thus enabling the setup in 4 distinct modes:

- PTO
- VFO/PWM
- Comparator counter 0 or 2
- Comparator counter 1 or 3

In addition, in the disabled mode, the outputs work as common digital outputs. The following figure exhibits the description of each one of the fast output settings.

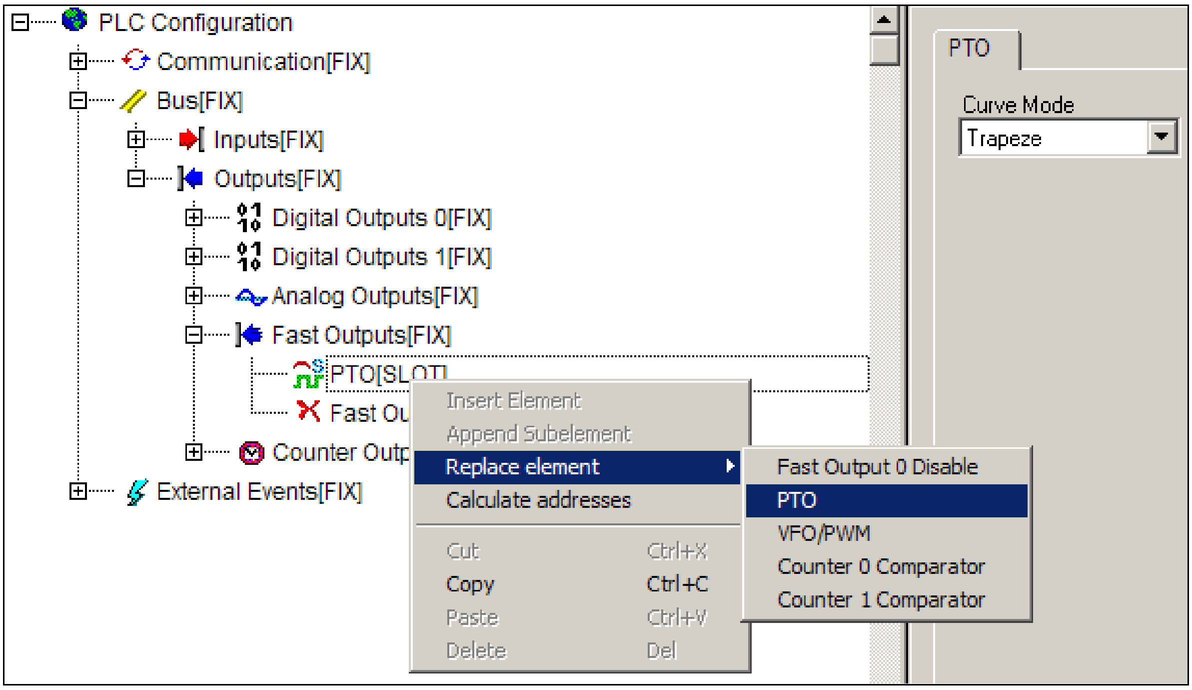


**Figure 3-14. Fast Outputs**

*PTO*

The PTO output generates a pulse train (duty cycle of 50%) containing the total pulse number, number of pulses in acceleration/deceleration and frequency configurable via software.

Below, there is an example of the PTO output operation configured to generate a total of 20.000

pulses (5.000 during acceleration/deceleration). The vertical axis represents the frequency, and the horizontal axis represents the time.

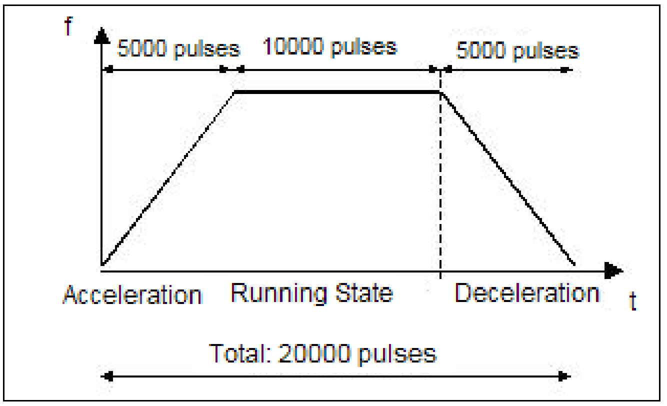


**Figure 3-15. PTO Output Example**

In the acceleration phase, the output frequency goes from 0 Hz up to the frequency defined by the user. From that point on, the steady state starts, and here is where the complementary total number of pulses, that was configured to those pulses of acceleration/deceleration, is generated.

After completing the number of pulses of the steady state, the fast output enters in the deceleration phase, where the pulse train frequency varies from the default value up to 0 Hz, then completing the fast output action. Thus, the new position, which is represented by the total number of user-defined pulses, is reached.

The main use of PTO outputs is on projects involving positioning control. Due to its characteristic of generating only a specified number of programmed pulses, this output is of great value in this type of application.

Another important aspect of PTO outputs is that they enable the start-up of motors with smooth acceleration. This implies that the motor does not accelerate in a single step, going from resting directly to the nominal speed, but it accelerates step-by-step to achieve the target speed. Therefore, it is possible to run huge systems, taking them out of the resting status smoothly.

Next, it is shown how the function parameters may be calculated as from the project data.

### *Acceleration Profiles*

Profile is the way in which the variation of the output signal frequency occurs in relation to time. The PTO outputs may be configured with a trapezoidal-type profile or an s-type profile.

The PTO outputs parameters are: total pulse number; number of pulses in acceleration/deceleration; permanent frequency. These parameters do not require configuration at each new PTO output trigger. At the triggering of a new pulse train, if the operand of number of pulses in acceleration/deceleration (**Fx_PLS_RMP**) or the operand of frequency scheme (**Fx_FREQ**) is changed, the controller recalculates the distribution of pulses for a new ramp generation. The time used for their calculation is inversely proportional to the number of ramp pulses used. This time may vary from 1ms to 3,5 ms.

### *Trapezoidal Type Profile*

In this type of profile the motor is accelerated in linearly, which means in constant acceleration.

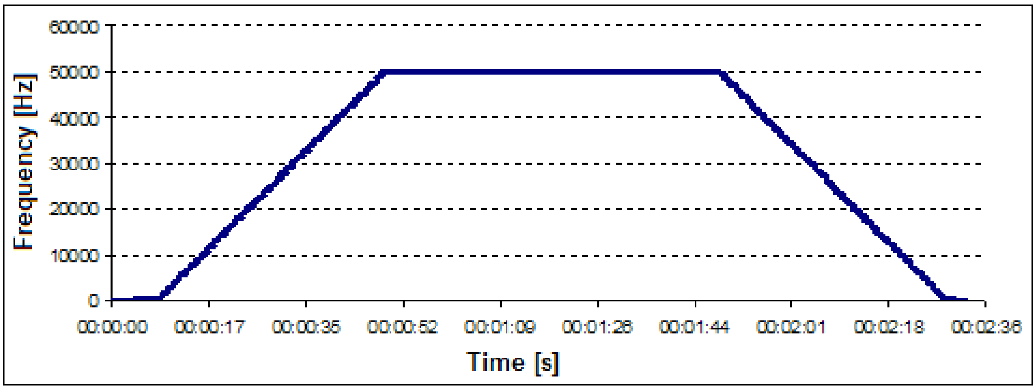


**Figure 3-16. Graph Generated by PTO Output Using Trapezoidal Type Profile**

The chart in Figure 3-17 shows an actual curve acquisition generated by a PTO output configured with the trapezoidal profile, 5000000 total pulses, 1000000 pulses in acceleration/deceleration and 50 kHz steady state frequency.

To calculate the number of frequency steps to be performed during acceleration/deceleration for a trapezoidal profile, the following expression is used:

$$divs \approx \sqrt{2 * Fx\_PLS\_RMP}$$

Where the "divs" integer portion is the number of ramp steps, and Fx_PLS_RMP is the number of pulses in the ramp (acceleration/deceleration) configured by the user.

Under the above expression, and with values that vary from 0 to 10000 pulses in acceleration/deceleration ramps, it is obtained an amount of 0 to 100 steps in the ramp. The amount of steps in the ramp will be limited to 100 for values greater than 10000 ramp pulses, however, the number of pulses by step will grow proportionately to the number of ramp pulses.

Due to rounding that occurs during the calculation of the profile, some pulses of the ramp may not happen during the ramp, and are offset during the steady state. In the worst cases the error by step will be of one pulse.

*S type Profile*

The value of acceleration is smaller at both the beginning and the end of the profile, being at its peak at mid-point, and 3.6 times higher than the acceleration of an equivalent trapezoidal-type profile. The acceleration and deceleration curves are symmetrical.

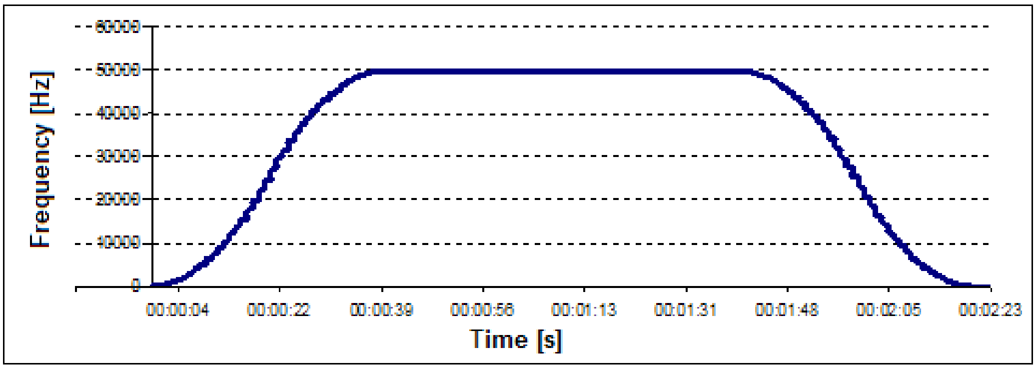


**Figure 3-17. Graph Generated by a PTO Output, Using the S Type Profile**

The chart in Figure 3-17 shows an actual curve acquisition generated by a PTO output configured with the S-type profile, 20000000 total pulses, 5000000 pulse acceleration/deceleration and 50 kHz of steady state frequency.

To calculate the number of frequency steps to be performed during acceleration/deceleration for a trapezoidal profile, the following expression is used:

$$divs^3 + 2 * divs^2 + divs + 4 * Fx\_PLS\_RMP = 0$$

Where the integer portion of the complex roots of “divs” is the number of ramp steps, and Fx_PLS_RMP is the number of pulses in the ramp (acceleration/deceleration) configured by the user.

Under the above expression, and with values that vary from 0 to 10000 pulses in acceleration/deceleration ramps, it is obtained an amount of 0 to 256000 steps in the ramp. The amount of ramp steps will be limited to 100 for values greater than 256000 ramp pulses, however, the number of pulses by step will grow proportionately to the number of ramp pulses.

Due to rounding that occurs during the calculation of the profile, some pulses of the ramp may not happen during the ramp, and are offset during the steady state. In the worst cases the error by step will be of one pulse.

*Soft Stop*

Each PTO output has a smooth stop command, which is also known as softstop. The soft stop is the early slowdown of the system triggered by a command, thus causing a soft stop before the end of the pulse train.

A soft stop can be performed at any time of the pulse train, including during acceleration ramp. When a softstop is generated during deceleration, the pulse train will not change its behavior because the system is already slowing down. See Figure 3 18.

The Fx_PTO_SOFTSTOP operand is responsible the soft stop.

In case of error in the number of pulses during acceleration due to rounding during a soft stop, the Fx_PTO_REG status can be triggered for some cycles while this compensation is being executed even if the target speed has not been reached yet.

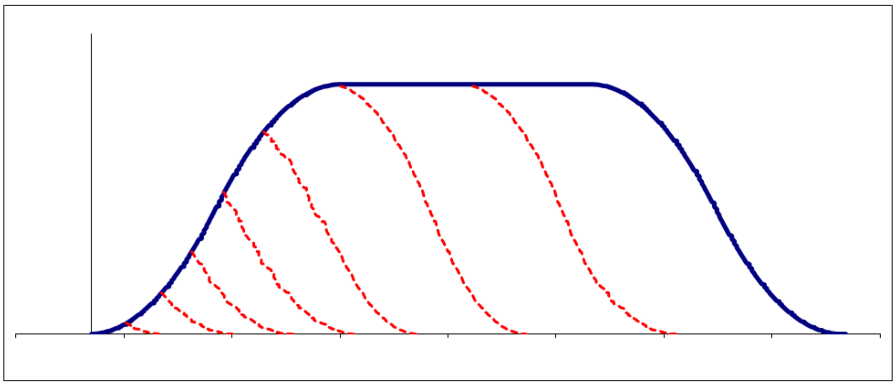

**Figure 3-18. Complete Pulse Train (blue) and Soft Stops Examples (red)**

*Internal Pulse Counter*

Each fast output has two pulse counters, one relative and one absolute. Both counters show information about the number of pulses generated by a PTO-type output in real-time.

The relative counter is reset at each new trigger of the pulse train, and its value grows constantly along the pulse train.

The absolute counter holds the absolute value of pulses generated by the PTO output and its value rises or falls according to the Fx_PTO_CNT_DIR command. The value of this counter is reset via Fx_PTO_CNT_CLR command thus defining a new reference for this counter.

The PTO counting and control pulses registers are:

Fx_PTO_CNT_REL: Fx output relative pulse counter. This counter has only the positive orientation, and is reset at each new trigger of a pulse train.

Fx_PTO_CNT_ABS: Fx output absolute pulse counter. This counter is incremented or decremented depending on the status of the Fx_PTO_CNT_DIR operand.

Fx_PTO_CNT_DIR: Defines the direction of the absolute counter. If it is FALSE the counter counts in the positive direction; if it is TRUE the counter counts in the negative direction. This register is read once at every trigger, and the changes in its value during a pulse train do not influence the counting direction.

Fx_PTO_CNT_CLR: Resets the Fx output absolute pulse counter.

Fx_PTO_CNT_CMP0: Comparator 0 of Fx output absolute pulse counter. This operand must be configured with the desired comparison value.

Fx_PTO_CNT_CMP1: Comparator 1of Fx output absolute pulse counter. This operand must be configured with the desired comparison value.

Fx_PTO_CNT_DG: Variable containing the PTO pulse counter diagnoses. The diagnoses are: Bit 0 – Fx_PTO_CNT_MAX_CMP0 and Bit 1 – Fx_PTO_CNT_MAX_CMP1.

Fx_PTO_CNT_MAX_CMP0: Signals that the Fx absolute pulse counter is greater than the comparator 0.

Fx_PTO_CNT_MAX_CMP1: Signals that the Fx absolute pulse counter is greater than the comparator 1.

### *Calculation of the Frequency State*

In general, the motors which are used for positioning have a specified resolution. This means that they specify the number of steps necessary to complete a round. One of the step motor parameters, for instance, is the angle developed at each step. Thus the number of steps per round may be calculated by dividing 360 by this amount.

$$p = \frac{360}{\theta}$$

where: *p = number of steps per revolution*
*θ = angle developed at each step*

On the other hand, the servos hold this data directly in their driver features.

To calculate the frequency state it is necessary to know the nominal motor speed. Step motors, in general, do not have this parameter properly defined. Manufacturers define a torque curve for speed. Thus, it is up to the designer to identify the application type as well as the torque speed, and in accordance to this, define the best target speed. In servomotors this feature is clearly defined. In a torque x speed chart it is evident a speed range in which the torque remains constant. These speeds are usually defined in RPM (revolutions per minute).

With this data it is possible to calculate the output frequency state as:

$$f = \frac{\mathrm{p} \times \mathrm{V}}{60}$$

where: *p = number of steps per round*
*f =PTO output state frequency (Hz)*
*V = motor target speed (RPM)*

### *Calculation of Acceleration Parameters*

The maximum acceleration to which the motor shall be subjected is directly proportional to the motor torque and inversely proportional to the motor inertia plus the load inertia.

$$A = 95.5 \times 10^6 \times \frac{T}{J}$$

where: *A = acceleration (rpm/s)*
*T = motor torque*
*J = total inertia in g.m² = Jm (motor inertia) + Jc (load inertia)*

For a linear acceleration (trapezoidal profile), the acceleration time may be calculated as follows:

$$t = \frac{V}{A}$$

where: *t = acceleration time*

This is the required time calculated, taking into consideration the motor parameters, in order to get maximum acceleration. This is also the minimum time in which the motor must remain in the acceleration phase. From this value on, it is possible to calculate the number of steps during the acceleration phase. This parameter is passed on to the Fx_PLS_RMP register.

$$N = \frac{5.5 \times f \times t}{10}$$

where: *N = number of pulses in acceleration/deceleration phase*

For an S-type profile, the maximum acceleration is 3.6 times higher than the acceleration required for the trapezoidal profile. Thus, for an S profile, the acceleration time must be 3.6 times greater than the one calculated for a trapezoidal profile in the same frequency state, so that maximum acceleration does not exceed the acceleration calculated according to the motor parameters.

### *PTO Configuration*

To configure the PTO output, after following the steps described above, right-click on the subnode “Fast Output 0 Disabled ” (output 0). A window appears with the options “Replace element ” “Calculate address ” and “Copy”. Within the item “Replace element” there are the following options: “Fast Output 0 Disabled ”, “PTO”, “VFO/PWM”, “Comparison Counter 0” and “Comparison Counter 1”.

By clicking on “PTO”, the “PTO” tab is shown on the right side of the page, where the configuration of the PTO curve type is done (“Trapeze” or “ S curve”).

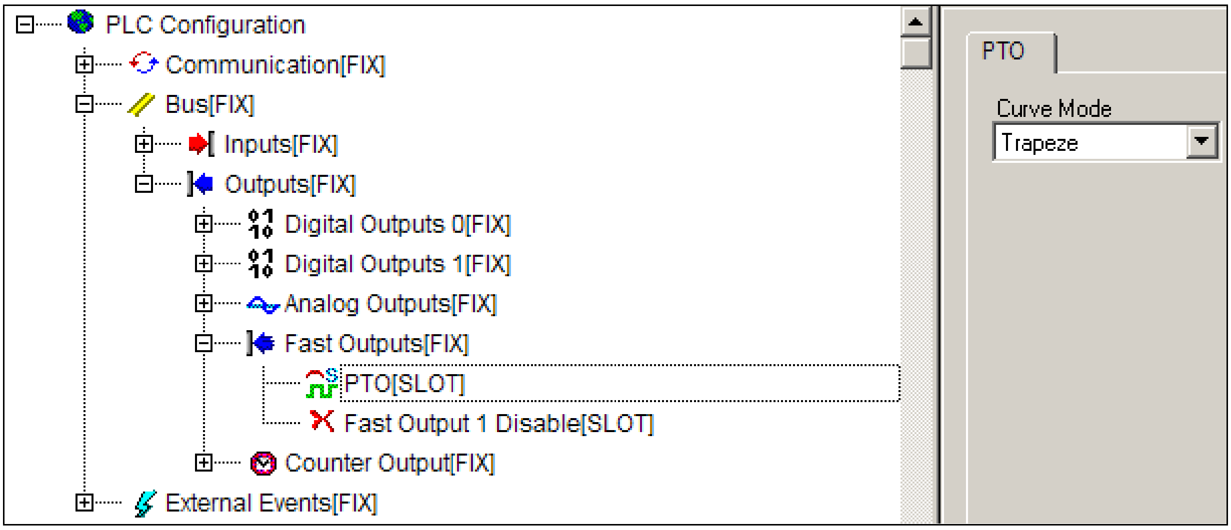


**Figure 3-19. PTO**

It is also necessary to configure some PTO special operands. They are:

- **Fx_FREQ** = frequency value for PTO/VFO/PWM [1 → 50.000] Hz
- **Fx_PLS_TOT** = total pulse value for PTO [1 → 4294967295]
- **Fx_PLS_RMP** = ramp pulse value for PTO [1 → ((PLS_TOT-1)/2)]

Where “x” is the output number.

In PTO outputs the duty cycle is fixed in 50%.

The special operands for frequency, total pulse and ramp pulse are previously mapped into a specific memory area. Thus, they can be used as a global variable. The names of the operands is best described in the special operands list, located in the section "Diagnostics" - "Reserved Operands List".

In order to start or stop the pulse generation, two special operands must be used:

- **Fx_PTO_START** = Triggers the pulse train (PTO) in the correspondent fast.
- **Fx_PTO_STOP** = Stops the pulse train (PTO) in the correspondent fast. If there is a trigger in standby, it will be rejected.
- **Fx_PTO_SOFTSTOP** = Starts the deceleration curve to perform a smooth stop. If this bit is set during the acceleration, a symmetrical deceleration (previous step) will be generated. If the bit is set during the steady state, the deceleration curve will be advanced.

Where "x" is the output number. The logic value 1 enables the operand function.

When the Fx_PTO_START,Fx_PTO_STOP and Fx_PTO_SOFTSTOP are enabled, the system identifies them, performs a reset (logical value 0) and executes the command.

Fx_PTO_STOP is prior to Fx_PTO_SOFTSTOP. Thus, when both of them go through reset in the same cycle, only the Fx_PTO_STOP command will be followed, and an abruptly stop will occur.

During the execution of a pulse train it is possible to write on the Fx_PTO_START operand and prepare a new pulse train which will run at the end of the first trigger. It is also possible to put a single new trigger in the line. This trigger will use the frequency values, the number of total pulses and ramp pulses which are configured at the time the new trigger is executed.

**WARNING:**
**The Fx_PTO_STOP command stops the pulse train ,which was started by the Fx_PTO_START command, abruptly.**

*VFO/PWM*

The variable frequency fast outputs (VFO), and the pulse width modulation (PWM) are used primarily for interconnection with frequency-voltage converters allowing, for example, the implementation of two additional analog outputs. The difference between these outputs is a parameter to be controlled. In VFO the duty cycle is constant and the frequency is variable whereas in PWM modulation the frequency is constant and the duty cycle is variable.

The following figure illustrates output behavior in both modes:

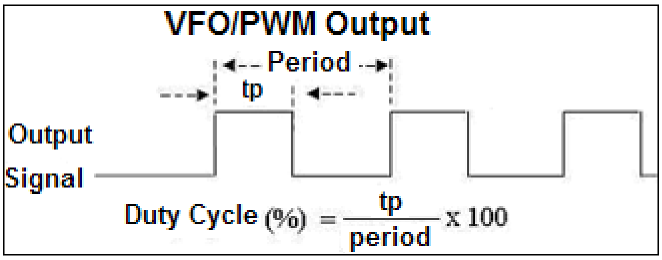


**Figure 3-20. VFO/PWM**

To configure the VFO/PWM output, expand the subnode "Fast Output" (previously described), right-click on the subnode "Fast Output 0 Disabled " (output 0). Then, select the options "Replace element"and finally, the option "VFO/PWM".

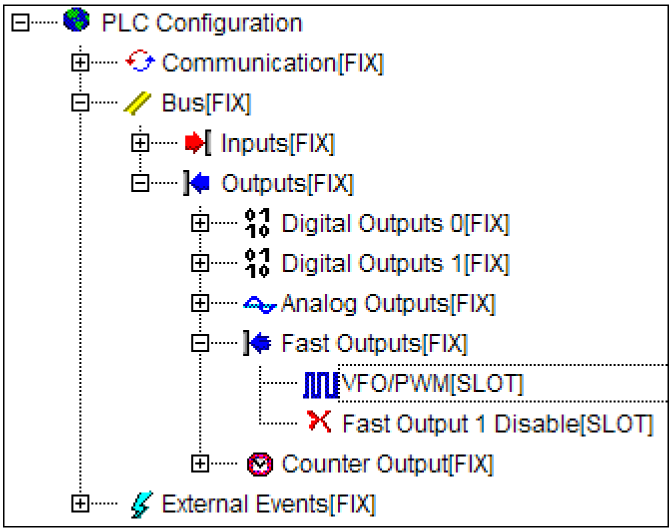


**Figure 3-21. Configuring to VFO/PWM**

In addition, the required parameters to the signal generation must be configured. These parameters are assigned to special operands:

- **Fx_FREQ** = Frequency value for PTO/VFO/PWM [1 – 50000] Hz
- **Fx_DUTY** = Duty Cycle value for VFO/PWM [0 - 100] %
- **Fx_DUTY_HR** = High-resolution Duty Cycle value for VFO / PWM [0.00 to 100.00]%
- **Fx_DUTY_SRC** = Defines the duty parameter source. Fx_DUTY or Fx_DUTY_HR

Where “x” is the output number.

To use the high-resolution duty it is necessary to set Fx_DUTY_SRC first, and choose Fx_DUTY_HR as the source of duty

The high-resolution duty allows for the use of a duty with a resolution up to two decimal places.

The special operands for frequency and duty cycle configuration are previously mapped into a specific memory area. Thus, they can be used as a global variable . The names of the operands is best described in the special operands list, located in the section “Diagnostics” - “Reserved Operands List”.

In order to either start or stop the PTO/PWM pulse generation, the following special operands must be used:

- **Fx_VFO** = 0 (FALSE) → Disables the VFO/PWM on the corresponding fast output

  1 (TRUE) → Enables the VFO/PWM on the corresponding fast output

Where “x” is the output number.

*Counters Comparator Fast Output*

To configure the counter comparator output, expand the subnode “Fast Output” (previously described), right-click on the subnode “Fast Output 0 Disabled ” (output 0). Then, select the options “Replace element ”, and finally, the option “Counter Comparator 0” or “Counter Comparator 1”. The fast output 1 configuration is performed in the same way, though the counter comparators 2 or 3 may be linked to this output.

After clicking on “Counter Comparator 0”, select the comparison logic to be employed (on the right side of the MasterTool IEC programmer). The following options are available to configure the comparison type:

- Counter > Comparator 0
- Counter < Comparator 1

- Comparator 0 < Counter < Comparator 1

When the comparator holds the logic value 1 (comparison is TRUE), the fast output is set up to 1 (high level). When the logic value is 0 (comparison is FALSE), the fast output is set up to 0 (low level). The comparators value is configured in special operands. They are:

- **CNTx_CMP0** = Comparator 0 value
- **CNTx_CMP1** = Comparator 1 value

Where "x" is the output number.

The Comparator 0 and Comparator 1 special operands are previously mapped into a specific memory area. Thus, they can be used as a global variable . The operands name is best described in the reserved operands list in the "Diagnostics"-"Reserved Operands List".

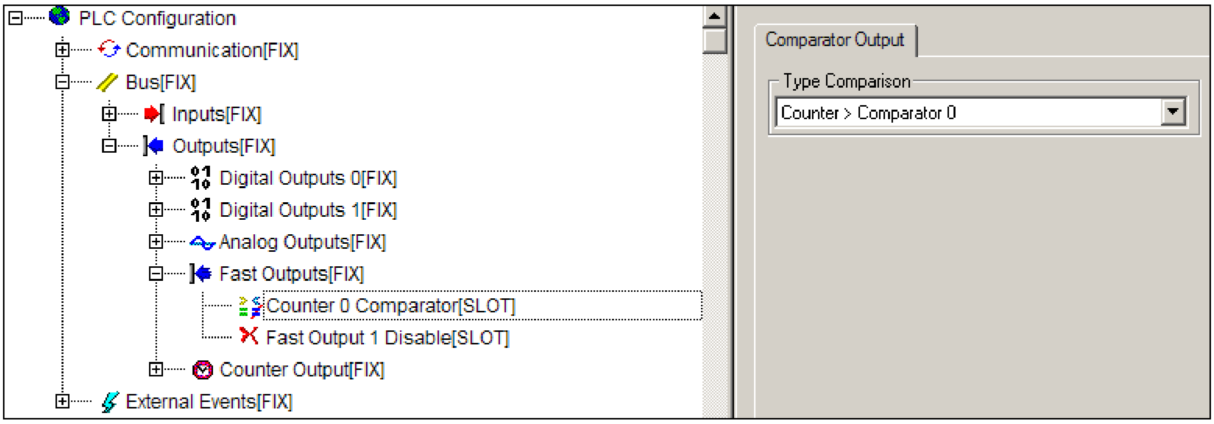


**Figure 3-22. Configuring Counter Comparator Output**

## Common Output of Counters Comparator

DU350 and DU351 controllers have 2 (two) common outputs which can be configured as counter comparison outputs.

The configuration of the common outputs as Counters Comparison Output is performed via the configuration tree. In order to access the configuration tree, open the MasterTool IEC and click "PLC Configuration", which is located in the "Resources" tab. Then the module "PLC Configuration" must be expanded. In this tab most of the required configurations for implementing the DU350/DU351 controllers can be configured.

The common outputs settings as Counters Comparison Outputs are located at the "Bus" module. By expanding this module, the "Inputs" and "Outputs" modules will appear. In order to set up the common outputs as Counters Comparison Output, the "Counter Common Output" module must be expanded.

The 2 common outputs are shown, then it is possible to configure the outputs as counter comparison outputs. The pins that receive the comparison outputs of both Block 0 and Block 1 counters are respectively the Q02 and Q03 outputs. If the output is not configured as a comparator output, it may be used as a common digital output.

Common comparison outputs are checked and updated every 1 ms.

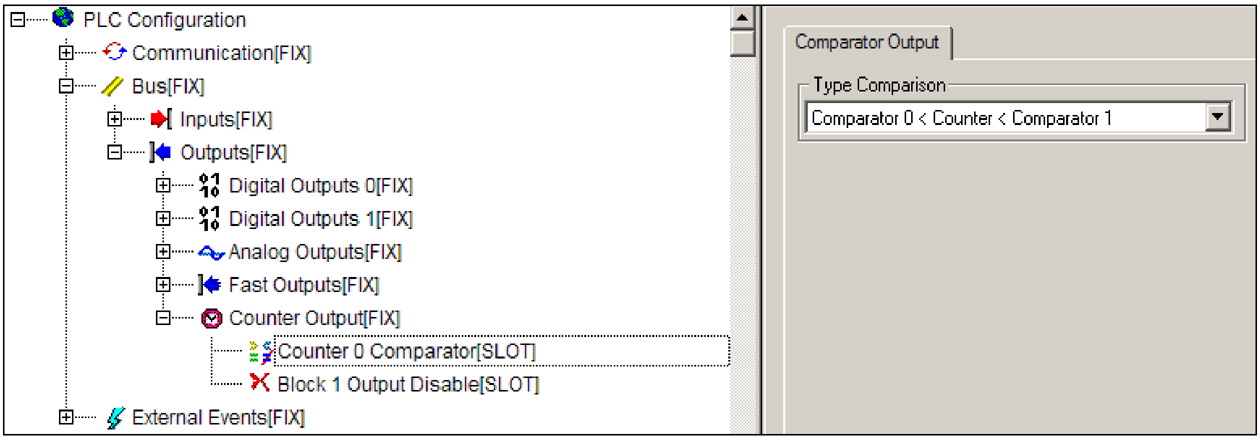


**Figure 3-23. Configuring the Counter Common Output**

## Positioning Functions

To facilitate positioning operations by using the Duo fast inputs, fast outputs, and analog outputs, a set of function blocks, which are responsible for the axes positioning is available. No configuration for the fast inputs and outputs or the analog outputs in the configuration tree is necessary.

The following functions are available in the DuoMotionLib.lib library. In addition to this library, the SysDuoHwConfig.lib and SysDuoMotionLib.lib, both of which have internal use functions of the DuoMotionLib library, should also be added to the project.

This library uses a common set of functions that perform boot, referencing of the axes, and a set of specific positioning functions that should be used for each type of control output.

There are four types of positioning:

1. Use of PTO fast outputs to position the motor connected to the axis, without the return of another type to validate the motion. It uses the PTO_MOTION function for positioning.
2. Use of PTO fast outputs to position the motor connected to the axis without the return of another type to validate the motion, but during the referencing it uses the input 0 of the encoder to reference. It uses the PTO _MOTION function for positioning.
3. Use of PTO fast outputs to position the motor connected to the axis, being the control closed through a fast counting input connected to an encoder which is also connected to the axis. It uses the PTO_MOTION function for positioning
4. Use of voltage analog outputs to position the motor connected to the axis, being the control closed through a fast counting input connected to an encoder which is also connected to the axis. It uses the ANALOG_MOTION function for positioning.

For each controlled axis an s_MOVE structure, which is described below, must be created. This structure is responsible for describing the physical patterns of the axis to be positioned. The CONTROL_TYPE_BYTE parameter indicates which of these types of motion will be performed. After initializing the structure for some determined input and output, the positioning function must be called with the same type used in this data structure.

ANALOG_INI and PTO_INI initialization functions do not perform movements, and must be called from within the program PLC_PRG or by a functional call within this block. On the other hand, the ANALOG_MOTION and PTO_MOTION positioning functions should be called within programs by time interruption.

| |
|---|
| ATTENTION:<br>This feature is available only as from the executive software version (1.11) and Mastertool IEC (1.04). |

## ANALOG_INI and PTO_INI Inicialization Functions

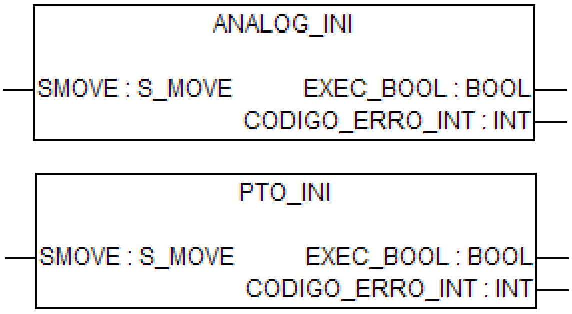


**Figure 3-24. ANALOG_INI and PTO_INI Function Blocks**

*Inputs*

SMOVE: it is an instance of the S_MOVE configuration structure used to define the machine parameters. Previously, the POS_INI function must be run, which checks parameters and consistence, and checks if there is some kind of error.

*Outputs*

EXEC_BOOL: initialization performed successfully. In case some error or inconsistence exists in the parameters of the movement structure, then the output will not be turned on and there will be an error code in the CODIGO_ERRO_INT output.

CODIGO_ERRO_INT: indicates the type of error that causes the termination of the function with an error. The cause may be consulted in Error Codes table (...Insert link).

*Operation*

To use the position control functionalities, first it is necessary to perform the initialization through the ANALOG_INI or PTO_INI functions depending on the type of the desired control. These functions perform the consistency of the S_MOVE structure parameters, and of the hardware configuration.

Before performing an initialization function it is necessary to fill in an S_MOVE structure type in accordance to the limits of the table below, and then perform the function which matches the type of control chosen.

The PTO_INI function performs the initialization of controls type 1(COUNT_PTO), 2 (PTO), and 4 (REFER_PTO), while the ANALOG_INI function initializes only controls type 3 (COUNT_ANALOG).

When an initialization function is executed successfully, that is, the S_MOVE structure carries all the correct values, the EXEC_BOOL output will be TRUE and the CODIGO_ERRO_INT output will show the value 0, indicating that there are no errors at startup.

Any change in the structure S_MOVE parameters should be performed when the system is stopped, and followed by calling their respective initialization function. The system is stopped when the ANALOG_MOTION and PTO_MOTION positioning functions are with the EXECUTANDO_BOOL output in FALSE.

The positioning functions check for correct initialization, so it is possible to run a positioning function only when the ANALOG_INI or the PTO_INI functions are executed successfully, otherwise an error is returned.

## Configuration Structure for COUNT_PTO, PTO, COUNT_ANALOG and REFER_PTO

| Structure variable | Unity | Allowed values | Variable description |
|---|---|---|---|
| Variables for PTO, COUNT_PTO and COUNT_ANALOG | | | |
| TIPO_CONTROLE_BYTE | | 1 – COUNT_PTO<br>2 – PTO<br>3 – COUNT_ANALOG<br>4 – REFER_PTO | Control type.<br>COUNT_PTO, control through a fast output with an encoder for feedback on the fast input.<br>PTO, control through a fast output without feedback.<br>COUNT_ANALOG, control through an analog output with an encoder for feedback on the fast input.<br>REFER_PTO, control through a fast output without feedback, but using its clear input for resetting during referencing. |
| SAIDA_BYTE | | 0 – Fast output F0 / Analog output AO0<br>1 – Fast output F1 / Analog output AO1 | Indicates that the PTO or analog output will be used for the function of positioning and referencing. The output type depends on the type of control chosen. |
| ENTRADA_BYTE | | 0 – Counter 0 (CNT0)<br>1 – Counter 2 (CNT2) | Indicates that the counter will be used used for the function of positioning and referencing.<br>For the PTO control type, this parameter is not relevant. |
| MODO_ENCODER_BYTE | | 0 – A increments and B decrements<br>1 – A increments or decrements and B gives the direction<br>2 – Square 2X<br>3 – Square 4X | Sets the configuration of the fast counter input. For more details on the counting modes, see chapter “Configuration” – “Input” – “Fast inputs”.<br>For the PTO control type, this parameter is not relevant |
| CONFIRMACOES_WORD | | 1 to 65.535 | Number of cycles of confirmations of position within the error range to validate a complete movement. |
| PERFIL_BYTE | | 0 – Trapezoidal Profile<br>1 – S Profile | Defines the acceleration/deceleration profile that will be used. |
| AMOSTRAGEM_TIME | time | t#1ms to t#1s | Positioning function calling time.<br>For the PTO or COUNT_PTO control type, this parameter is not relevant. |
| REFER_SENTIDO_BOOL | | FALSE – Positive search direction<br>TRUE – Negative search direction | Indicates whether the search direction of referencing zone is done with a logic level "0" or "1”. |
| REFER_VELOCIDADE_INICIAL_DWORD | um/s | 1 to 2.000.000.000 | Indicates the searching initial speed of the referencing zone. This parameter allows the speed is different from that configured in the field VELOCIDADE_REGIME_DWORD. |
| REFER_PERCENTUAL_VELOCIDADE_INICIAL_BYTE | % | 1 to 100 | It is the percentage compared with the initial speed that will be used to complete the search for the zero machine. In case the initial velocity is set to10 mm/s and this parameter is 20, the final speed of the search for zero is 2 mm/s, which corresponds to 20% of initial value |
| REFER_TEMPO_INVERSAO_ZONA_TIME | time | t#0ms to t#1m | Sets the time after which it will reverse the direction of search if it starts in the opposite direction to the signal search. As the deceleration time is also relevant in this direction the total time for reversal depends on the deceleration time plus this parameter. |
| VELOCIDADE_REGIME_DWORD | um/s | 1 to 2.000.000.000 | Defines the target speed (speed after acceleration) of the axis. The unit is length per second. |
| TENSAO_VELOCIDADE_REGIME_WORD | mV | 1 to 10.000 | Voltage that, applied to the servo control, sets the maximum speed.<br>For the PTO or COUNT_PTO control type, this parameter is not relevant |
| TENSAO_MAXIMA_WORD | mV | 1 to 10.000 | Maximum voltage to be supplied to the servo control.<br>It is recommended to use at least 10% higher than TENSAO_VELOCIDADE_REGIME_WORD to allow “overshoot”.<br>For the PTO or COUNT_PTO control type, this |

| | | | |
|---|---|---|---|
| | | | parameter is not relevant. |
| KP_REAL | | 0 to 50 | Proportional gain of positioning loop.<br>For the PTO or COUNT_PTO control type, this parameter is not relevant. |
| KI_REAL | | 0 to 50 | Integral gain of the positioning loop, activated only at the end to correct the error in the steady state.<br>For the PTO or COUNT_PTO control type, this parameter is not relevant. |
| TEMPO_ACELERACAO_TIME | time | t#0s to t#10m | Motor acceleration / deceleration time. |
| LIMITE_DE_PARADA_DINT | um | 1 to 10.000.000 | Tolerance range around the stop position.<br>In case of sliding greater than the allowable the control loop opens.<br>For the PTO or COUNT_PTO control type, this parameter is not relevant. |
| ERRO_MAXIMO_DINT | um | 0 to 10.000.000 | Maximum error allowed for the displacement of an axis to a position. |
| FIM_CURSO_SW_POSITIVO_DINT | um | 1 to 2.000.000.000 | Software limit switch in the positive direction of the axis. When the current value of the position is greater than the value specified in this parameter is perfomed a soft stop. This parameter should be set so that this value will allow stopping before the physical positive limit of the axis. |
| FIM_CURSO_SW_NEGATIVO_DINT | um | -1 to -2.000.000.000 | Software limit switch in the positive direction of the axis. When the current value of the position is less than the value specified in this parameter is made a soft stop. This parameter should be set so that this value will allow stopping before the physical negative limit of the axis. |
| TIMEOUT_TIME | time | t#1s to t#10h | Timeout for implementation of axis referencing. If the reference is not completed after the time set in this parameter a soft stop of the output is performed without the completion of the positioning. |
| COMPENSACAO_DE_FOLGA_DINT | um | -10.000.000 to 10.000.000 | Clearance for systems with indirect measurement.<br>The compensation signal is used to differentiate between the situation in which the measurement system counts, but the axis, due to the clearance, does not move (use the + sign) and the one in which the axis moves , but the measuring system, due to the clearance, starting late count (using sign -).<br>If the value of play is different from zero and the functions PTO_MOTION and ANALOG_MOTION try to use a relative motion, functions will return error |
| NUMERO_PULSOS_MOTOR_DINT | pulses | 1 to 1.000.000 | Sets the number of pulses required for a stepper motor or other motor executes a turn in the axis.<br>For the COUNT_ANALOG control type, this parameter is not relevant. |
| DESLOCAMENTO_MOTOR_DINT | um | 1 to 10.000.000 | Sets the displacement in unit of length which corresponds to a turn of the motor.<br>For the COUNT_ANALOG control type, this parameter is not relevant. |
| NUMERO_PULSOS_ENCODER_DINT | pulses | 1 to 1.000.000 | Sets the number of pulses generated by an encoder when it performs a turn in the axis. Should be filled with the nominal number of pulses of the encoder. |
| DESLOCAMENTO_ENCODER_DINT | um | 1 to 10.000.000 | Sets the displacement in unit of length which corresponds to a turn of the encoder. |

**Table 3-9. Variables of an S_MOVE Structure**

ATTENTION:
The maximum error should be greater than or equal to the ratio (DESLOCAMENTO_ENCODER_DINT / NUMERO_PULSOS_ENCODER_DINT), ), the same as the resolution µm / pulse, and this resolution must have a value of at least 1.

### Referencing Operation

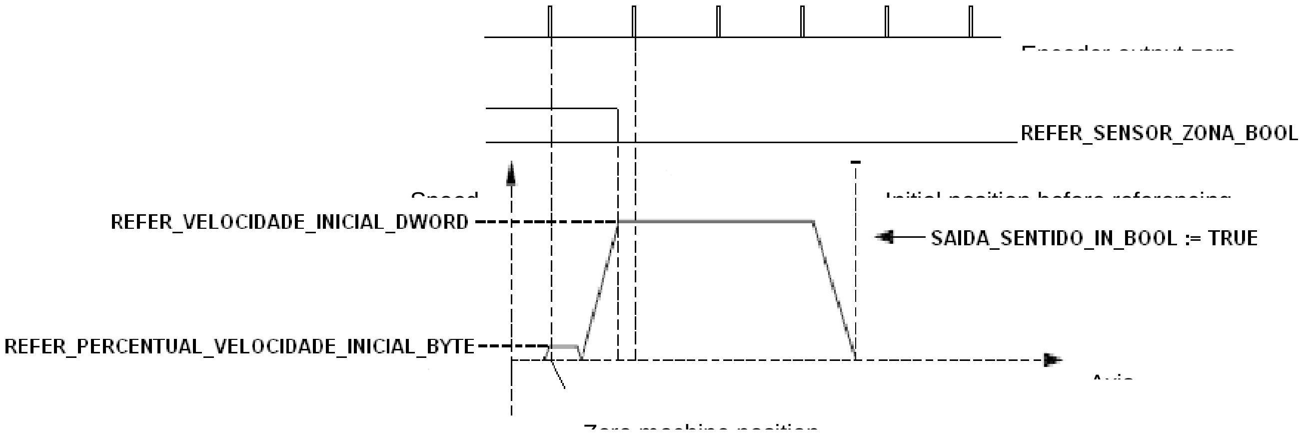


**Figure 3-25. Referencing Diagram**

The figure 3-25 shows the diagram for the referencing of an axis. In it is represented the axis acceleration profile for the movement and the entries of the referencing zone sensors, REFER_SENSOR_ZONA_BOOL, as well as the output of the encoder that indicates passage by zero. The referencing zone is the axis range where the zero encoder pulse is analysed by the referencing input. In this way a referencing begins shifting the axis towards declared parameter REFER_SENTIDO_BOOL of the S_MOVE structure. The axis is accelerated until it reaches the speed of search for the REFER_VELOCIDADE_INICIAL_DWORD referencing zone. When finding a rising edge in REFER_SENSOR_ZONA_BOOL a soft axis stop is performed decelerating until it stops.

If during this movement a zero has been detected in the referencing zone, this zero shall be considered the position zero of the machine. Otherwise the machine keeps moving in the same direction but with a percentage of initial speed declared in REFER_PERCENTUAL_VELOCIDADE_INICIAL_BYTE. When finding a zero coming from the output of the encoder this point is the zero of the machine and a soft stop is performed.

When COMPENSACAO_DE_FOLGA_DINT (configured in the S_MOVE structure) is greater than zero, after reaching the zero of the machine, a shift of the size of the backslash of the system will be generated in order to ensure that the slack will be compensated at the end of the reference.

At the end of this process if the axis is stopped within the ERRO_MAXIMO_DWORD tolerance of error, no movement is performed. In case is out, new movements must be executed to stay within the maximum error range around zero.

Before the referencing triggering, the REFER_SENSOR_ZONA_BOOL input is tested and in case it is “TRUE” the referencing will initiate the search in the opposite direction to REFER_SENTIDO_BOOL, in order to leave the referencing zone. After detecting a falling edge of the referencing sensor, the axis remains with the initial speed of seacrch, REFER_VELOCIDADE_INICIAL_DWORD, for a determined time interval, which was defined in REFER_TEMPO_INVERSAO_ZONA_TIME. Only after that, it starts to slow down. By ceasing this movement, the search continues normally, in the search direction defined in the REFER_SENTIDO_BOOL parameter of the function.

It is recommended that the zone sensor is mounted in a way that, after a rising edge, it remains in status “1” up to the end of the axis. Otherwise the axis may be positioned outside the zone, but opposite to the search direction. In this case the axis will be stopped abruptly by the end of the course located on this side of the axis.

In the case of a control using only the PTO output, the counter goes to zero to indicate the zero of the machine on the REFER_SENSOR_ZONA_BOOL rising edge. This is because the encoder not being used, there is no indication of passage through the zero of the encoder.

*Mechanical  Assembly of an Axis*

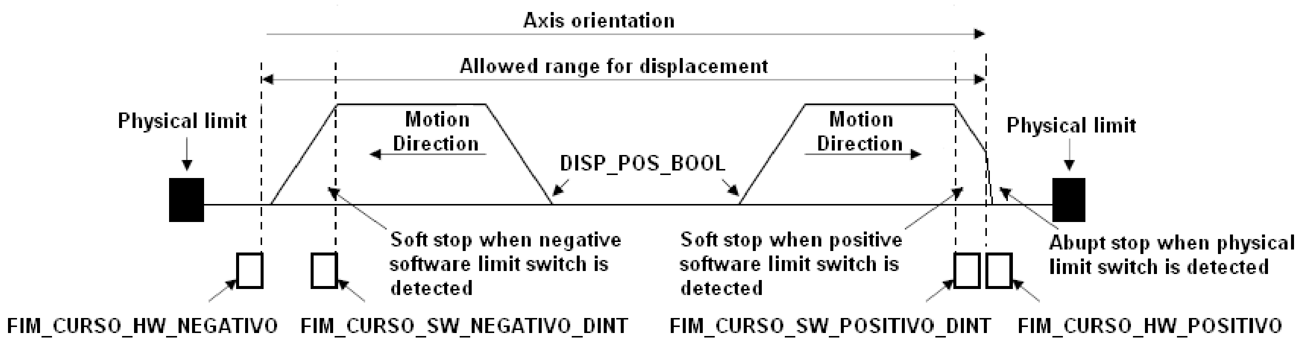


**Figure 3-26. Mounting of the Axis**

### Referencing Functions and Positioning

The DU350 and DU351 motion libraries have two positioning and referencing functions. One uses a fast output configured to generate pulse train (PTO) as an actuator, and  the other function uses an analog output.

*PTO_MOTION Function*

PTO_MOTION function uses a fast output set as PTO to act on the system. This function may be configured to use a two-way counter associated with an encoder or an internal pulse counter as feedback control loop. The differentiation of the  behavior desired is made through the TIPO_CONTROLE_BYTE parameter of the SMOV structure. In this case when this parameter is equal to 1 (COUNT_PTO) the loop is closed by the use of a fast counter. If it is set to 2 (PTO) the control is performed by using the absolute counter of the PTO output.

In addition, there is another way of configuration that  combine the characteristics of the other two modes that involve the PTO output. For this case the TIPO_CONTROLE_BYTE parameter should be 4 (REFER_PTO). In this configuration the behavior is identical to that of the configuration as PTO for positioning, but if the axis has a position statement of zero through an encoder, for example, this mode should be used, and the position counter will go to zero on the zeroing position pulse when the axis is referenced.

To select the type of control that should be used, the type of drive used should also be taken into consideration. If  the triggering servo holds the position control, which means that the control loop is closed within the driver itself, the control to be used is the direct PTO. If this system has information passing through zero, the REFER_PTO type should be used to increase referencing accuracy. However, if the system uses a driver with combinational logic, such as the drivers for triggering step motors, which are coupled to an encoder axis, the COUNT_PTO control type must be used.

This function should be periodically called through a POU triggered by time interruption, which must be configured as shown in “Configuration” – “POUS triggered by time interruption”. In order to obtain accurate control, it is recommended to use the shortest possible time to call this function.

After setting this control module through the PTO_INI function, and periodically calling the PTO_MOTION function, it will monitor the function inputs periodically. In case either a referencing or a positioning is requested through the inputs, the function runs the state machine responsible for each function. During execution both a new positioning and referencing will not be allowed, unless the motion is completed or in case of an enforced break through emergency stop (EMERGENCIA_BOOL) or soft stop (SOFTSTOP_BOOL). While running, the EXECUTANDO_BOOL output stays on. If there is any error, the ERRO_BOOL output is triggered, and the respective error is placed into the CODIGO_ERRO_INT variable.

Usually this type of function does not need CONFIRMACOES_WORD greater than 1. But in systems with large inertia it is necessary to increase the number of confirmations in order to ensure the placement within ERRO_MAXIMO_DINT. If the value of the error is very small the axis may be not correctly positioned. The number of confirmations is used only for the COUNT_PTO type.

In case the positioning is not closed properly, there is the TIMEOUT_TIME parameter that stops the movement after a set time even if the movement has not been completed successfully.

### *ANALOG_MOTION Function*

ANALOG_MOTION function uses an analog output to act on the system, and the control is fed through an encoder. This analog output, ranging from 0 to 10 Volts, has an associated digital output to define the direction of motion, where FALSE sets the positive direction, and TRUE the negative direction

The analog control is proportional-integral, and the gains are set by the parameters KP_REAL(proportional gain), and KI_REAL (integral gain), in the S_MOVE structure. However, integral control is activated only at the end of the theoretical profile route, with the purpose of correcting the characteristic error of a system that uses only one proportional control. If there is no need for an integral control, the KI_REAL parameter must be set to zero.

This function must be periodically called by a POU triggered by time interruption, and which must be configured as described in the chapter “Configuration – POUs triggered by time interruption”. For a more accurate control, it is recommended to use the shortest possible time for calling the function (1 ms).

After setting this control module with the ANALOG_INI function, and periodically calling the ANALOG_MOTION function, the control loop starts in open, signaled as FALSE in CLOSED_LOOP_BOOL.To close the control loop it is necessary to force its closing through the CLOSE_LOOP_BOOL command, and also trigger a new positioning via DISP_POS_BOOL command or trigger a new referencing via DISP_REFER_BOOL command.

When the control loop is closed, the stop limit is being checked. This check is to ensure that in case of loss of control or undue movement, generated by factors outside the system, the loop will open, which prevents accidents. The LIMITE_DE_PARADA_DINT parameter of the S_MOVE structure is responsible for setting the value of the stop limit.

The open control loop can occur in the following ways: by the emergency control command (EMERGENCIA_BOOL), by passing the stop limit (LIMITE_DE_PARADA_DINT) ,by the triggering of one limit switch by hardware (FIM_CURSO_HW_POSITIVO_BOOL or FIM_CURSO_HW_NEGATIVO_BOOL), by restarting the control module with erroneous values in the structure S_MOVE, by the bursting of the current position or by an attempt to trigger a positioning with a value (POSICAO_DINT) which is out of range.

By triggering a position or a referencing, the EXECUTANDO_BOOL output will be set, and it will be cleaned only at the end of the positioning, which may happen due to a successfully completed positioning or due to some exception, such as bumping into one limit switch.

The monitoring error between the theoretical profile generated by the positioning module and the current average position through the encoder is shown at the ERRO_ACOMPANHAMENTO_DWORD output ,and its signal is reported at the SINAL_ERRO_ACOMPANHAMENTO_BOOL output. With this data it is possible to see the behavior of the system in relation to gains defined in S_MOVE structure. If the monitoring error is increasing constantly, this is a sign that the proportional gain does not carry a value which is suitable enough to cause the real position to track the theoretical position with a constant error. So, the proportional gain should be increased.

Another important information for the control calibration is the SATURADO_BOOL output. This output is set when the control theoretical signal exceeds the maximum voltage of the analog output which is set in TENSAO_MAXIMA_WORD parameter in the S_MOVE structure. The saturation of the signal leads the monitoring error to increase steadily as described above.

The completion of a positioning occurs when the axis position is detected, and the number of consecutive confirmations is set at the CONFIRMACOES_WORD parameter of the S_MOVE structure. If the system can not find a position and an emergency stop is not triggered, the control will open after the timeout and an error code will be generated.

The reasons why the system can not find the final position are: ERRO_MAXIMO_DINT too small for the system dynamics, time of ANALOG_MOTION function call very large, or very small integral gain for a system that requires a relatively high torque to work with proportional control only.

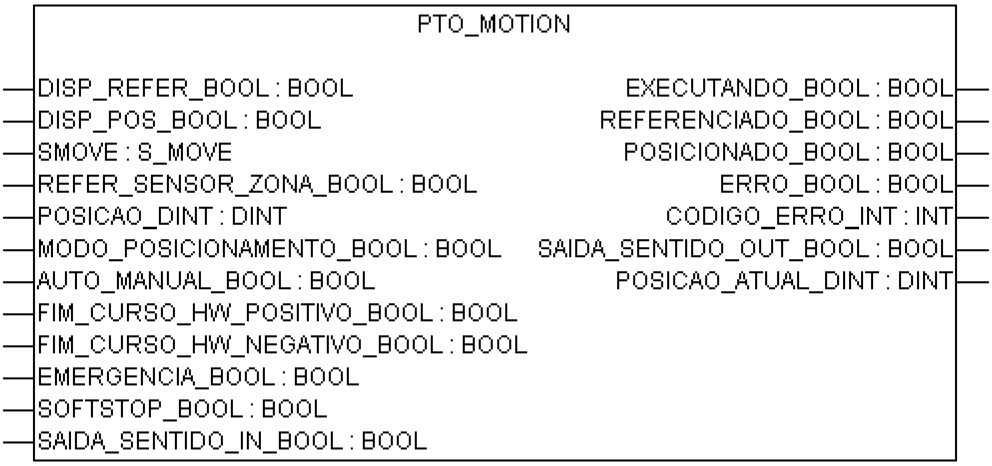


**Figure 3-27. PTO_MOTION Function Block**

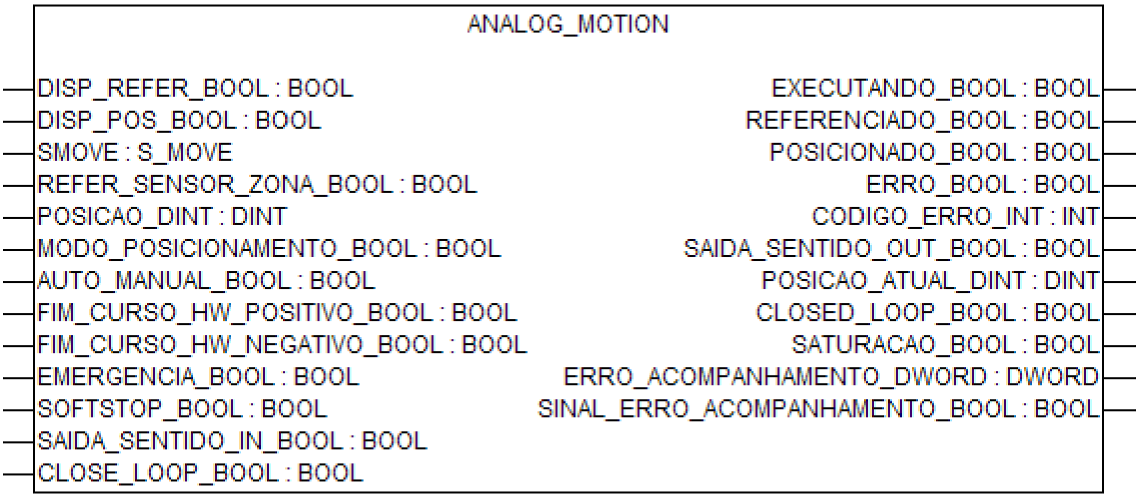


**Figure 3-28. ANALOG_MOTION Function Block**

*Backslash Compensation*

The positioning functions have a backslash compensation for systems that use indirect measurement, where the motor is connected to the encoder through a system.

To use the backslash compensation it is necessary to configure the COMPENSACAO_DE_FOLGA_DINT parameter in the S_MOVE structure. When there is no need for backslash compensation, this parameter must be set to zero.

The compensation signal is used to differentiate between the situation in which the measurement system counts, but the axis does not move due to the backslash (use a positive sign), and one in which the axis shifts, and the measurement system starts the counting late (use minus sign) because of the backslash.

The backslash compensation work only when the motion is absolute. In case of a relative motion and non-zero compensation, the function returns an error code.

*Inputs*

DISP_REFER_BOOL: Input used to trigger a new referencing. The input is sensitive to level. After the function is energized it remains running the EXECUTANDO_BOOL output enabled until the referencing is completed. After completing, the REFERENCIADO_BOOL output is turned on and a new referencing or positioning can be performed.

DISP_POS_BOOL: Input used to trigger a new placement. The input is sensitive to level. When the function is performed with this input turned on, if a positioning is not in progress then a new positioning will be executed. After the function is energized it remains running with the EXECUTANDO_BOOL output activated until the positioning is completed. After completing, the POSICIONADO_BOOL output is turned on and a new positioning or referencing may be performed.

SMOVE: Is an instance of the S_MOVE configuration structure used to define the machine parameters. Previously the POS_INI function must be executed, which verifies the parameters and consistence and checks if there is some type of error. Any change in this structure should be performed when the system is stationary, that is, when EXECUTANDO_BOOL is FALSE.

REFER_SENSOR_ZONA_BOOL: Input for the referencing zone sensor. The referencing zone is always indicated by logical level “1” in this input.

POSICAO_DINT: Indicates the new position for the next movement. In the case of absolute movement this value must be the new position. In case of relative movement , this value represents the displacement relative to the current position. For the analog control, in referencing, the value of this parameter is used as the absolute position for a new position performed automatically at the end of the referencing.

MODO_POSICIONAMENTO_BOOL: Indicates whether the movement to be performed for the next trigger is absolute or relative. When this input is FALSE, the movement is absolute and the positioning is performed in a way that the axis is positioned at the value indicated by the POSICAO_DINT input. When this input is TRUE, the movement is relative and the positioning will be performed in a way that the axis is positioned at the current value plus the value of the POSICAO_DINT input.

AUTO_MANUAL_BOOL: This entry is used to bypass the state of physical limit switch inputs FIM_CURSO_HW_POSITIVO_BOOL and FIM_CURSO_HW_NEGATIVO_BOOL. When this input is FALSE the limit switch inputs are treated normally. If this input is TRUE then the limit switch inputs will be ignored and movements may be performed even if the axis is on its physical limits. This inputs exists only to enable the axis to be removed from the physical limit in case of stop due to the limit switch sensor. In this case the input must be triggered and must beware of the direction in which the motion shall be carried out in order to avoid damage to the equipment. This in input is interpreted only if the DISP_POS_BOOL input is triggered, so it is not interpreted in the case of an attempt at referencing. After a completed referencing, if the input is TRUE, the error output does not remain on even if the axis same is over one limit switch by hardware.

FIM_CURSO_HW_POSITIVO: This input is used to indicate the physical limit in the positive side of the axis. When this input is triggered, it indicates to the functions that the motion shall not continue in this direction, or it may cause damages in the structure of this axis. Thus, when this active sensor is detected, an abrupt stop is performed in the axis so that it will stop immediately without deceleration. If it is stopped, it is only possible to move the axis through a positioning with the AUTO_MANUAL_BOOL input on or through a forced shift.

FIM_CURSO_HW_NEGATIVO: This input is used to indicate the physical limit in the negative side of the axis. When this input is triggered, it indicates to the functions that the motion shall not continue in this direction, or it may cause damages in the structure of this axis. Thus, when this active sensor is detected, an abrupt stop is performed in the axis so that it will stop immediately without deceleration. If it is stopped, it is only possible to move the axis through a positioning with the AUTO_MANUAL_BOOL input on or through a forced shift.

EMERGENCIA_BOOL: This input is used for emergency. The emergency stop generates an abruptly stop in the axis and cannot be bypassed in the function, in a way that it can be activated in case the security mechanisms like the limit switchess come to fail.

SOFTSTOP_BOOL: If a positioning is in progress and the system is accelerating or in target speed, this input starts a soft stop.

SAIDA_SENTIDO_IN_BOOL: Digital output used to indicate the direction of movement performed by the motor. It must be declared the same parameter that in SAIDA_SENTIDO_OUT_BOOL.

CLOSE_LOOP_BOOL: Closes the control loop in case it is open. This input is used when an abrupt stop (emergency or limit switch by hardware) has been performed, and therefore the control loop has been opened. All the triggering of both DISP_REFER_BOOL and DISP_POS_BOOL close the loop automatically. This input is used only in the analog control.

*Outputs*

EXECUTANDO_BOOL: Execution of the function successfully performed.

REFERENCIADO_BOOL: Finalization of the axis referencing in the last cycle of the application program.

POSICIONADO_BOOL: Finalization of the axis referencing in the last cycle of the application program

ERRO_BOOL: An error occurred during the execution of the function. The error can be seen in the output COD_ERRO of the function. This bit remains only one cycle with the error value and requires constant verification.

CODIGO_ERRO_INT: Indicates the type of error that causes the termination of the execution of the function with the error. This bit remains only one cycle with the error value and requires constant verification. The cause may be consulted in Error Codes table (Table 3-10).

SAIDA_SENTIDO_OUT_BOOL: Digital output used to indicate the direction of movement performed by the motor. When the movement happens in the positive direction of the axis the output will be set to the logic state “0”. When the movement happens in the negative direction of the axis the output will be set to the logic state “1”.

POSICAO_ATUAL_DINT: the current position in the unit of length set by the user, according;

CLOSED_LOOP_BOOL: CLOSED_LOOP_BOOL:Indicates whether the control loop is closed. Normally this output is used to define the moment when the brake should be activated. When the loop is opened it is recommended to disconnect the servo through its enabling input. This output is used only for analog control.

SATURACAO_BOOL: Indicates that the control theoretical signal has exceeded the maximum voltage declared in the S_MOVE structure. This output is used only for analog control.

ERRO_ACOMPANHAMENTO_DWORD: Displays the value of the difference between the theoretical position and the real position of the system. This data, as well as the SATURACAO_BOOL variable, are important to set the value of the KP_REAL proportional gain, which should be used in control. When in target speed, this value should be around a “constant” value. If this value is constantly increasing, the output signal is saturating and the maximum rotation of the motor is not enough to make the system follow the theoretical profile . This output is used only for analog control.

SINAL_ERRO_ACOMPANHAMENTO_BOOL: Indicates what the signal of the variable ERRO_ACOMPANHAMENTO_DWORD is.

## Error Codes

| Function Error | Function Error Code | Function Error Description |
|---|---|---|
| General error codes | | |
| COD_ERRO_EMERGENCIA | 1 | When abrupt stop occurs due to emergency input triggering. |

| | | |
|---|---|---|
| COD_ERRO_FIM_CURSO_HW_POSITIVO | 2 | When abrupt stop occurs due to logical state "1" in the positive limit switch input. |
| COD_ERRO_FIM_CURSO_HW_NEGATIVO | 3 | When abrupt stop occurs due to logical State "1" in the negative limit switch input. |
| COD_ERRO_FIM_CURSO_SW_POSITIVO | 4 | When soft stop occurs due to the current position greater than the configured value at the positive software limit. |
| COD_ERRO_FIM_CURSO_SW_NEGATIVO | 5 | When soft stop occurs due to the current position less than the configured value at the negative software limit. |
| COD_ERRO_NAO_INICIALIZADO | 6 | Attempt to run a positioning or referencing without first having executed an initialization using the ANALOG_INI function or PTO_INI function. |
| COD_ERRO_INTERTRAVAMENTO | 7 | Returns this error when trying to run a function when another is still running using the same resource (for example the same PTO output). |
| COD_ERRO_TIMEOUT | 8 | When a motion exceeded timeout established by the user |
| COD_ERRO_SOFTSTOP | 9 | When soft stop occurs due to the soft stop input triggering. |
| COD_ERRO_POSICAO_ATUAL | 10 | Failure in the calculation of the current position due to the variables overflow. |
| COD_ERRO_POSICAO | 11 | Position parameter outside of the range. |
| Referencing error codes | | |
| COD_ERRO_BUSCA | 80 | Internal error of the function in the search by zero states machine. |
| COD_ERRO_BUSCA_INV | 81 | Internal error of the function in the search by zero states machine. |
| PTO_MOTION function error codes | | |
| COD_ERRO_ESTADO_PTO | 110 | Internal error of the function in the positioning states machine. |
| COD_ERRO_ESTADO_COUNT_PTO | 140 | Internal error of the function in the positioning states machine. |
| ANALOG_MOTION function code errors | | |
| COD_ERRO_LIMITE_DE_PARADA | 170 | The axis displacement supposed to be stopped, but exceeded the stop limit. |
| COD_ERRO_ESTADO_COUNT_ANALOG_POS | 171 | Internal error of the function in the positioning states machine. |
| COD_ERRO_ESTADO_COUNT_ANALOG_REFER_1 | 172 | Internal error of the function in the referencing states machine. |
| COD_ERRO_ESTADO_COUNT_ANALOG_REFER_2 | 173 | Internal error of the function in the referencing states machine. |
| Errors in the SMOVE structure parameters | | |
| COD_ERRO_PARAM_ERRO_MAXIMO | 200 | Maximum error value is smaller than the resolution of the motor or encoder. This code is also used to indicate that the maximum error variable has a value less than zero. |
| COD_ERRO_PARAM_PERFIL | 201 | Invalid profile type. |
| COD_ERRO_PARAM_MODO_ENCODER | 202 | Invalid encoder mode value. |
| COD_ERRO_ PARAM_SAIDA | 203 | Invalid output value. |
| COD_ERRO_ PARAM_ENTRADA | 204 | Invalid counter value. |
| COD_ERRO_ PARAM_TIPO_CONTROLE | 205 | Invalid value of the positioning control type. |
| COD_ERRO_ PARAM_VELOCIDADE_REGIME | 206 | Target speed value outside the speed range or in the case of control using the PTO output, the frequency is out of range. In this case the frequency is calculated as:<br>(VELOCIDADE*NUMERO_PULSOS_MOTOR_DINT/ DESLOCAMENTO_MOTOR_DINT) |
| COD_ERRO_ PARAM_NUMERO_PULSOS_MOTOR | 207 | Value of pulse number per turn of the motor is outside the range. |
| COD_ERRO_ PARAM_DESLOCAMENTO_MOTOR | 208 | Value of displacement per turn of the stepper motor is outside the range. |
| COD_ERRO_ PARAM_NUMERO_PULSOS_ENCODER | 209 | Value of pulse number per turn of the encoder is outside the range. |

| | | |
|---|---|---|
| COD_ERRO_ PARAM_DESLOCAMENTO_ENCODER | 210 | Value of displacement per turn of the encoder is outside the range. |
| COD_ERRO_TEMPO_ACELERACAO | 211 | Acceleration time value outside the allowed range. |
| COD_ERRO_PARAM_COMPENSACAO_DE_ FOLGA | 212 | Value of backslash compensator non-zero and out of range or non-zero and less than the maximum error (ERRO_MAXIMO_DWORD). This code is also generated when a relative motion with backslash compensation non-zero is triggered. |
| COD_ERRO_PARAM_FIM_CURSO_SW_POS ITIVO | 213 | Value of positive software limit is outside the allowed range. |
| COD_ERRO_PARAM_FIM_CURSO_SW_NEG ATIVO | 214 | Value of negative software limit is outside the allowed range. |
| COD_ERRO_PARAM_TIMEOUT | 215 | Time value for timeout is outside the allowed range. |
| COD_ERRO_PARAM_REFER_PERCENTUAL _VELOCIDADE_INICIAL | 216 | Parameter of speed percentage in the zone outside the range of 1 to 100%. |
| COD_ERRO_PARAM_TEMPO_INVERSAO_Z ONA | 217 | The inversion time to leave the referencing zone is out of range. |
| COD_ERRO_PARAM_REFER_VELOCIDADE_ INICIAL | 218 | Initial search speed greater than target speed set up in the structure, outside the speed range or in the case of control using the PTO output, the frequency is out of range. In this case the frequency is calculated as:<br>(VELOCIDADE*NUMERO_PULSOS_MOTOR_DINT/ DESLOCAMENTO_MOTOR_DINT) |
| COD_ERRO_PARAM_CONFIRMACOES | 219 | Positioning confirmations value outside the allowed range. |
| Errors in the parameters of the structure SMOVE specific of the analog control | | |
| COD_ERRO_PARAM_AMOSTRAGEM | 220 | Sampling time outside the allowed range. |
| COD_ERRO_PARAM_TENSAO_VELOCIDAD E_REGIME | 221 | Target speed voltage outside the allowed range or greater than the maximum voltage. |
| COD_ERRO_PARAM_TENSAO_MAXIMA | 222 | Maximum voltage outside the allowed range. |
| COD_ERRO_PARAM_KP | 223 | Proportional gain outside the allowed range. |
| COD_ERRO_PARAM_KI | 224 | Integral gain outside the allowed range. |
| COD_ERRO_PARAM_LIMITE_DE_PARADA | 225 | Value of stop limit outside the allowed range or smaller than the maximum error. |

**Table 3-10. Error Codes of the Motion Functions**

## Communication

In order to configure the ports, open the MasterTool IEC and click “PLC Configuration”, located in the “Resources” tab. Then, the module “PLC Configuration” must be expanded.

COM port settings are located in the communication module. By expanding this module, the “COM1”and “COM2”modules will be shown. In order to configure COM1, for example, click on “COM1” module. A tab with the port settings will pop up on the right side. There, the following may be set:

- **Parity**
  - No parity
  - Odd
  - Even
  - Always 1
  - Always 0

- **Stop Bits**
  - 1 Stop Bit

- 2 Stop Bits

- **Modem Signals**
  - No RTS/CTS
  - With RTS/CTS
  - With RTS without CTS
  - RTS always on

- **Delay**
  - 5 to 1000 ms

- **Baud Rate (bps – bits per second)**
  - 1200
  - 2400
  - 4800
  - 9600
  - 19200
  - 38400
  - 57600
  - 115200

| ATTENTION:<br>The time defined in the delay field indicates the minimum time between the reception of a MODBUS package and the sending of another (delay between frames). This definition refers to the minimum amount of time, which may vary according to the POUs runtime employed. E.g., if there is a 20 ms runtime POU, the delay between frames may be of approximately 20 ms, even though it is configured as 5 ms. |
|---|

| ATTENTION:<br>Always is added a time of 30 ms at the time defined on the field delay, to the COM2, because the manner as the serial interface is build. In this case when a time is configured to 15 ms on this field, the real delay will be 45 ms. |
|---|

The MODBUS protocol configuration in both ports is shown in the following:

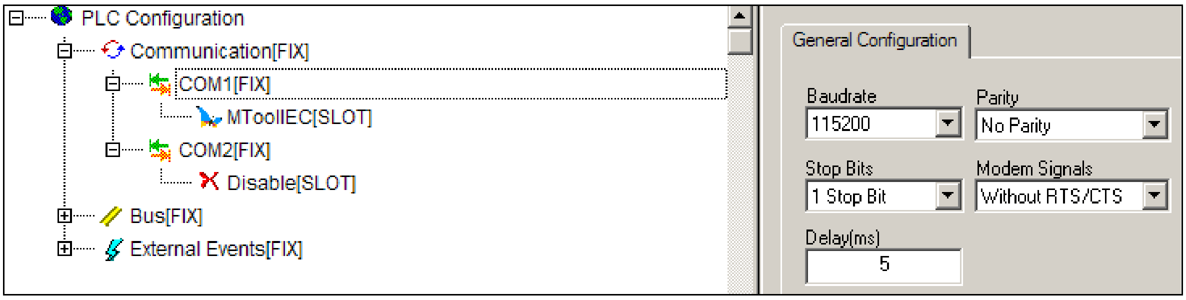


**Figure 3-29. Configuring COM1**

ATTENTION:
For the relation between CPU cycle time and communication with MasterTool IEC, see the warning message, in the chapter “Technical Description” - “Performance” - “Cycle Time”.

ATTENTION:
Ixx reserved operands(digital inputs), Qxx (digital outputs), AIx (analog inputs) and AOx (analog outputs) are mapped into I and Q operands, which are accessible via MODBUS. To check the memory address used for each operand, use the of Operands Reserved List table in this manual.

## MODBUS Master

The two COM ports may be configured as MODBUS Master or Slave. To configure COM1 as MODBUS Master, open the MasterTool IEC, then click “PLC Configuration” located at “Resources” tab. Then, the module “PLC Configuration” must be expanded.

COM port settings are located in the communication module. By expanding this module, “COM1” and “COM2” modules will appear. In order to configure COM1, the “COM1” module must be expanded. Similarly, the “COM2 module” must be expanded to configure the communication interface of COM2.

By expanding this block, COM1 port will be disabled for MODBUS. To enable it, it is necessary to click with the right mouse button on the “MToolIEC” module, then select the option “Replace element” and then choose “MODBUS Master” to enable COM1 as MODBUS Master. For the COM2 Port configuration, the same procedure must be performed, except that for this interface the “MToolIEC” option is not available.

By clicking on the enabled module “MODBUS Master” two dialog boxes containing the following options will appear on the right side:

- Timeout (ms) – defines the time interval the master will wait for a slave to reply. If the response time is greater than the value configured in the time-out field, the controller indicates a communication error in the respective operand (if there is no more Fail Try). In case of time-out the controller retransmits the packet to a particular slave the number of times defined in the Fail Try before performing a new defined MODBUS relationship. The time-out may be configured with values from 1ms to 10 s  (10000 ms).
- Fail Try – defines the number of times the Master will retransmit the package in case there is no reply from the Slave  (after the configured time). The fail try may be configured from 1 to 10.

After these two configurations, it is necessary to enable the desired MODBUS relations.

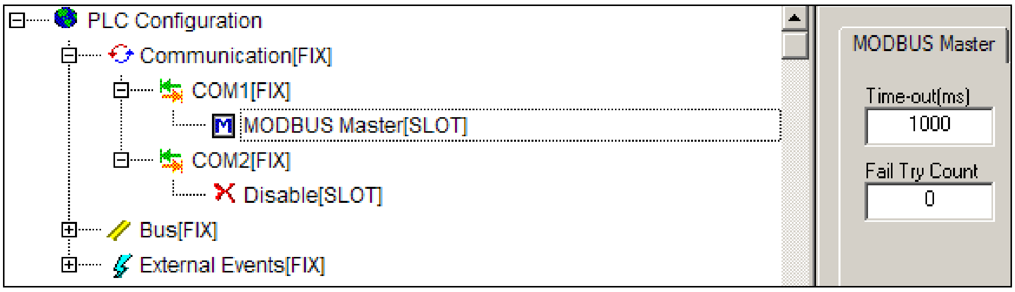


**Figure 3-30. MODBUS Master**

*MODBUS Relation*

A MODBUS relation is a MODBUS protocol message addressed to a particular slave module. It It is allowed the use of 32 MODBUS relations, distributed to both COM ports (the limit is 16 relations for each COM port). The relations treatment is done sequentially as soon as they are added to the configuration tree.

MODBUS relations may be added to the Master by clicking the right mouse button on “MODBUS Master module” and by selecting the option “Append MODBUS Relation”. Thus, the subnode keys “MODBUS Relation” will be added to the MODBUS Master.

If the operand COMx_DR is TRUE, the relations of the x port communication are disabled. Where “x” is the communication port number, which may hold the values 1 or 2.

The following configuration parameters must be adjusted for each relation:

| **MODBUS Function** | | **Read Coils** | **Reads a variable number of digital outputs** |
|---|---|---|---|
| **Devices Address** | | 1 – 247 | Slave address |
| **Quantity** | | 1 – 2000 | Quantity to be read |
| **MODBUS Address** | | 1 – 65535 | Initial reading MODBUS address on slave |
| **Polling** | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| **MasterTool IEC Operand** | **Type: %M** | %MX0 - %MX3186 | Writing address range of command response value |
| | **Type: %Q** | %QX0 - %QX63 | Writing address range of command response value |
| | **Type: %I** | Writing is not allowed in the inputs | Writing address range of command response value |

**Table 3-11. Read Coils Function**

| **MODBUS Function** | | **Read Discrete Inputs** | **Reads a variable number of digital inputs** |
|---|---|---|---|
| **Devices Address** | | 1 – 247 | Slave address |
| **Quantity** | | 1 – 2000 | Quantity to be read |
| **MODBUS Address** | | 1 – 65535 | Initial reading MODBUS address on slave |
| **Polling** | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| **MasterTool IEC Operand** | **Type: %M** | %MX0 - %MX3186 | Writing address range of command response value |
| | **Type: %Q** | %QX0 - %QX63 | Writing address range of command response value |
| | **Type: %I** | Writing is not allowed in the inputs | Writing address range of command response value |

**Table 3-12. Read Discrete Inputs Function**

| **MODBUS Function** | | **Read Holding Registers** | **Reads a variable number of registers** |
|---|---|---|---|
| **Devices Address** | | 1 – 247 | Slave address |
| **Quantity** | | 1 – 123 | Quantity to be read |
| **MODBUS Address** | | 1 – 65535 | Initial reading MODBUS address on slave |
| **Polling** | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| **MasterTool IEC Operand** | **Type: %M** | %MW0 - %MW3186 | Writing address range of command response value |
| | **Type: %Q** | %QW0 - %QW63 | Writing address range of command response value |
| | **Type: %I** | Writing is not allowed in the inputs | Writing address range of command response value |

**Table 3-13. Read Holding Registers Function**

| MODBUS Function | | Read Input Registers | Reads a variable number of input registers |
|---|---|---|---|
| Devices Address | | 1 – 247 | Slave address |
| Quantity | | 1 – 123 | Quantity to be read |
| MODBUS Address | | 1 – 65535 | Initial reading MODBUS address on slave |
| Polling | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| MasterTool IEC Operand | Type: %M | %MW0 - %MW3186 | Writing address range of command response value |
| | Type: %Q | %QW0 - %QW63 | Writing address range of command response value |
| | Type: %I | Writing is not allowed in the inputs | Writing address range of command response value |

**Table 3-14. Read Input Registers Function**

| MODBUS Function | | Write Single Coil | Forces one single coil |
|---|---|---|---|
| Devices Address | | 1 – 247 | Slave address |
| Quantity | | 1 – 1 | Quantity to be written |
| MODBUS Address | | 1– 65535 | Initial writing MODBUS address on slave |
| Polling | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| MasterTool IEC Operand | Type: %M | %MW0 - %MW3186 | Address range to read the value to be sent |
| | Type: %Q | %QW0 - %QW63 | Address range to read the value to be sent |
| | Type: %I | %IW0 - %IW63 | Address range to read the value to be sent |

**Table 3-15. Write Single Coil**

| MODBUS Function | | Write Single Register | Preset o one single register |
|---|---|---|---|
| Devices Address | | 1 – 247 | Slave address |
| Quantity | | 1 – 1 | Quantity to be written |
| MODBUS Address | | 1 – 65535 | Initial writing MODBUS address on slave |
| Polling | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| MasterTool IEC Operand | Type: %M | %MW0 - %MW3186 | Address range to read the value to be sent |
| | Type: %Q | %QW0 - %QW63 | Address range to read the value to be sent |
| | Type: %I | %IW0 - %IW63 | Address range to read the value to be sent |

**Table 3-16. Write Single Register Function**

| MODBUS Function | | Write Multiple Coils | Forces a variable amount of coils |
|---|---|---|---|
| Devices Address | | 1 – 247 | Slave address |
| Quantity | | 1 – 1968 | Quantity to be written |
| MODBUS Address | | 1 – 65535 | Initial writing MODBUS address on slave |
| Polling | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| MasterTool IEC Operand | Type: %M | %MX0 - %MX3186 | Address range to read the value to be sent |
| | Type: %Q | %QX0 - %QX63 | Address range to read the value to be sent |
| | Type: %I | %IX0 - %IX63 | Address range to read the value to be sent |

**Table 3-17. Write Multiple Coils Function**

| MODBUS Function | | Write Multiple Registers | Preset of a variable amount of registers |
|---|---|---|---|
| Devices Address | | 1 – 247 | Slave address |
| Quantity | | 1 – 120 | Quantity to be written |
| MODBUS Address | | 1 – 65535 | Initial writing MODBUS address on slave |
| Polling | | 0 – 10000 ms | Time interval between the relation trigger and a new trigger |
| MasterTool IEC Operand | Type: %M | %MW0 - %MW3186 | Address range to read the value to be sent |
| | Type: %Q | %QW0 - %QW63 | Address range to read the value to be sent |
| | Type: %I | %IW0 - %IW63 | Address range to read the value to be sent |

**Table 3-18. Write Multiple Registers Function**

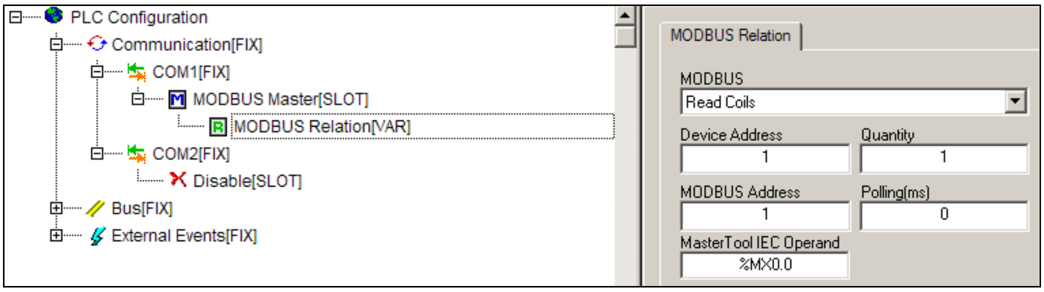


**Figure 3-31. MODBUS Relation**

## MODBUS Slaves

The two COM ports may be configured as MODBUS Master or Slave. To configure COM1 as MODBUS Slave, open the MasterTool IEC, then click “PLC Configuration” located in the “Resources” tab. Then, the module “PLC Configuration” must be expanded.

COM port configurations are located in the Communication module. By expanding this module, “COM1” and “COM2” modules will appear. In order to configure COM1, the “COM1” module must be expanded. Similarly, “COM2 module” must be expanded to configure the communication interface of COM2.

By expanding this block, COM1 port will be disabled for MODBUS. To enable it, it is necessary to click with the right button of the mouse on the “MToolIEC” module, then select the option “Replace element”, and then select “MODBUS Slave” to enable COM1 as MODBUS Slave. For the COM2 Port configuration, the same procedure must be performed, except that for this interface the “MToolIEC” option is not available.

By clicking on the enabled module “MODBUS Slave” a dialog box referring to the MODBUS Slave will appear on the right side:

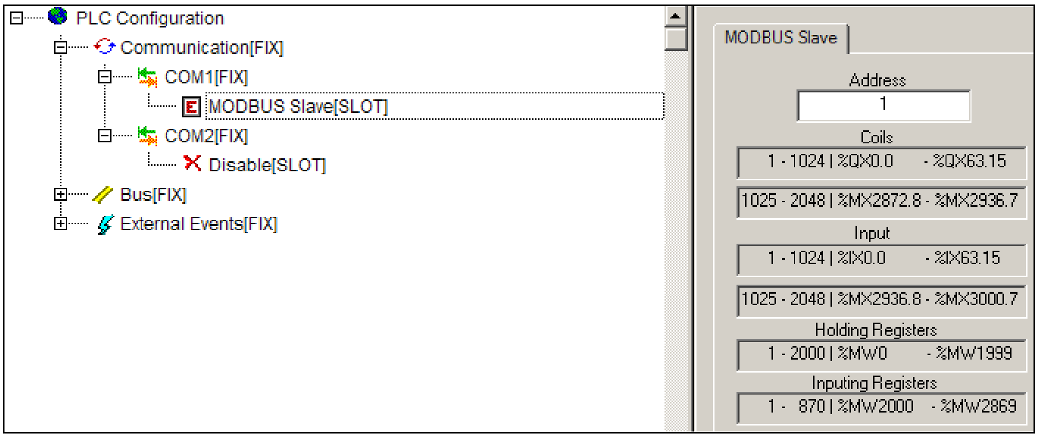


**Figure 3-32. MODBUS Relation**

The only parameter to be configured in the MODBUS Slave is the address, which stands for the slave address. The relation between the MODBUS operands and the MasterTool IEC operands is fixed, and described in the table below.

| Coils | |
|---|---|
| **1 – 1024** | %QX0.0 - %QX63.15 |
| **1025 – 2048** | %MX2872.8 - %MX2936.7 |
| **Input** | |
| **1 – 1024** | %IX0.0 - %IX63.15 |
| **1025 – 2048** | %MX2936.8 - %MX3000.7 |
| **Holding Register** | |
| **1 – 2000** | %MW0 - %MW1999 |
| **Input Register** | |
| **1 – 870** | %MW2000 - %MW2869 |

**Table 3-19. MODBUS Slave Data Area**

Relations between MODBUS operands and MasterTool IEC operands indicate in which memory position each one of the MODBUS operand values are present, thus allowing for the protocol to be used in a simple way. In the following tables are some usage examples:

| MODBUS Address | MasterTool IEC Operands | | | |
|---|---|---|---|---|
| | Coil | Input | Holding Register | Input Register |
| 1 | %QX0.0 | %IX0.0 | %MW0 | %MW2000 |
| 16 | %QX0.15 | %IX0.15 | %MW15 | %MW2015 |
| 17 | %QX1.0 | %IX1.0 | %MW16 | %MW2016 |
| 1024 | %QX63.15 | %IX63.15 | %MW1023 | - |
| 1025 | %MX2872.8 | %MX2936.8 | %MW1024 | - |
| 2048 | %MX2936.7 | %MX3000.7 | - | - |

**Table 3-20. MODBUS Address and MasterTool IEC Address**

| Reserved Operand | Description | MasterTool IEC Operand | MODBUS Address | Size | MODBUS function |
|---|---|---|---|---|---|
| AO0 | Analog Output | %QW3 | 49 | 16 | Write Multiple Coils |
| AO1 | Analog Output | %QW4 | 65 | 16 | Write Multiple Coils |
| AI0 | Analog Input | %IW4 | 65 | 16 | Read Discrete Inputs |
| AI1 | Analog Input | %IW5 | 81 | 16 | Read Discrete Inputs |
| AI2 | Analog Input | %IW6 | 97 | 16 | Read Discrete Inputs |
| AI3 | Analog Input | %IW7 | 113 | 16 | Read Discrete Inputs |

**Table 3-21. Analog I/O MODBUS Address for Direct Access**

## Generic Communication Protocol

Communication ports (COM1 and COM2) may be configured to support a generic communication protocol, that it is able to send and receive frames with up to 256 characters. In order to do this, the "Generic Protocol" element must be selected. The following figure illustrates this PLC Configuration.

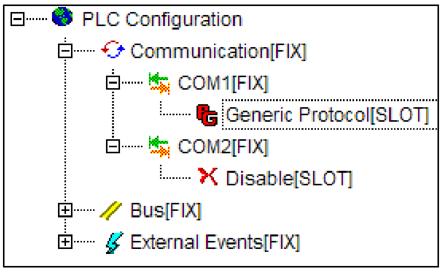


**Figure 3-33. "Generic Protocol" Option for Communication Ports**

| |
|---|
| ATTENTION:<br>This feature is available only as from the executive version (1.02) and Mastertool IEC (1.01). |

To select this option in the communication port, it is necessary to replace the element currently configured. By pressing the right mouse button on the desired port, a submenu appears and the command "Replace element" may be performed. The following figure illustrates this procedure.

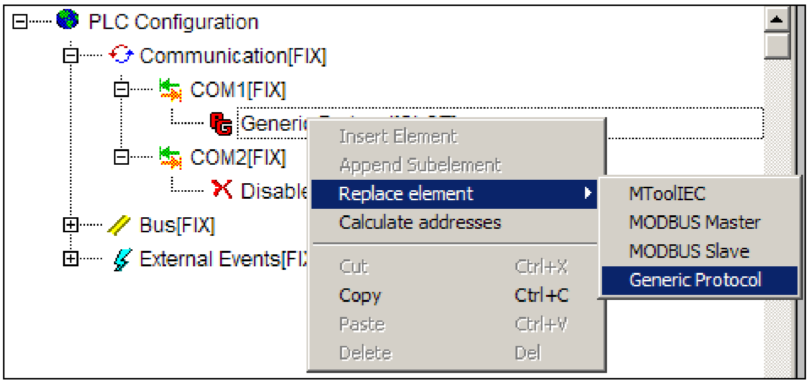


**Figure 3-34. Replace Element Procedure on the Communication Port**

### *UartLib Library*

In order to use all the resources of the generic protocol communication functions, the library **UartLib** (extension ".lib") must be added.

To include UartLib in the Project, select the "Insert" menu in the Library Manager, where the command "Additional Library…" (shortcut key "Ins") is available. The following figure shows this selection.

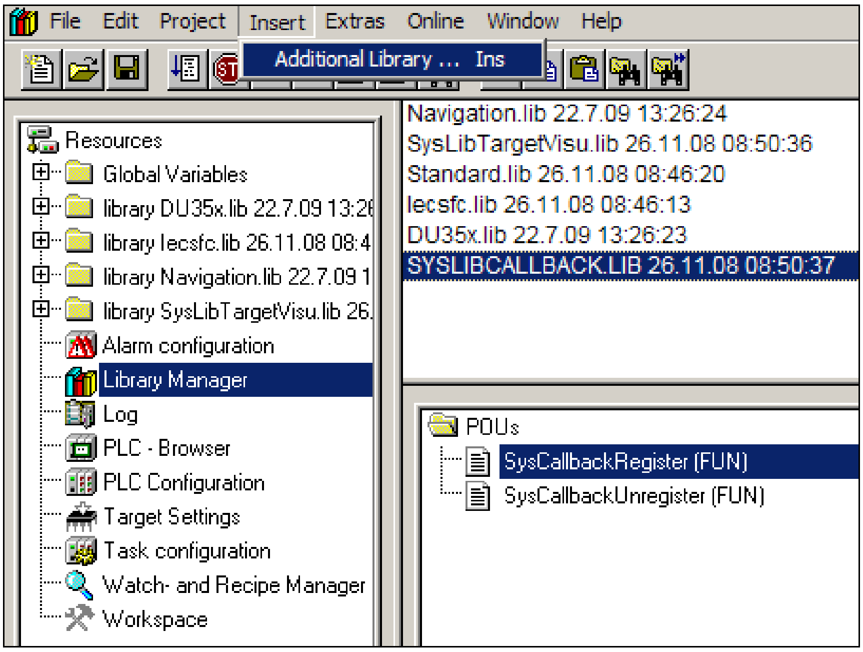


**Figure 3-35. Library Inclusion Procedure**

Next, select the library to be included in the project by pressing the "Open" Button (see in the next figure).

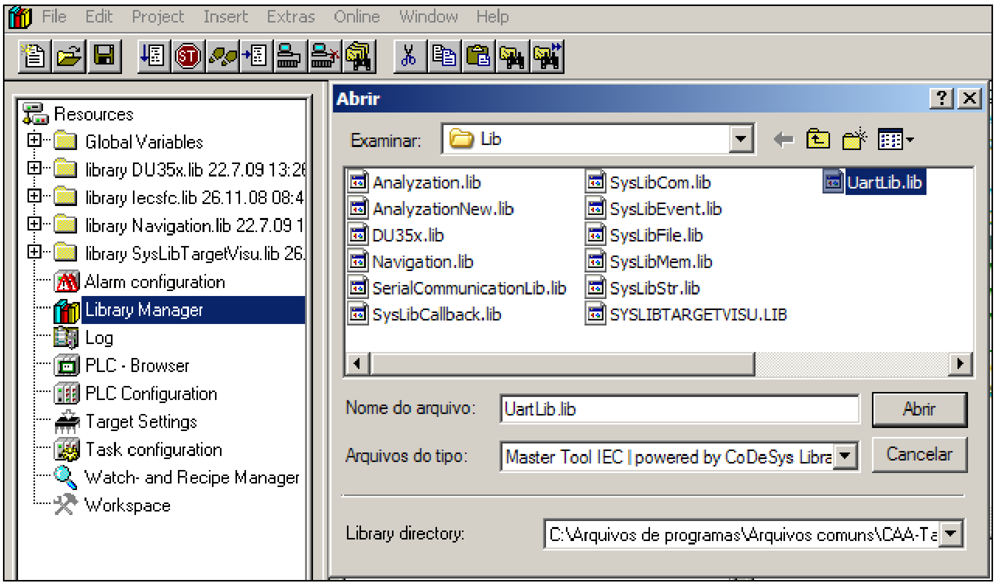


**Figure 3-36. UartLib Library Inclusion**

*SerialCommunicationLib Library*

The generic protocol operates through a specific library ("extension.lib") called **SerialCommunicationLib,** which must be added to the project so that it can be used.

The procedure is the same as of the **Uart** library inclusion.

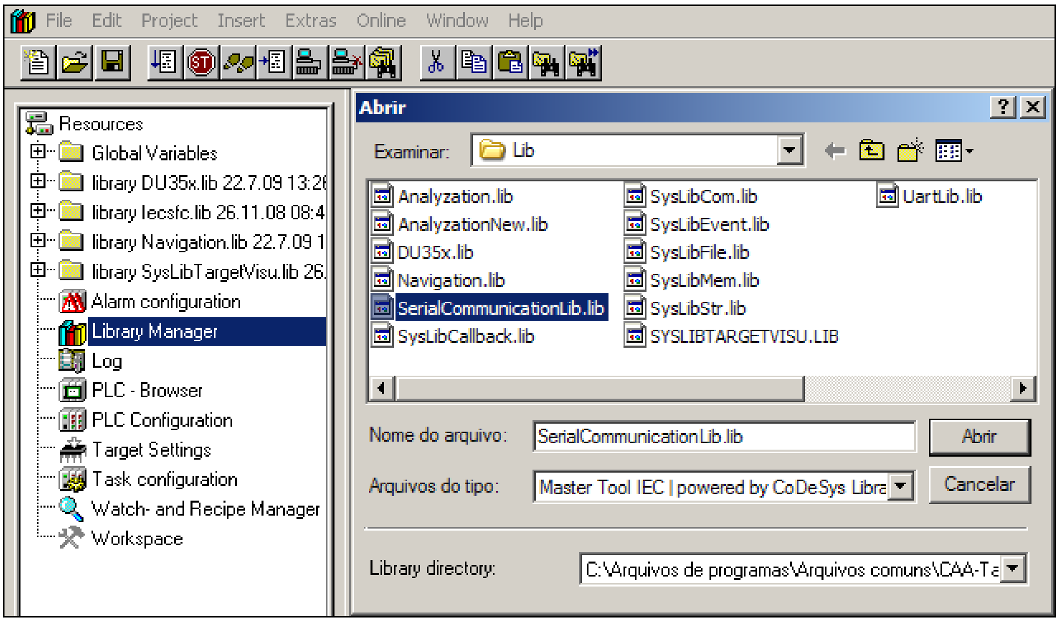


**Figure 3-37. Communication Library Inclusion**

After the inclusion, the library is shown as in the next figure.

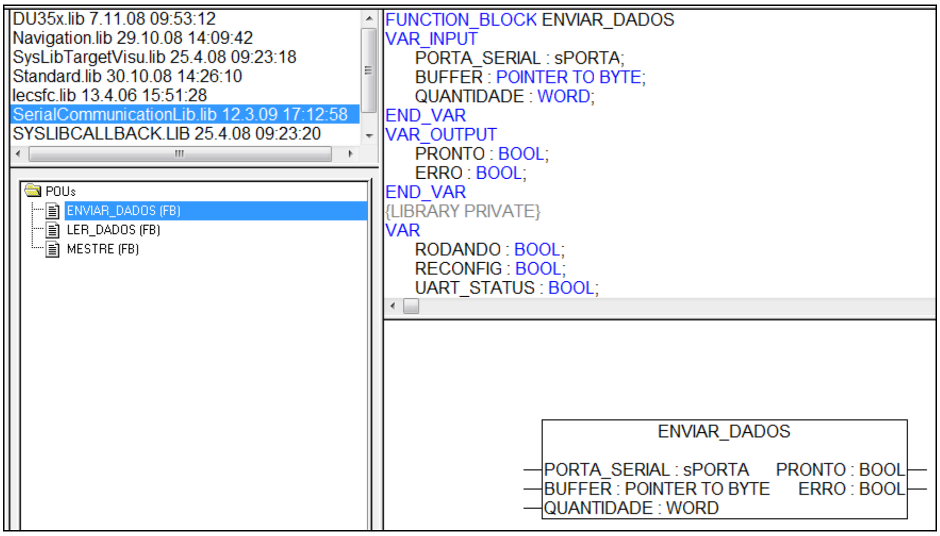


**Figure 3-38. Loaded Communication Library**

This library is formed by three function blocks: ENVIAR_DADOS, LER_DADOS, and MESTRE. Function block is POU type (Program Organization Unit), which is a encapsulated software element, which can be reused. It defines behavior (internal logic), data structure (instance), and external interface (input/output parameters).

### *ENVIAR_DADOS*

This function block has three inputs (VAR_INPUT) and two outputs (VAR_OUTPUT) as shown in the next figure.

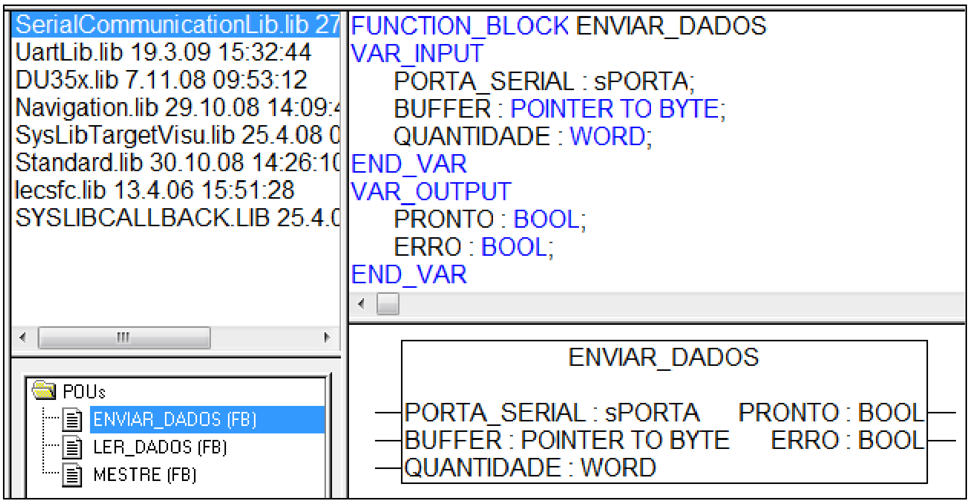


**Figure 3-39. Declaration of the I/O of ENVIAR_DADOS Function Block**

### Inputs Description

The **PORTA_SERIAL** input is of the structure type (sPORTA). Structure is a cluster of elements of different data types. The sPORTA is formed by the following elements:

- PORTA (BYTE type), which allows the user to select the type of the communication port. Option “1” is equivalent to RS-232, and option “2” to RS-485
- BAUDRATE (DWORD type), where the communication speed is defined by accepting a range value from 1200 to 115200. The “bps” unit is implicit
- PARIDADE (BYTE type), which enables the following configurations: 0 – no parity; 1- ODD; 2 – EVEN; 3 – FORCED_0; 4 – FORCED_1
- STOPBITS (BYTE type), which allows the options 1 or 2 stop bits (1 – 1 stop bit and 2 – 2 stop bits)
- DELAY (TIME type) defines the minimum time interval between the receipt and a new sending. This input is not used for this function block

The **BUFFER** input (POINTER TO BYTE) configures the buffer memory address. In this case, the MasterTool IEC ADR function must be used

The **QUANTIDADE** input (WORD) defines the number of bytes to be sent/received and may vary from 1 to 256

### Outputs Description

The outputs are activated in accordance with the behavior described in the following section (operation). They remain in their states for the period of a cycle, and are cleared in the next cycle.

### Operation

When the function block is called, it checks whether the channel is not being used by another instance. If it is not in use, the next step to be performed is checking the configuration. If any of the parameters of the sPORTA structure has changed in relation to the last setting, the channel is reconfigured. If any problem occurs, the ERRO output is enabled and the implementation of the instance is terminated.

If the port opening is successful, the instance signalizes that it is using the serial port. Then, data is ready to be sent, and the process is initialized. If startup fails, the ERRO output is enabled, and the instance is terminated by releasing the communication channel. In the case the startup is successful, the instance releases the processing because it does not lock the application until the end of the sending. Thus, it is necessary to process the function block with a POOLING according to the user needs regarding the recognition of the end of writing. It is flagged by the PRONTO output.

To implement protocols that require silence in the line to signalize the end of the package, the POU

GET_TIME (DU35x .lib library) must be used to create a delay as a protocol requirement.

### *LER_DADOS*

This function block has three inputs (VAR_INPUT), four outputs (VAR_OUTPUT) and one input/ output parameter (VAR_IN_OUT) as shown in next figure:

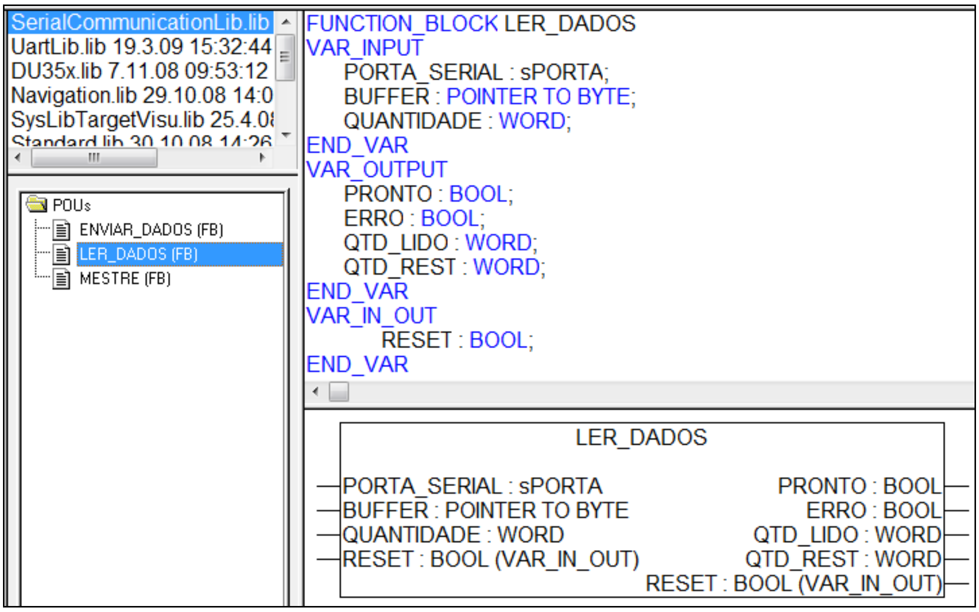


**Figure 3-40. Declaration of the I/O of LER_DADOS Function Block**

### Inputs Description

Just like it was shown in the function block described above, the **PORTA_SERIAL** is of the structure type (sPORTA). The sPORTA is formed by the following elements:

- PORTA (BYTE type), which allows the user to select the type of the communication port. Option “1” is equivalent to RS-232, and option “2” to RS-485
- BAUDRATE (DWORD type), where the communication speed is defined by accepting a range value from 1200 to 115200. The “bps” unit is implicit
- PARIDADE (BYTE type), which enables the following configurations: 0 – no parity; 1- ODD; 2 – EVEN; 3 – FORCED_0; 4 – FORCED_1
- STOPBITS (BYTE type), which allows the options 1 or 2 stop bits (1 – 1 stop bit and 2 – 2 stop bits)
- DELAY (TIME type) defines the minimum time interval between the receipt and a new sending. This input is not used for this function block

The **BUFFER** input (POINTER TO BYTE) configures the buffer memory address. In this case, the MasterTool IEC ADR function must be used.

The **QUANTIDADE** input (WORD) defines the number of bytes to be sent/copied and may vary from 1 to 256.

The **RESET** variable (BOOL type) acts as an I/O parameter (VAR_IN_OUT). It clears the status machines and the internal data of the BUFFERS.

### Outputs Description

The outputs are activated in accordance with the behavior described in the following section (operation). They remain in their states for the period of a cycle and are cleared in the next cycle.

### Operation

When the function block is called, it checks whether the channel is not being used by another instance. If it is not in use, the next step to be performed is checking the configuration. If any of the parameters of the sPORTA structure has changed in relation to the last setting, the channel is reconfigured. If any problem occurs, the ERRO output is enabled and the implementation of the instance is terminated

If the port opening is successful, the instance signalizes that it is using the serial port. Then, data is ready to be sent, and the process is initialized. If startup fails (FRAMMING, parity error, STOPBITS...), the ERRO output is enabled and the instance is terminated by releasing the communication channel. In the case the startup is successful, the instance releases the processing because it does not lock the application until a FRAME receiving. Thus, it is necessary to process the function block with a POOLING according to the user needs regarding the recognition of the end of writing. It is flagged by the PRONTO output.

With such signals, the amount of bytes read and copied to the user buffer is quantified at QTD_LIDO output. If there are bytes that were not copied at UART BUFFER, this amount is expressed in the QTD_REST output. The READY signal will remain active until all bytes received by UART are consumed by the user, which means that the remaining amount must be ZERO.

**NOTE**:
1- In the function blocks LER_DADOS and ENVIAR_DADOS the sPORTA structure delay must not be used. Thus the blocks can operate individually, with no time interval between sending and receiving. If these two functions are in use together to implement a protocol, the delay should be controlled in the application.

**NOTE**:
The function blocks LER_DADOS and ENVIAR_DADOS must not be disabled during data transmission and reception, always it must wait until PRONTO output sets a value that inform the operation was finishing. In case that this condition is not respected, the function block maybe does not release the serial port to a new transmition or reception. It is necessary to write a FALSE value in PORTA_OCUP[x] (x = 0 to COM1 and x = 1 to COM2) global variable to release the serial port manually.

### *MESTRE*

This function block has six inputs (VAR_INPUT), four outputs (VAR_OUTPUT) and one input/output parameter (VAR_IN_OUT) as shown in the next figure:

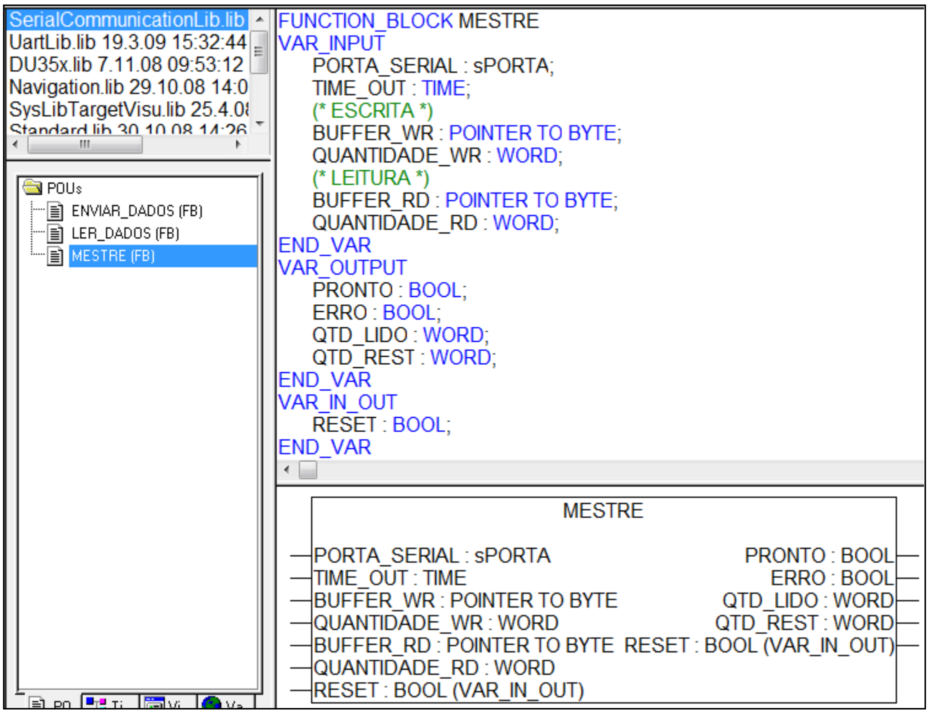


**Figure 3-41. Declaration of the I/O of MESTRE Function Block**

## Inputs Description

The **PORTA_SERIAL** input is of the structure type (sPORTA). The sPORTA is formed by the following elements:

- PORTA (BYTE type), which allows the user to select the type of the communication port. Option “1” is equivalent to RS-232 and option “2” to RS-485
- BAUDRATE (DWORD type), where the communication speed is defined by accepting a range value from 1200 to 115200. The “bps” unit is implicit
- PARIDADE (BYTE type), which enables the following configurations: 0 – no parity; 1- ODD; 2 – EVEN; 3 – FORCED_0; 4 – FORCED_1
- STOPBITS (BYTE type), which allows the options 1 or 2 stop bits (1 – 1 stop bit and 2 – 2 stop bits)
- DELAY (TIME type) defines the minimum time interval between the receipt and a new sending. This input varies from 5 ms (T#5ms) to 1 second (T#1000ms)

The **TIME_OUT** input (TIME type) defines de maximum time interval between the end of the sending action and  the moment of the receiving action. This input may vary from100 ms (T#100ms) to 10 seconds (T#10000ms).

The **BUFFER_WR** input (POINTER TO BYTE) defines the memory address of the writing buffer. In this case, the MasterTool IEC ADR function must be used.

The **QUANTIDADE_WR** input (WORD) defines the number of bytes to be sent and may vary from 1 to 256.

The **BUFFER_RD** input (POINTER TO BYTE) defines the memory address of the reading buffer. In this case, the MasterTool IEC ADR function must be used.

The **QUANTIDADE_RD** input (WORD) defines the number of bytes to be received and copied to the BUFFER and can vary from 1 to 256.

The **RESET** variable (BOOL) acts as an I/O parameter (VAR_IN_OUT). It clears de status machines and the internal data of the BUFFERS.

### Outputs Description

The outputs are activated in accordance with the behavior described in the following section. They remain in their states for the period of a cycle, and are cleared in the next cycle

### Operation

When the function block is called, it checks whether the channel is not being used by another instance. If it is not in use, the next step to be performed is checking the configuration. If any of the parameters of the sPORTA structure has changed in relation to the last setting, the channel is reconfigured. If any problem occurs, the ERRO output is enabled and the implementation of the instance is terminated. If the port opening is successful, the instance signalizes that it is using the serial port.

Then, data is ready to be sent, and the process is initialized. If startup fails, the ERRO output is enabled and the instance is terminated by releasing the communication channel. In the case the startup is successful, the instance releases the processing because it does not lock the application until the end of the sending .Thus, it is necessary to process the function block with a POOLING according to the user needs. This value must be lower than the lowest DELAY of the others devices connected to the net.

With such signals, the amount of bytes read and copied to the user buffer is quantified at QTD_LIDO output. If there are bytes that were not copied at UART BUFFER, this amount is expressed in the QTD_REST output. The PRONTO signal will remain active until all bytes received by UART are consumed by the user, which means that the remaining amount must be ZERO.

When the sending is complete, the receiving is released and the process is initialized. If data receiving fails (FRAMMING, parity errors, STOPBITS, response higher than 256 bytes...) the ERRO output is enabled and the instance is terminated by releasing the communication channel.

In the case the startup is successful, the instance releases the processing because it does not lock the application waiting for a FRAME receipt. The function must be processed with a POOLING. The package receipt is flagged by the PRONTO output. With such signals, the amount of bytes read and copied to the user buffer is quantified at QTD_LIDO output. If there are bytes that were not copied at UART BUFFER, this amount is expressed in the QTD_REST output. The PRONTO signal will remain active until all bytes received by UART are consumed by the user, which means that the remaining amount must be ZERO .

**NOTES**:
1- In the function block MESTRE there is no indication that the requisition frame has just been sent, there are only indications that the response packet was received by the UART (PRONTO on), and the response packet was copied to the application BUFFER of the user (PRONTO off). Therefore, it is not possible to use MODEM signals, since there is no way to know the exact moment you should turn the RTS off.
2-The silence interval time to end a package is of 5 characters.

**NOTE**:
The serial UART used does not detect errors when the communication occurs with fewer bits. For instance: a computer configured with 8 data bits and communicating with a Duo with 8 data bits and without parity. In this case the functions of communication will indicate error. The byte received should be made to check that the frame is as expected. The same happens if this computer was configured for 5 data bits, for example.

#### *MODEM Signals*

MODEM signals may be accessed through special variables. The following figure illustrates the declaration of global variables in the reserved operands library focusing on the assignment of MODEM RTS and CTS signs to the % MB6373 and MB6374% memory position respectively.

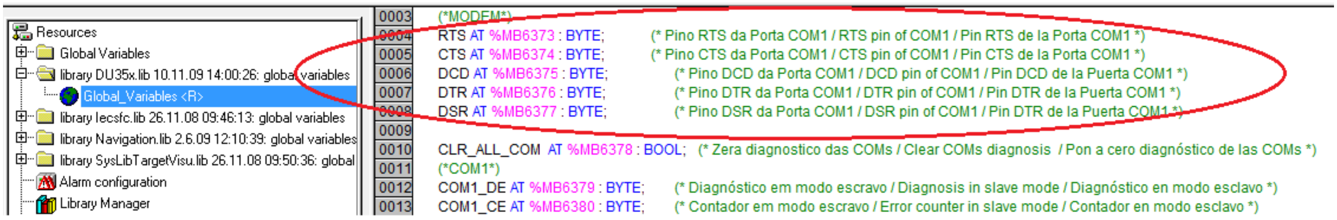

**Figure 3-42. MODEM Signals**

| ATTENTION: |
|---|
| The reserved variables are listed in the chapter Diagnostic (Reserved Operands List). |

### *Best Programming Practices*

The function blocks are designed to be used with only one instance. The input and output BUFFERS may be changed in accordance to the users' needs. Thus the creation of multiple instances may lead to unwanted behavior.

Another point which must be taken into account is the fact that the MESTRE function block was designed to facilitate the development of applications. Hence its simplicity, and the lack of MODEM signal control. If these signals are required, the use of the function blocks ENVIAR_DADOS and LER_DADOS is recommended.

### *BUFFER Overflow*

The Duo BUFFER is able to receive 256 bytes, as well as the sending BUFFER. In case of trying to write over 256 bytes, the function block will signalize an error through the proper output and will not transmit anything. On the other hand, the receiving function block, that is able to receive more than 256 bytes, will be able to copy to the BUFFER and will indicate the BUFFER overflow via PRONTO and ERRO outputs.

### Priority Management

The MESTRE and ENVIAR_DADOS function blocks hold preference over the LER_DADOS function block. Thus, these blocks can stop the LER_DADOS block instances and take control of the channel. The MESTRE and ENVIAR_DADOS function blocks do not interrupt each other; each block must complete its action (whether successful or not) before the other is able to start its process.

The use of multiple instances must be controlled in accordance with its outcome. For by interrupting the processing of one of them before its completion, the channel may remain allocated to that given instance and the others will not be able to act on it.

| ATTENTION: |
|---|
| Special attention should be given when configuring cyclical task assignments in KEYBOARD_USAGE (concerning its execution interval). Unexpected behavior may occur if a communication request is made before the serial is reconfigured. |

ATTENTION:
The processing task of *MAINTARGETVISU_PAINT_CODE* display may consume up to 30 ms. Due to this feature in a communication instance in which the pooling of a master request is less than that time, a loss of parts of the package in which the processing is running is likely. To prevent loss of packets, it is recommended , for generic communication the master pooling time to be greater than 30 ms. The time of the MAIN task must also be added to the master pooling time when the main task is greater than 1ms (large amount of codes and loops may increase runtime). The time may be measured by output triggering and immediate updating with the AES blocks, or via timers in its own application.

## HMI – Human Machine Interface

DU350 and DU351 controllers have a monochrome graphics LCD display with a 128 x 64 dot matrix with backlight and contrast control as well as a membrane keyboard with 25 keys used to perform the user interface.

### Graphics LCD Display

The MasterTool IEC development tool used in DU351and DU350 controllers programming, has an integrated HMI programming interface, which provides easy and friendly integration between the application and the HMI.

In order to add a new screen, click on the folder “Visualizations”, next right-click on “Visualizations” (inside the “Visualizations” tab). Then, select the option “Add object”, type in the name of the screen in uppercase, and then click OK.

In the “Visualizations” tab, the screens are sorted in alphabetical order. After the product energization the startup screen will come first. To get a different screen at the startup, the function block functions NAVIGATION or CHANGE_SCREEN should be used.

In order to use the HMI, the *MAINTARGETVISU_PAINT_CODE* function must be added. This function is responsible for the screens updating.

It is recommended to use a task of “Cyclical” type, with a 500 ms time period for the *MAINTARGETVISU_PAINT_CODE* function call. If more frequent updating of the screen is required, the task period should be reduced. By reducing the time between the updating function calls of the **MAINTARGETVISU_PAINT_CODE**, it is likely to occur a loss in the processing capability of the other cycles. The *MAINTARGETVISU_PAINT_CODE* function may consume between 15 and 30 ms depending on the complexity of the screen to be designed.

To display contrast control, the reserved operand CONTRAST is used. This operand may be loaded with integer values between 0 and 100, having equivalence with 0% (lowest possible contrast) and 100% (highest contrast possible). The control of the time during which the backlight will remain lit after some key being pressed may be modified through the BACKLIGHT operand. This operand may be loaded with integer values between 0 and 255. The BACKLIGHT operand value represents time in seconds.

The DU350 and DU351 display is a monochrome Graphics LCD display with a 128 x 64 dot matrix. Due to restrictions in the display resolution, the following items available in the IEC MasterTool software cannot be displayed clearly:

**“Polygon” function**: It works properly, but the color-fill functionality of this function has no effect, and the element remains unfilled even though this option is set.

**“Visualization” function**: this function operation runs properly, but it is not recommended for DU350 and DU351 because it takes too much memory space.

**“Trend” function:** not supported by the product.

**“Scrollbar” function:** not supported by the product.

**“Button” function**: this function works properly in both the MasterTool IEC and the HMI. However, in supervisory mode it is possible to associate the Button with a mouse click, while in the HMI the association of the Button with any other key must be done by performing actions in Visualization (Extras – Keyboard usage).

**“Rounded Rectangle” function**: not supported by the product.

**“PIE” function**: not supported by the product.

**“WMF file” function:** not supported by the product.

**“Table” function:** this function works properly in the MasterTool IEC. In the HMI of DU350 and DU351 controllers, the tables are properly shown, but they cannot be edited via keyboard.

**“Bitmap” function**: not supported by the product.

**“Alarm Table” function**: not supported by the product.

**“ActiveX Element” function**: not supported by the product.

**“Scroll bar” function**: not supported by the product.

**“Line width” configuration:** this configuration works only when associated with a Rectangle-type object.

**“Colors” configuration**: the HMI has a monochrome Graphics LCD display with a 128 x 64 dot matrix, therefore all colors may be converted into black/white.

| ATTENTION:<br>Some items available in the software may not appear in the PLC as displayed by the MasterTool IEC, consequently demanding adjustments in the size of objects, guidelines and content displayed. |
|---|

## Keyboard

The keyboard of DU350 and DU351 controllers has 25 keys membrane type.

The action of a button may be associated with any screen (Visu) or it may be performed through the use of the function isKeyPressed().

To perform the handling of a key in relation to the enabled screen, select the desired screen and click on the menu “Extras”– “Keyboard functions”. It causes the pop up of a window containing the button actions that will occur when the screen is enabled. To change the screen while pressing a button, select the option “Zoom ” in the field “Action”, then select the desired key in the field “Key” and type in the name of the screen that must be enabled while pressing the selected key on the field “Expression” .

The table below shows the association of the keys with the key names in the MasterTool IEC software. For instance: the use of “Up Arrow” to enable a screen must be selected from the option “VK_UP ” from the field “Key”, “ Zoom”, from the field “Action”, and then typing in the name of the key to be enabled in the field “Expression”.

To associate keys and screens, POU MAINTARGETVISU_INPUT_CODE must be added to the project. This POU is set internally, and used to update the keyboard; we recommend the use of a “Cyclical”- type task with a time period of 20 ms to call MAINTARGETVISU_INPUT_CODE POU. The treatment of the keyboard identifies the pressing of only one key at a time; if two keys are pressed simultaneously the system understands that there is no key pressed. Thus, the keyboard should be used by pressing just one key at a time. If a key remains pressed the keyboard treatment will consider only once pressing regardless of the time it remains pressed.

Other options in the “Action” field are shown in the MasterTool IEC Programming Manual

(MU299608).

The use of keys via isKeyPressed() function, which is located in the SysLibTargetVisu .lib library, is performed through the key code. The function returns TRUE only if the key whose code was passed on as a parameter is pressed. Otherwise it returns FALSE. E.g., isKeyPressed (16 # 24,0, 5, 0), returns TRUE if the main key is pressed. It is important to point out that the isKeyPressed input parameters in both DU350 and DU351 controllers are necessarily: key code, 0 and 0.

ATTENTION:
To use the keyboard on a screen or with the isKeyPressed () function, the sampling is used. So, once the key is read, it can not be read again. Therefore, the two features can not be used for the same key.

ATTENTION:
When the function ControleTelas () is used (present in the navigation logic), the MAIN, UP, and DOWN keys cannot be used with the functions IsKeyPressed or in Screens

ATTENTION:
Whenever a key is pressed, its register remains when there is a change in the module. If this is done in the stop mode then an event may be registered as it goes to the “Run” module status.

The table with associations between acronyms and keyboard buttons in the HMI is shown below.

| Keyboard position | Overlay symbol | MasterTool IEC symbol | Key code |
|---|---|---|---|
| 1 | F1 | F1 | 16#70 |
| 2 | F2 | F2 | 16#71 |
| 3 | F3 | F3 | 16#72 |
| 4 | F4 | F4 | 16#73 |
| 5 | F5 | F5 | 16#74 |
| 6 | F6 | F6 | 16#75 |
| 7 | F7 | F7 | 16#76 |
| 8 | Main | VK_HOME | 16#24 |
| 9 | 7 [abc] | 7 | 16#37 |
| 10 | 8 [def] | 8 | 16#38 |
| 11 | 9 [ghi] | 9 | 16#39 |
| 12 | Up arrow | VK_UP | 16#26 |
| 13 | - + <br>. | Not supported (by symbol) | 16#2E |
| 14 | 4 [jkl] | 4 | 16#34 |
| 15 | 5 [mno] | 5 | 16#35 |
| 16 | 6 [pqrs] | 6 | 16#36 |
| 17 | Left arrow | VK_LEFT | 16#25 |
| 18 | Right arrow | VK_RIGHT | 16#27 |
| 19 | Esc | VK_ESCAPE | 16#1B |
| 20 | Enter key | VK_RETURN | 16#0D |
| 21 | 0 | 0 | 16#30 |
| 22 | 1 [tuv] | 1 | 16#31 |
| 23 | 2 [wxzy] | 2 | 16#32 |
| 24 | 3 [%$/] | 3 | 16#33 |

| | | | |
|---|---|---|---|
| **25** | Down arrow | VK_DOWN | 16#28 |

**Table 3-22. Code Table for Keyboard Usage**

***Notes***

**13:** The function of 13 key is changing the variable signal value between "+" and "-" or adding the decimal point "." for real variables. When it is editing strings this key is used to insert the "+", "-" and "-".

*Momentaneous Function Key (KEY_PRESSED)*

Through the KEY_PRESSED function it is possible to read the current status of each key of either the DU350 or the DU351 controller. The usage of this function allows the user to perform continuous actions through the keyboard without the need to press and release a particular key several times, so its operation is different from the isKeyPressed function.

This function has a single parameter, which must be completed with the key code of interest. The key codes are in Table 3 22.

The return of the KEY_PRESSED function is a variable of BOOL type, where TRUE indicates that the key was pressed at the same moment that the function was performed, and FALSE indicates that the key was not pressed.

It is possible to read the status of a particular key with up to three keys pressed at the same time. If more than three keys are pressed at the same time, then the return of function might be not as expected.

Example of the use of the KEY_PRESSED function:

```
LEFT_BOOL := KEY_PRESSED(16#25);
```

When the arrow to the left (VK_LEFT) is pressed, LEFT_BOOL is TRUE, otherwise it is FALSE.

## Editing of Variables

For edit a variable by HMI it is necessary to select the field 'Text input of variable Textdisplay', as it is showed in Figure 3-43, in the object when the variable is showing. The fields 'Min' and 'Max'must be filled to editing numerical variables. These fields have no effect in variable that is editing when these variables are booleans or strings.

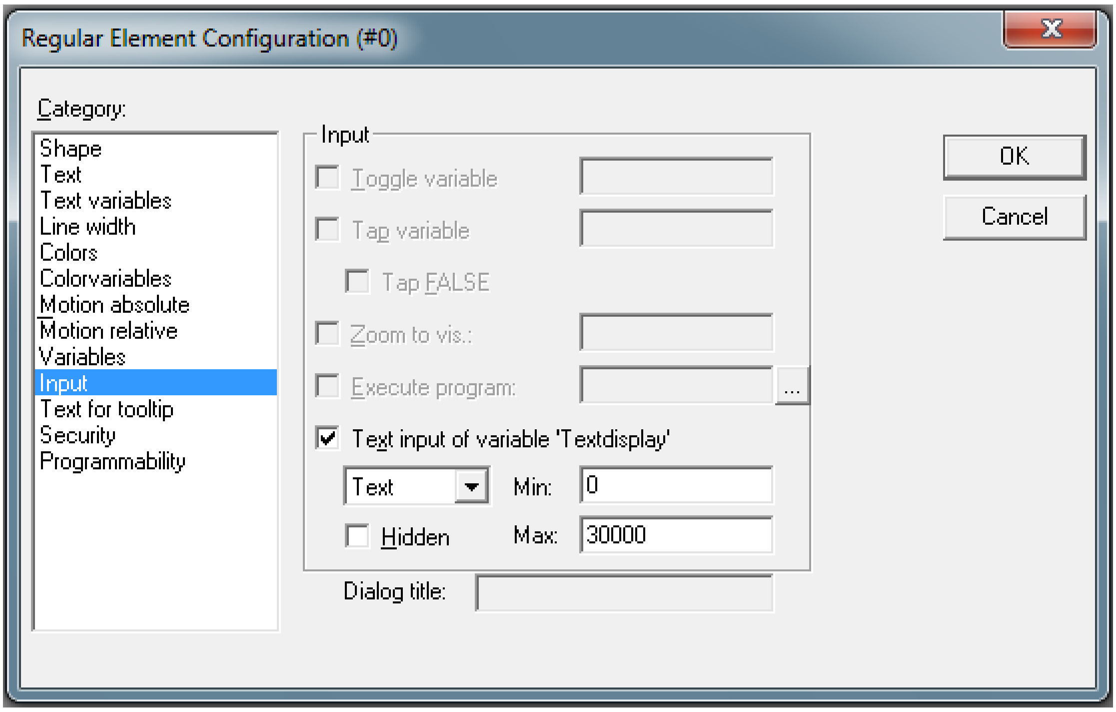


**Figure 3-43. Editing of Variables**

The navigation among the editing objects of the same screen is doing through right and left keys, and it will do the background of the object to stay black. The object can clear pressing the Esc key.

To call the editing of object selected it just press any of numerical keys, and this action will open the edition screen with the value of key pressed in the field of editing variable. This edition is able to be doing in another way, pressing Enter key upon the selected object, however when the edition screen is showed the variable will maintain the current value until it is edited.

Use numerical keys to edit variables and left key to erase any character. If Esc key will press, the edition will be canceled and the main screen will show again. Press Enter key to confirm an edition.

**Editing of Variables type BOOL:** to edit variables of type BOOL, just press in the edit screen the key 0 to 'FALSE' or key 1 to 'TRUE'.

**Editing of variables type DATE, TIME_OF_DAY, DATE_AND_TIME:** its operation is made possible by using the key “3” for the special characters “-” and “:”. It is not necessary to mention the type when this editing occurs via the controller interface as in a PC supervisory.

**Editing of variables type TIME:** the operation is made possible by using the keys “5” and “6” for the characters “m” and “s”. It is not necessary to mention the type when this editing occurs via the controller interface, as in a PC supervisory.

| ATTENTION:<br>The maximum number of editing object in the same screens is 12 objects. |
|---|

For details about formats and separations of the variable types described above, check the MasterTool IEC Programming Manual.

## Shortcut Keys

DU350 and DU351 controllers have 5 keystrokes that perform special operations. They are:

**[MAIN + UP ARROW ]** = display/hide the Special Pages (the browsing between pages is made with the keys upwards and downwards);

**[MAIN + DOWN ARROW]** = toggle between MODBUS Protocol and Programming Protocol (COM1 Port);

**[PRESSING ESC ON INITIALIZATION] =** starts the controller without loading the user application, thus enabling the application rewriting in case of serious failure or a watchdog action. To return to current application, just de-energize and energize the controller without pressing ESC.

The functions of special key sequences are previously configured. The user does not need to perform any special configuration to be able to use them

## Special Pages

DU350 and DU351 controllers comprise 7 special pages previously included in the CPU to make the use and the diagnostic of peripherals easy:

1- **DIGITAL INPUTS** – displays the status of the 20 digital inputs present in the CPU

2- **DIGITAL OUTPUTS** – displays the status of the 16 digital outputs present in the CPU

3- **ANALOG** – displays the values of the registers of the 4 analog inputs and the 2 analog outputs

4- **INFORMATION** – displays the model information (Model), software version (version) and serial number (Serial Number)

5- **COUNTER** – displays the values of the registrars of the 4 counters present in the CPU

6- **CONTRAST** – contrast adjustment page of the graphical display

7- **BACKLIGHT** – backlight adjustment page of the graphical display

To enable and disable the special pages, press MAIN + UP ARROW simultaneously.

To browse through the special pages use UP ARROW and DOWN ARROW.

To change the contrast value, as well as the backlight time value through the special pages, select the page of interest and press LEFT ARROW to decrease, or RIGHT ARROW to increase the value by one unit.

## RTC (REAL TIME CLOCK)

DU350 and DU351 controllers have an internal clock that may be used through the standard .lib. library. The RTC functional block returns the current date and time as from the value which has been previously configured in the format DT # 1970-01-01-00: 00: 00.

**Resolution** = 1 second.

**Maximum variation** = 2 seconds per day.

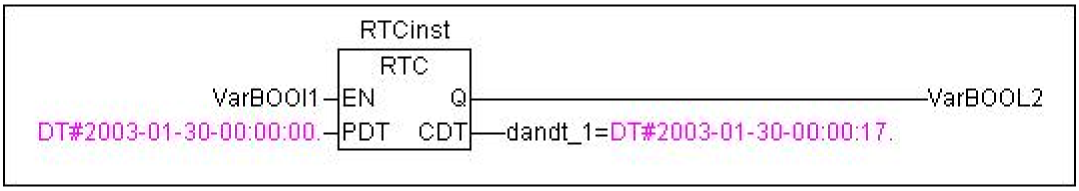


**Figure 3-44. RTC Function Block**

The clock is used via function block in the format RTC (EN, PDT, Q CDT), where EN and PDT are input variables, EN is of BOOL type, and PDT is of DT type. Q and CDT are output variables, Q is of BOOL type, and CDT is of DT type. When EN is FALSE, both outputs Q and CDT are given the values FALSE and DT # 1970-01-01-00: 00: 00, respectively.

In the first EN input rising edge the functional block checks for the occurrence of date and time loss in the real-time clock. In case it has happened, the functional block loads the real-time clock with the PDT value. If the clock did not lose time and date, it will not be updated with the value of the PDT variable.

To update time and date in the real-time clock with the value of the PDT variable when the clock is not signalizing date and time loss (FALHA_RTC), a falling edge followed by a rising edge must occur.

While the EN input signal is enabled, CDT is updated with the date and time of the real-time clock.

The diagnostic of clock loss is mapped in a special operand known as:

**FALHA_RTC -** Indicates information loss when TRUE.

After the handling of the FALHA_RTC operand, the value FALSE may be assigned to it.

The Clock special operand is previously mapped in a specific area of the memory. Thus, it may be used as a global variable. The name of the operand is best described in the special operands list , in the section “Diagnostics”-“Reserved Operands List”.

| ATTENTION:<br>Use dates from the year 2000 up to the year 2105. |
|---|

| ATTENTION:<br>It is not possible the use of the RTC function block in POUs started via time interruption. |
|---|

## Clock Page Display

To display the value of the clock on a application page the CDT output variable (which was used in the RTC function block instance) must be declared in the field ‘Variables’ ‘Textdisplay’ of the object where the clock is shown.

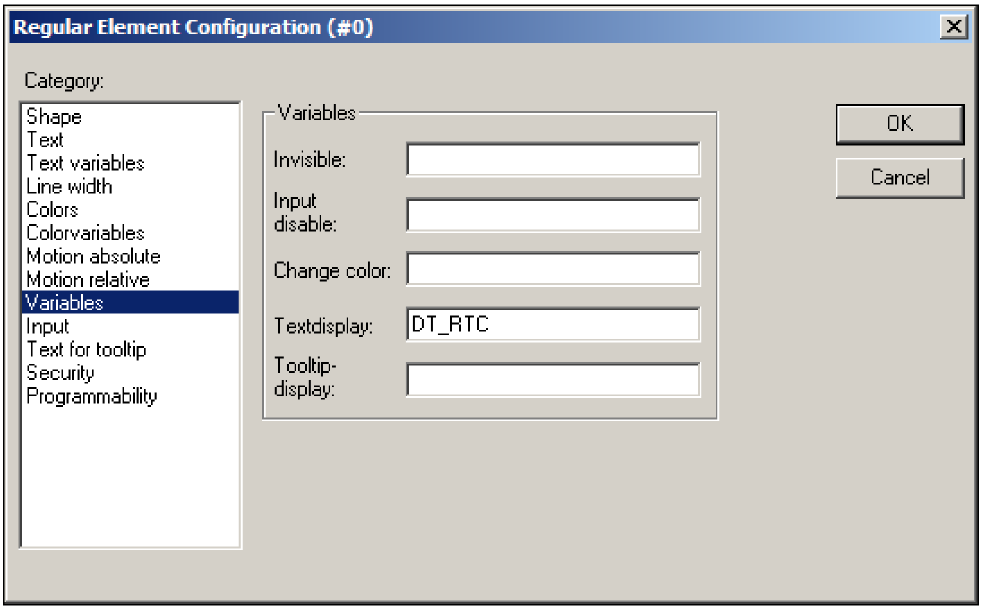


**Figure 3-45. Clock Variable Declaration**

In the field ‘Text’ ‘Content’, the clock value can be displayed using **%s** only. Then the desired text will be included, and instead of **%s** the date and time will be shown.

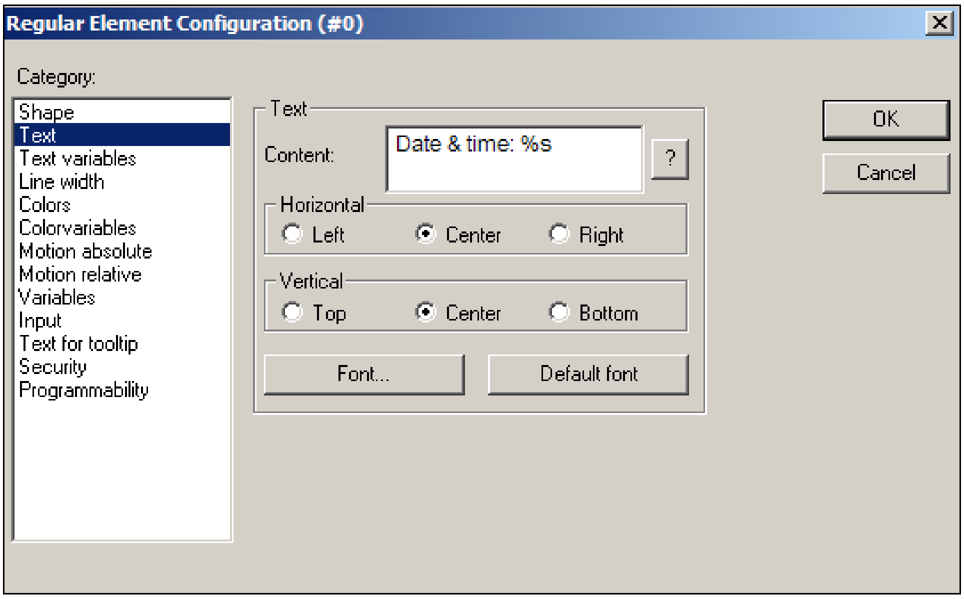


**Figure 3-46. Object Configuration for Clock Display**

In addition to displaying the clock value in the date and time format, it is also possible to view it in either data or hour format only. In order to do this, the conversion functions DT_TO_DATE and DT_TO_TOD are used respectively.

| |
|---|
| ATTENTION:<br>Use %s in lower case. Nothing will be displayed if this declaration is in upper case. |

ATTENTION:
Do not use %t in the pages, for this is used by the MasterTool IEC to display the computer system time where the programmer is running in online mode.

## POU Calls

Each user-defined POU must be linked to a task. A task has priority and range of activation. If two tasks are "ready" to be performed, when the time interval which is defined for each task is past, the task of higher priority runs in first place. If during the execution of a given task, a second higher-priority task becomes "ready" after the time interval defined for that task, it will be performed only after the full execution of the lower-priority task. Each task priority must be set to values 0 - 31 (0 – higher; 31 - lower).

## Time Interruption POUs

If it is required that a given cyclical task holds an accurate range of execution (without being delayed by other less important tasks), it must be configured to be triggered from the external TIMER_INT event. The range of this time-interruption driven task must be configured in the settings tree as follows: open the "Resources" tab and select “PLC Configuration”. Expand the settings tree by clicking on “+” next to "PLC Configuration ", and then expand the external events “option” again by clicking on “+”. Select “Timer event”, and enter the desired interval in the interval in milliseconds (MS). The range must be an integer between 1 and 1000, which corresponds to the time between calls of this task.

In order to create and configure a new task, click “Task configuration”, which is located in the “Resources” tab, click on “+”, next to the phrase “Task configuration”, right-click on “System events” and select the option "Insert task". Change the name of the task as well as its settings as described above. To relate a POU to a task, right-click on the task, and select the option “Includes program call”, then select the desired POU in the field “Call”.

Whenever time-interruption driven tasks (TIMER_INT) are employed, a meticulous check must be performed for another POU of that same type in the common memory addresses (present in the same DWORDmemory, e.g.: Q00 and Q01 operands share the same BYTE (%QB0) and DWORD (%QD0) in memory which is used in time-interruption driven POUs. If this condition occurs when assigning any value to these other POU operands, the function “EXT_EVENT_OFF” must be employed to disable the time-interruption driven task momentarily. To re-enable the task triggered by time-interruption, the function “EXT_EVENT_ON” should be used.

ATTENTION:
When MasterTool IEC is used in the application simulation mode, the time interruption feature is not available.

## Input and Output Instantaneous Updating (AES)

In case of using either a high-cycle time or a routine performed by time-interruption driven POUs, the AES functions may be used to update the operand values of inputs and outputs as well as of their respective diagnostics during a routine processing.

### AES_DIGITAL_INPUT

This function updates the normal digital input operands with the current value of the Ixx digital input.

### AES_DIGITAL_OUTPUT

This command updates the normal digital outputs with the current value of the Qxx digital output operand.

### AES_ANALOG_INPUT

This function updates the value operands of the AIx, AIx_DG and AIx_OPN analog input and analog input diagnostics. The update rate of the analog inputs is 60 ms, that is, by employing the AES_ANALOG_INPUT command, the analog input values are updated with the read values (and processed by the corresponding filter) in the last complete cycle of 60 ms.

### AES_ANALOG_OUTPUT

This function updates the analog outputs with the current value of the AOx operands, and updates the AOx_DG, AOx_ERR diagnostic operands.

### AES_COUNTERS

This function updates the counter with the control operands value CNTx_PRESET, CNTx_CMP0, CNTx_CMP1, CNTx_CMD, CNTx_CLR, CNTx_STOP, CNTx_LOAD, CNTx_AMG, CNTx_OVER and updates both the value and the diagnostic operands of the counters CNTx, CNTx_HOLD, CNTx_DG, CNTx_OVERFLOW, CNTx_UNDERFLOW, CNTx_DIR, CNTx_MAX_CMP0, CNTx_MAX_CMP1, CNTx_EQ_CMP0, CNTx_EQ_CMP1, CNTx_ZERO.

### AES_FAST_OUTPUTS

This function updates the fast outputs with the control operands value Fx_FREQ, Fx_PLS_TOT, Fx_PLS_RMP, Fx_DUTY, PTO_CMD, VFO_CMD, Fx_PTO_START, Fx_PTO_STOP, Fx_VFO and updates the fast output diagnostic operands , Fx_PTO_DG, VFO_DG, Fx_PTO_ON, Fx_PTO_ACE, Fx_PTO_REG, Fx_PTO_DES, Fx_PTO_PRM, Fx_PTO_ERR, Fx_VFO_ON, Fx_VFO_PRM, Fx_VFO_ERR.

This AES function returns TRUE if it was successfully executed; otherwise it is FALSE . This situation occurs only if the function has been called in a time-interruption driven POU during a previous call in another POU (in the main cycle), and if there is some alteration in the operands employed for the PTO fast output control and configuration.

If the operands used for PTO control and configuration are changed only in the main loop or changed in the timer-driven POU, the handling of the AES_FAST_OUTPUTS return function is not necessary (it always returns TRUE for these conditions).

## Navigation through the User Pages via Keyboard (NAVIGATION)

The NAVIGATION function block, which is defined in the Navigation .lib library, enables navigation by the use of the Up and Down directional keys or by using the block control inputs.

Page List (VISU_LIST):

The NAVIGATION Function block has an ARRAY [0..30] OF STRING(10) input . This array comprises 30 string positions of up to 10 characters. Thus, it is possible to use up to 30 pages with navigation through the function block; each page must contain up to 10 characters. There is no need for every application page to be on the page list which is used by navigation. In this case the navigation using the function block will be restricted to the pages which are added to the list.

"MAIN" key, directional up and down:

By pressing the DU350 and DU351 controllers "MAIN" key, the function block enables the added page at position 0 in the page list. By pressing the directional up key, the function block enables the next page of the list as to the last page enabled by the NAVIGATION function block employed. By pressing the directional key down, the function block enables the previous page as to the last page enabled by the NAVIGATION function block employed.

Control inputs ENABLE_UPDW, UP, DOWN:

The ENABLE_UPDW control input enables the use of the UP and DOWN inputs. The UP input behaves the same way as the UP directional key and DOWN behaves the same way as the DOWN directional key.

Example of use:

In the project template as well as in the chapter “Initial Programming”-“Analyzing the template", an example of the utilization of the NAVIGATION function block is shown:

## Page Enabling (CHANGE_SCREEN)

The CHANGE_SCREEN function, defined in the Navigation .lib library, enables a given page defined in VISU_NAME input when the ENABLE input is TRUE. This function is used to enable a screen due to any user-defined internal logic.

**Page to be enabled (VISU_NAME)**

The CHANGE_SCREEN Function block has a STRING (10) input . This input must contain the name of the user page to be enabled when the ENABLE input is TRUE. The page name must have a maximum of 10 characters.

**ENABLE input**

When the ENABLE input is TRUE, the VISU_NAME page is enabled. When it is FALSE, the VISU_NAME page remains updated until another page is enabled.

## Upload

DU350 and DU351 controllers enable the recording of a project in the product memory, which can be recovered and reused through the MasterTool IEC software.

To store a project in the product memory the DU350/DU351 must be in the “logged” mode, and “stop” state. In the “Online” menu the option “Source code download” should be selected.

To retrieve the previously stored project the option "Open" in the "File" menu must be selected. On the file selection page, click on the "PLC ..." button. On the next screen, the DU35x controller must be selected in the “Configuration” field.

| |
|---|
| ATTENTION:<br>The size of memory area used to store a project in DU350/DU351 is 256kB. |

| |
|---|
| ATTENTION:<br>Upload retrieves the last project stored in the controller as described in the preceding paragraphs. If the loading occurs only with the purpose to run a given application, it will not be possible to recover it via the Upload procedure. |

## Watchdog

DU350 and DU351 controllers comprise a watchdog system that indicates to the user that a cycle time greater than 2 seconds has occurred. Whenever a watchdog is identified, the CPU displays a page informing that the system entered the watchdog status. In this case, the physical outputs go to the safety State and the CPU remains locked until it is restarted (de-energized and energized again). The WATCHDOG operand is set when there is a watchdog and this operand may be read at the next system startup. If the operand value is TRUE, it means that there was a watchdog action in the previous execution. The operand may be cleared by writing the value FALSE, which facilitates the execution of check applications, and watchdog diagnostic treatment.

If the application is continuously generating watchdog, in order to write a new application, press ESC when restarting the PLC, as stated in the chapter "Configuration"-"HMI- Human-Machine Interface"-"Keyboard"-"Shortcut keys". Such action prevents the application to run and allows communication with the IEC MasterTool software, so that you can write a new application.

The special operand watchdog indication has been previously mapped in a specific area of the memory. Thus, it may be used as a global variable.

## Brownout

DU350 and DU351 controllers have a brownout system that informs the user when the voltage level in the PLC power supply is below 19 V. There are two ways to diagnose such voltage fall:

1 – If the PLC remains with the voltage below 19 V for more than three seconds, the PLC displays a page indicating a brownout . The physical outputs go to a safe state and the PLC remains locked until it is restarted (de-energized and re-energized).

2 – If the PLC power has a voltage drop which takes it to below 19 V, and then returns to a value greater than 19 V in less than three seconds, the CP is rebooted, and a page signaling that the controller was restarted via brownout is displayed during the boot, and the BROWNOUT special operand is set TRUE, thus indicating that the application is running after a brownout reset. The user may change the BROWNOUT operand value to FALSE during the application execution, facilitating the execution of verification applications and brownout diagnostic treatment.

The special operand BROWNOUT has been previously mapped in a specific area of the memory. Thus, it may be used as a global variable. The name of the operand is better described in the list of special operands in the chapter “Diagnostics”-“Reserved Operands List”.

## System Error

DU350 and DU351 controllers comprise an error identification system that alerts the user whenever there is a critical error event in the system. To identify the error, the controller displays a page reporting the error, then the physical outputs go to a safe State and the controller remains locked until it is restarted (de-energized and re-energized).

If the application is generating the error it is necessary to press the ESC key when restarting the PLC to reprogram the controller, as stated in the chapter "Configuration" - "HMI Human-Machine Interface" - "Keyboard" - "Short keys", thus preventing the application to run, and enabling communication with the IEC MasterTool software, so that you can write a new application. If this procedure does not solve the problem, it is recommended that the user contacts support.

The system errors are identified by the pages containing the information: ERROR + "error number".

## Safe State

While the DU350 and DU351 controller are in the safe state , the digital outputs (common and fast) will be forced to the logic level 0 (FALSE). The analog outputs will be forced to 0 V or 0 A, depending on the way the output is operating. If the analog outputs are disabled, they will be forced to 0 V in case of safe state.

The input goes into the safe state in case of:

- Brownout

- Watchdog
- Error indication
- Controller programming
- Stop (via MasterTool IEC software)

# 4. Installation

## Electrical Installation

> **DANGER:**
> **When performing any electrical panel installation, make sure that its main power supply be de-energized.**

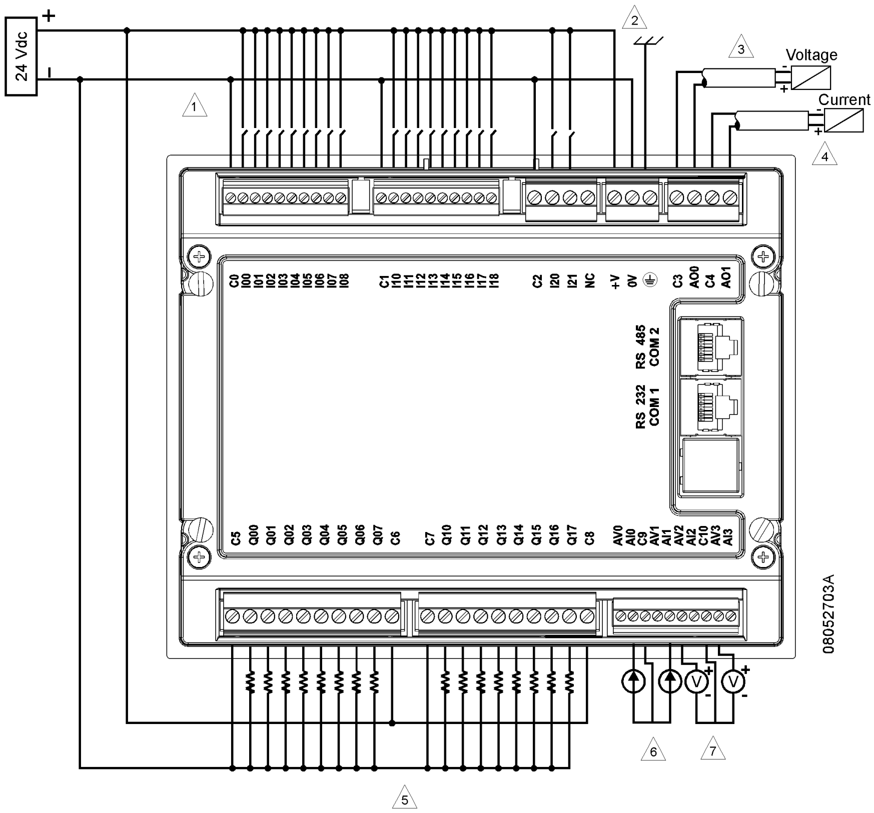


**Figure 4-1. DU350/DU351 Connection Example**

***Diagram Notes***

1 – Typical usage of “sink” digital inputs. C0, C1 and C2 are the common (0 V) to their respective input group I00 to I08, I10 to I18 and I20 to I21.

2 – 24 V power supply with the 24 V, 0 V and ground protection pins.

3 – Example of an analog output configured for voltage mode output.

4 – Example of an analog output configured for current mode output.

5 – Typical usage of digital output transistor types (DU350) and fast outputs (DU350 and DU351). In order to use the relay digital output (DU351) it is not necessary to connect C5 and C7 pins; but only common, C6 and C8.

6 – Example of two analog inputs configured for current mode input.

7 – Example of two analog inputs configured for voltage mode input. The connection of a 0 to 10 V signal in a current input pin may cause damage to the product.

## Connections

The correct connection of the DU350 and DU351 controllers and system modules ensure the equipment safety, and its proper operation. For that purpose, the following must be checked:

- The cables' diameter and insulation voltage must be consistent with the application.
- The cables on mounting panel terminals must be safely and firmly connected.
- The system power supply pins and grounding parts must be strong and well connected, ensuring good current flow.
- The connection between equipment and mounting panel ground must be strong and with the correct cable diameter to ensure proper grounding and noise insulation. It is recommended to use 1.5 $mm^2$ cables.
- It is recommended to identify all cables with plastic rings or similar, to make easy the assembly and maintenance operations.

| ATTENTION:<br>To fix cables at the product terminals blocks an appropriate screwdriver should be used due the size of the fixing screw diameter. The terminals of digital and analog inputs must be screwed with a screwdriver with a maximum width of 2.5 mm at its end. |
|---|

## Grounding

The terminal block ⏚ must be directly connected to the panel grounding bar through a 1.5 $mm^2$ cable.

## Analog Interfaces

Shielded cable: it is recommended the use of shielded cables at the analog inputs and outputs. The shield must be connected to the grounding in only one of the cable ends, preferably grounded on the side of the closet. If the end is connected to the DU350/DU351 for grounding the shield, a terminal should be used as close as possible to the analog inputs and outputs.

Common analog input: analog inputs 0 (AV0 or AI0) and 1 (AV1 or AI1) share the same common terminal block (C9). Similarly, the analog inputs 2 (AV0 or AI0) and 3 (AV0 or AI0) share the same common terminal block (C10). To use two outputs which share the same common terminal block, the two commons of the analog input signals must be connected into a terminal chart. As well as the respective DU350/DU351 common must be connected to the same terminal chart:

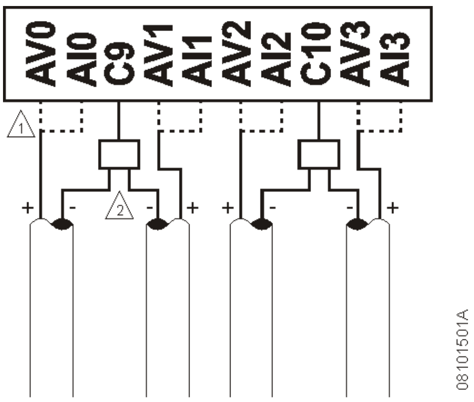


**Figure 4-2. Analog Inputs Connection**

***Diagram Notes***

1 – Each analog input channel has two input terminals AVx and AIx. On the use of the channel as an input voltage only the corresponding AVx terminal should be used. On the use of the channel as current input, the corresponding AIx terminal should be used.

2 – Channel 0 and channel 1 share the same common terminal. The connection should be made with a proper terminal chart (as closer as possible of the equipment).

## Square Encoders Interface

In order to employ encoder signal handling, the encoder must fulfill the functional requirements of fast inputs and the timing of fast inputs in the squaring mode, observing the characteristics of the fast input blocks to be used. The encoder signals A and B must be connected to either the I00 and I01 (Block 0) or the I10 and I11 (Block 1) terminals. Optionally the clear value signal connected to either the I02 (Block 0) or the I12 (Block 1) terminal may be used. The encoder 0 V must be connected to C0 (Block 0) or C1 (Block 1):

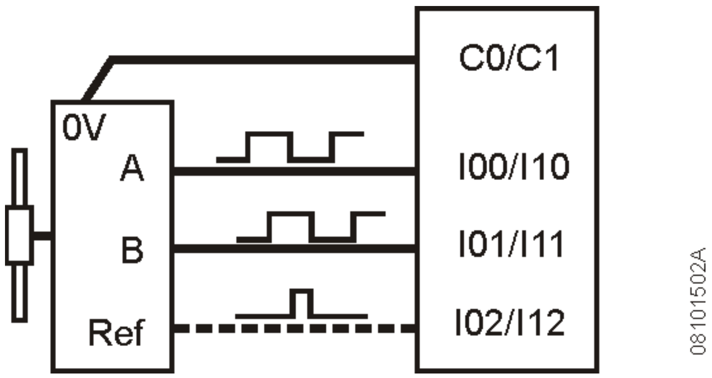


**Figure 4-3. Encoder's Connection Example**

The configurable input may be used as clear value if it is connected to the encoder reference output position.

## Digital Output Protection

To activate inductive loads, a protection diode should be used as close as possible to the load so that there is no occurrence of eventual voltage peaks coming from sudden variation in the inductive load.

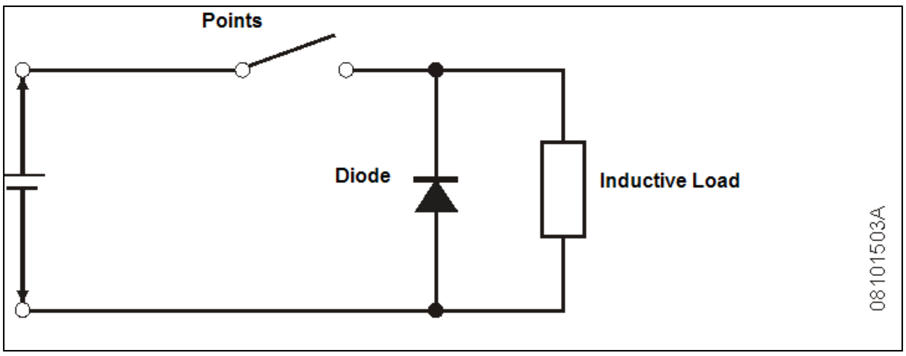


**Figure 4-4. Digital Output Protection Circuit**

## Mechanical Assembly

For the correct installation and operation of this product, the panel must be cut according to the following dimensions shown below. Dimensions in mm.

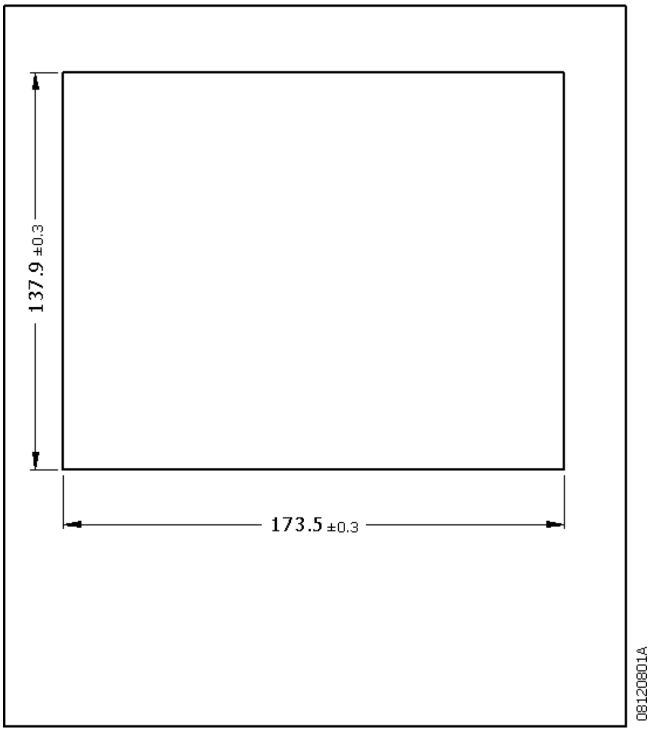


**Figure 4-5. Panel Cut for Installation**

The side tab settings should be collected for the controller installation in the panel.

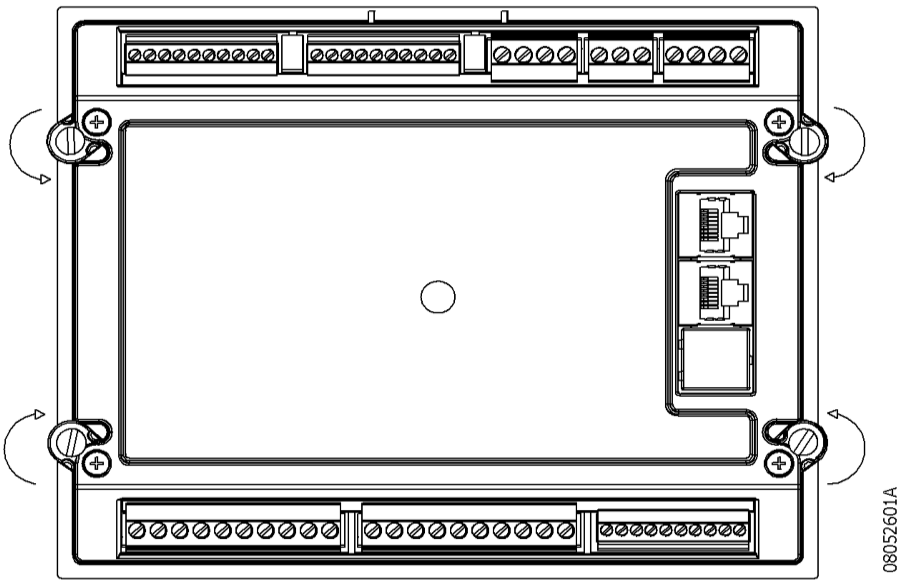


**Figure 4-6. Side Tabs**

The controller must be installed in the panel, docking first the top.

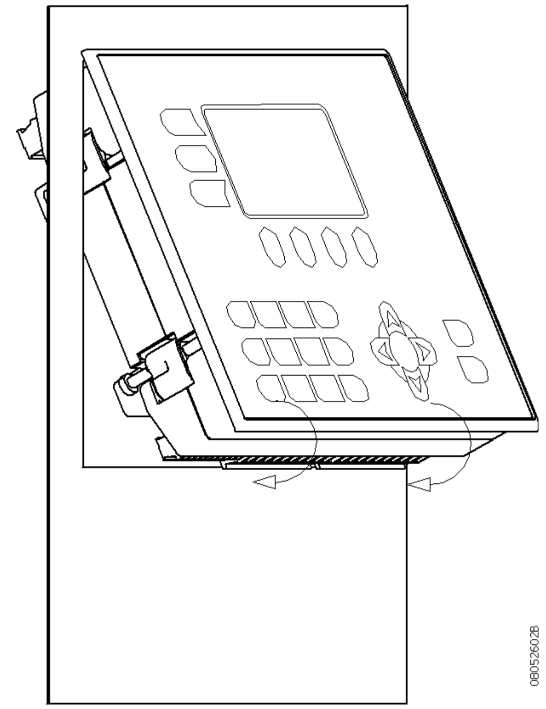

**Figure 4-7. Installation on the Panel**

After duly docked, the tabs must be opened (spinning, like described in the following figure) and the fixing bolts shall be tight using a proper tool.

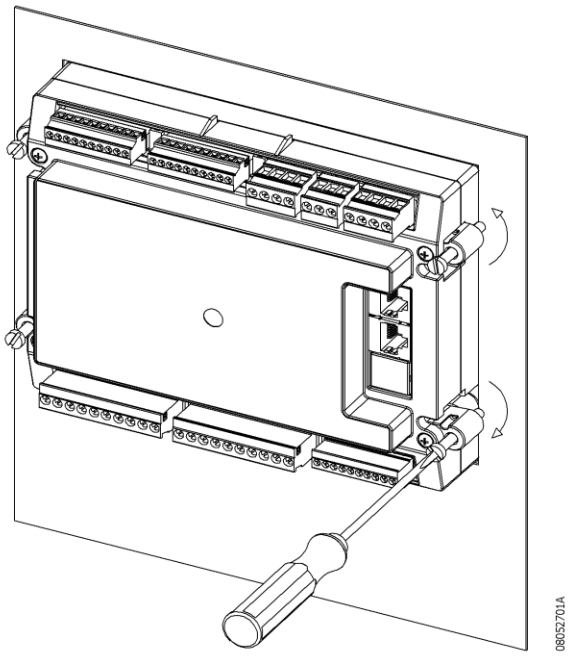

**Figure 4-8. Mechanical Fixing**

After these steps, the controller is mounted in the panel and you may start the field connection wiring. The terminal blocks located at the back of the equipment are identified by the interfaces to which they belong. The terminal blocks are of the screw type and they are detachable in order to facilitate the connection. The field wiring must be connected to the equipment without power sourcing in order to avoid electric shocks. After the connections are checked, the equipment can be powered.

DU350 and DU351 modules have a protective film on the overlay to protect them from damage during transportation, and when being handled in the installation phase. Once the module is installed, this film may be removed allowing for better visualization of the keyboard, and display.

## Programming Installation

To perform the installation of MasterTool IEC development software, first you have to download the setup file on the Altus site (www.altus.com.br).

After the download of the software, close all the active programs on your desktop, and then double-click the installation file. The following installation page will be displayed:


**Figure 4-9. Extracting Files**

Please wait while the installer extracts the requested files to perform the installation.

Then the following pages appear:

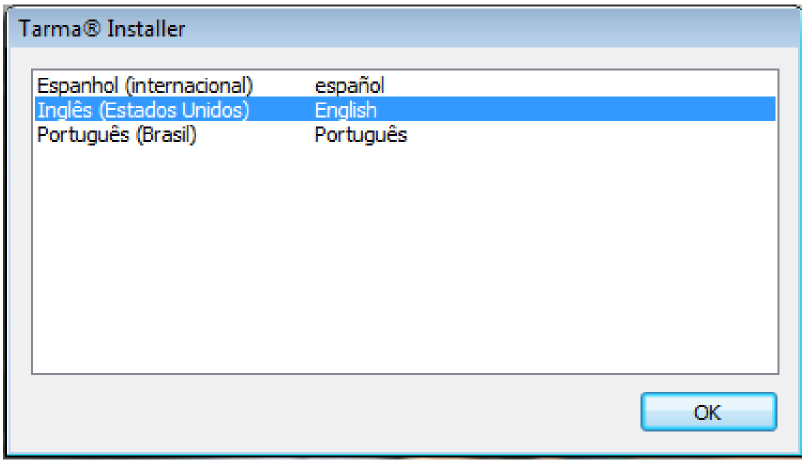


**Figure 4-10. Language Selection**

The language in which the MasterTool IEC will be installed is selected on the first page. Select the desired option and click “OK”.

The next page will be shown indicating the installation start. Click “Next” to continue.

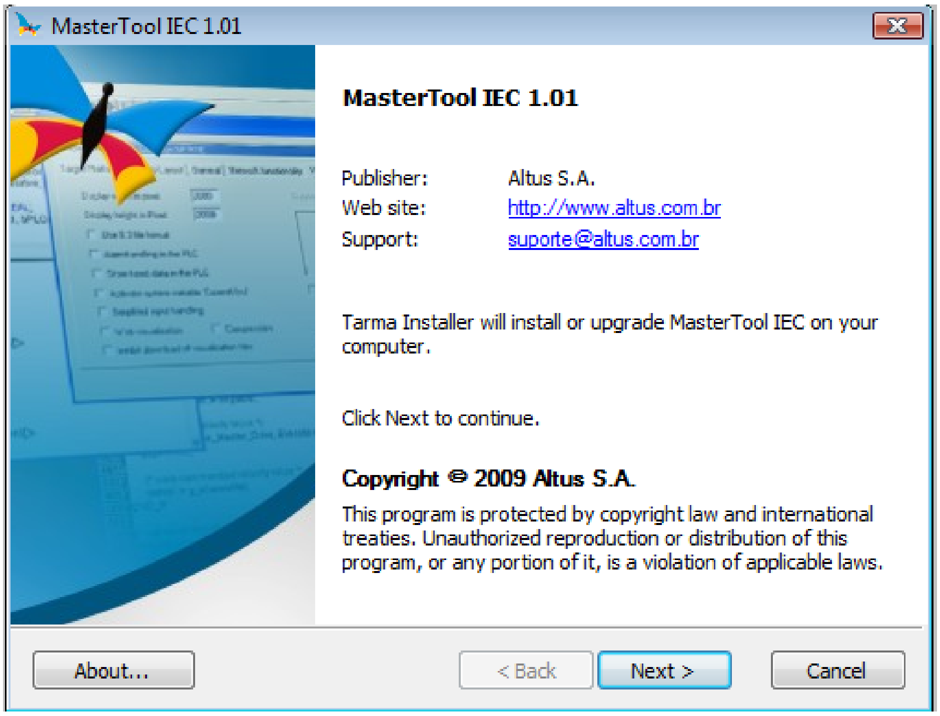


**Figure 4-11. Welcome Page**

The next page shown refers to the license agreement. Read the license carefully, and select the option “I agree with these terms and conditions”.

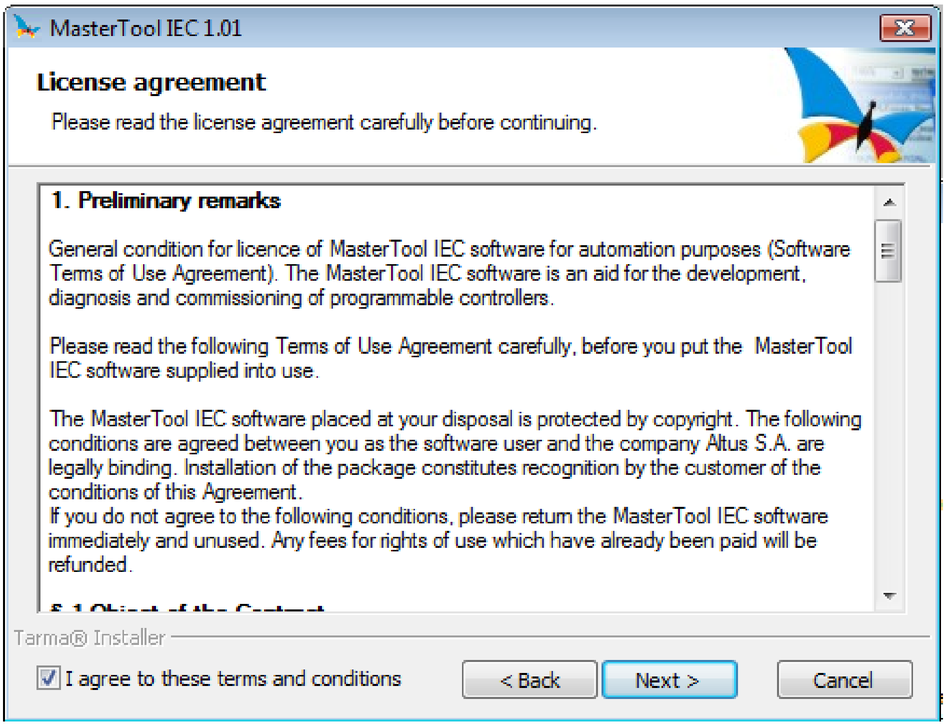


**Figure 4-12. License Page**

Click “Next” to continue. A page asking for the registration information will appear. After filling out the fields correctly, click “Next” to continue.

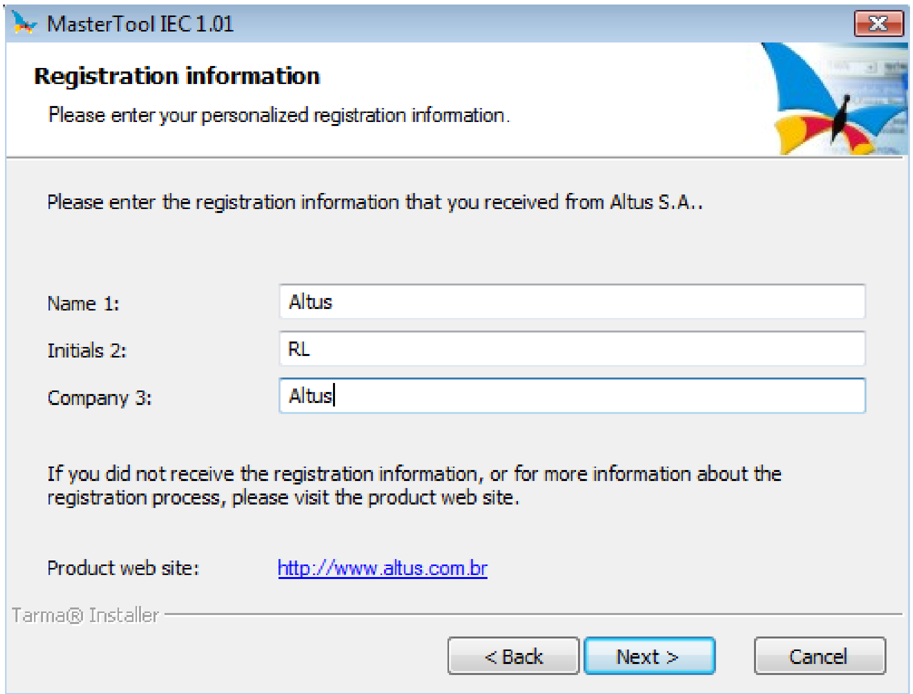


**Figure 4-13. Register Page**

On the next page the options to select the components to be installed, and the path to the software installation are offered. It is recommended the default path "C:\Program file\Program Altus\ MasterTool IEC". Click "Next" to continue.

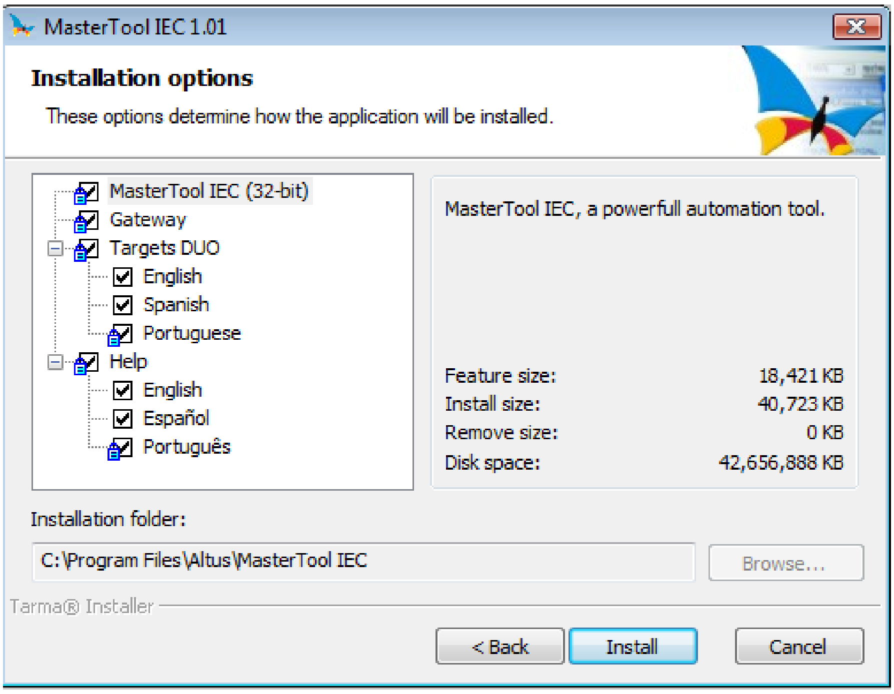


**Figure 4-14. Components Selection**

At this point the MasterTool IEC installation has started. Please wait while the requested files are installed on your computer. This may take several minutes depending on the computer's configuration.

After the installation is complete, the next page is shown, where the decision must be made whether the MasterTool IEC should initialize automatically after the completion of the installer. Click "Finish" to complete the installation procedure.

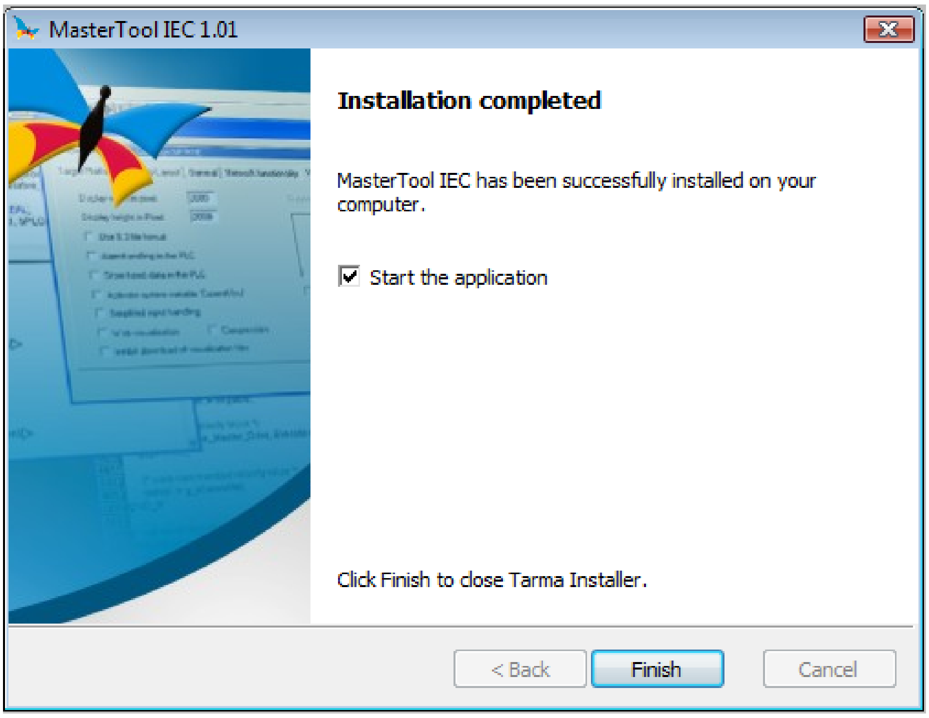


**Figure 4-15. Installation Complete**

MasterTool IEC is installed and ready to be used. To execute the MasterTool IEC click on the "MasterTool IEC" shortcut inside the group "Altus S.A" → "MasterTool IEC", that was created at the Start menu during the installation.

# 5. Initial Programming

## First Steps with MasterTool IEC and DU350/DU351

### MasterTool IEC Startup

It is recommended that the project is started from the project template. To create a new project from the template, click on the File menu, and then click "New from template ..." Select the project Template_DU350_DU351 .pro and then click "Open".

### Tasks and POUs Concepts

The Template_DU350_DU351.pro template has a set of configured tasks as shown in the following table:

| Task name | POU called by task | Activation interval |
|---|---|---|
| MAIN | PLC_PRG(); | 1 ms |
| NAVIGATION | SCREEN_CONTROL(); | 20 ms |
| VISUALIZATION | MAINTARGETVISU_PAINT_CODE | 500 ms |
| KEYBOARD_USAGE | MAINTARGETVISU_INPUT_CODE | 20 ms |

**Table 5-1. Template POUs Times**

The POU "SCREEN_CONTROL" is already included in the Template_DU350_DU351.pro. The MAINTARGETVISU_PAINT_CODE, and MAINTARGETVISU_INPUT_CODE POUs are already defined. To compile the project without errors, the PLC_PRG POU must be added, because it is called by the MAIN task. The PLC_PRG POU is called every 1 ms (if when a period of 1ms of the MAIN task is complete, another POU is running, the PLC_PRG POU will be executed only after the end of the actual POU). For more information about Configuring tasks and POUs check the chapter "Configuration" - "POU calls" or use the MasterTool IEC Programming Manual.

### Creating the PLC_PRG POU

In order to add the "PLC_PRG" POU, follow these steps: click on the "POUs" tab, then right-click on the word "POUs" (inside the "POUs" tab), select the option "Add object...", select the option "Program" in the field "POU type" , select the language in the "Language POU", type "PLC_PRG " in the field "Name of the new POU", and then click "OK" to confirm the adding of "PLC_PRG POU".

Each POU must have at least a logic to be compiled correctly; if the POU uses ST language, simply write ";" in the first command line.

### Compiling

In order to perform the full compilation of the project, click on the "Project" menu, and then on "Compile All".

**WARNING:**
**Before performing a compilation, or if any problem occurs during the compilation, it is recommended to click on the "Project" menu→ "Clean All" to remove any previous compilation.**

## Download

To download the Project, the communication interface should be properly configured. To configure the communication interface follow these steps: click on the "Communication" menu and select the option "Communication parameters ...". A new connection must be created, and to do that, press the button "New". The page in Figure 5.1 is shown so that you may set the connection name. By clicking OK, the connection is created. The connection parameters must be configured as shown in Figure 5 2. The communication port on the computer to be used must be configured in order to allow the connection of the serial which is connected to the PLC. To edit the parameters it is necessary to double click on the value, and then change this value by using the Up and Down keys on the keyboard.

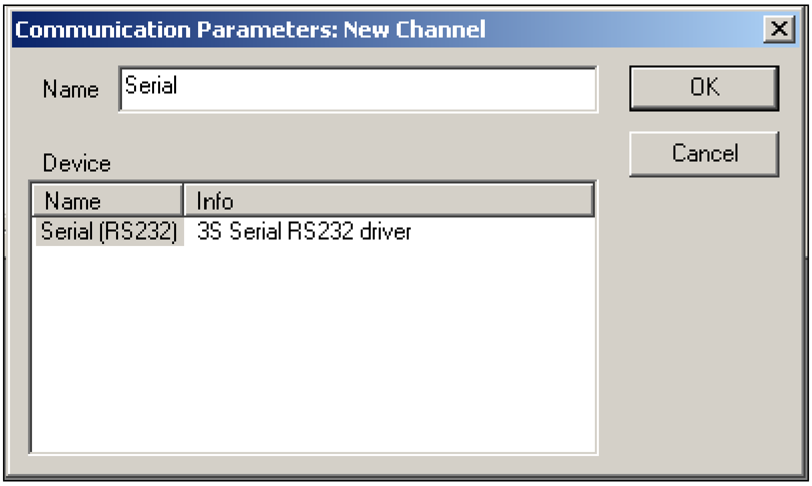


**Figure 5-1. New Connection**

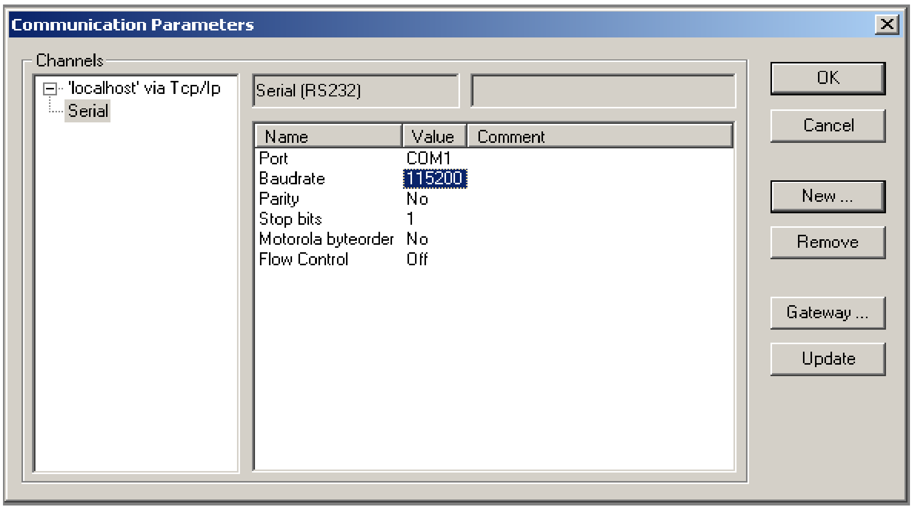


**Figure 5-2. Communication Parameters**

**WARNING:**
**The timeout for download must be of 8000 ms at the maximum. The default value is 2000 ms. It is not recommended to change this value without previous analysis. To configure this item, click on Project →Options..., select Desktop and configure the communication timeout for download [ms].**

After configuring the COM port, click on the "Communication" menu and select "Login". Thus, the Project will be sent to the controller. To start up the execution, click on the "Communication" menu and select "Run".

**WARNING:**
**During the download, the PLC takes its physical outputs to safety mode, as described in chapter “Configuration” – “Safe State”.**

## Analyzing the Template

DU350 and DU351 controllers must go through some basic configurations, which are necessary for their proper operation. These configurations are already implemented in Template_DU350_DU351.pro. These are:

### Configuring the Display

To ensure correct display visualization, the backlight time and contrast of the PLC HMI screen must be configured. This configuration is performed by both the CONTRAST and BACKLIGHT operands, or by the special pages (see “Configuration” –“Special Pages”).

### Page Browsing

In order to browse through the project pages by using the up and down directional arrows in DU350 and DU351 controllers, the NAVIGATION function block (instantiated as ScreenControl) should be included. This function block must be “fed” by a list with the name of all the pages comprised in the project (or the screens through which you want to navigate by using the keys). This list must have a maximum of 10 pages, and the name of each one must be less than 30 characters long. Another feature of the use of this function block is that by pressing the D350 and DU351 controller MAIN key, the “MAIN” page is enabled (the “MAIN” page must be set at position 0 in the list).

In the template case, only one page is added to the list, because there is only one page in the project. This implementation is performed in the following code lines.

```
(* Add MAIN page as main page *)

VisuList[0] := 'MAIN';
```

Obs: To add a new page on the list, add the following line on the code:

```
VisuList[x] := 'PAGE_NAME'; (* Line for adding a new page on the list *)
```

Where “x” is the number of the page and “PAGE_NAME” is the name of the respective page added to the list (the name of the pages must be written in upper case letters).

```
(* Enables the browsing automatic control only by the keyboard *)

ScreenControl(ENABLE_UPDW := FALSE, VISU_LIST := VisuList, UP := FALSE , DOWN := FALSE);
```

### Tasks Configuration

DU350 and DU351 controllers use Tasks (with intervals, activation type and priority defined) to call the POU routines . The provided template presents four tasks previously configured in the “Resources” tab, in sub item “Task Configuration”. These are:

**MAIN** – This task calls the “PLC_PRG POU” created by the user with an activation range of 1ms.

**NAVIGATION** –This task is used to call the “SCREEN_CONTROL(); ” which is responsible for page configurations, and for the Navigation function block. It is recommended to use it as cyclical-type with 200ms interval.

**VISUALIZATION** –This task is required for display use. It is recommended to use it as cyclical-type with a 500ms interval function. The called function is internal, and named *MAINTARGETVISU_PAINT_CODE*.

**KEYBOARD_USAGE** – This task is required for keyboard use (using “Keyboard functions ...” from the “Extras” menu. This option is available during a visualization edit). It is recommended to use it as cyclical-type with a range of 20 ms. The called function is internal and named *MAINTARGETVISU_INPUT_CODE*. In case of using “Keyboard functions ...” and there is the need to get faster response from the keys, the range of activation of this task may be reduced to obtain the desired result. However, if the application has a long-cycle task it is recommended to call the *MAINTARGETVISU_INPUT_CODE* POU created by a time-interruption driven POU.

The following figure exhibits tasks configuration:

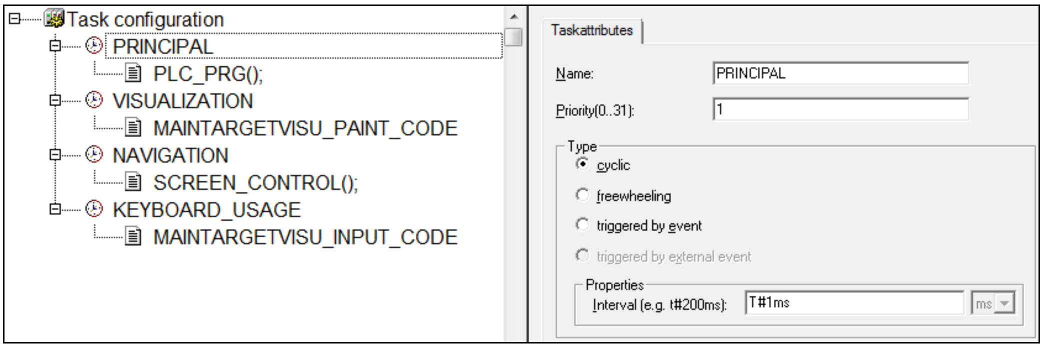


**Figure 5-3. Tasks Configuration**

## Project Conversion

DU350 and DU351 Modules have some features which were implemented starting from version 1.10 of the software. For this reason, the MasterTool IEC installer, as from version 1.03 has the configuration of devices for this version "DU350 and DU351 v1.10 ...". There is also a model project using this device "Model_DU350_DU351_v110".

When there are projects using a device of versions previous to the 1.10, and if you want to convert them to the device of the 1.10 or higher versions, it must be changed in "Device Settings"on the "Resources" tab. As shown in Figure 5 4, select the "DU350 and DU351 v1.10 ..." device of the open project and the new settings will be loaded.

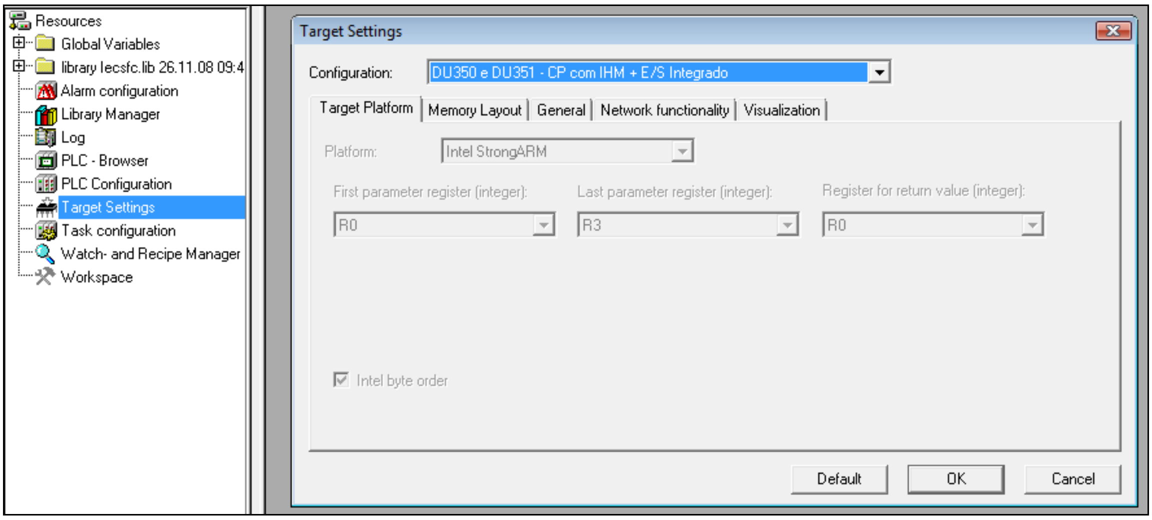


**Figure 5-4. Changing the Device Configuration**

Changing the device will not change the CP settings. This is to avoid the loss of the settings. In case the settings must be updated to match the new device settings, the menu Extras → Default setting must be selected

## Simulation Mode

The MasterTool IEC has an impotant recourse of simulation that allows the users to test their applications without to use any equipament, it gives more agility at program development. However, there are some specific features with hardware of DU350/DU351 dependency cannot be simulated.

The following features are not able in simulation mode:

- Fast Inputs and Outputs
- Serial Ports
- Keyboard
- Time Interruption POUs
- Real Time Clock

It is also unable the following functions:

- Motion Functions
- Generic Communication Functions
- Screen Navigation Functions
- Real Time Clock Functions

# 6. Maintenance

## Diagnostics

DU350 and DU351 controllers contain a series of special operands that provide diagnostic data from several devices in the controller.

For quick access to information, status and diagnostics of inputs and outputs, special pages may be consulted, as described in the "Configuration"-"Special Screens" chapter.

### General Diagnostics

DU350 and DU351 controllers have some general diagnostics which are available through special registers. These special operands are previously mapped into a specific area of memory. Thus, they may be employed as a global variable.

**BROWNOUT** = Indicates a brownout (indicates that the PLC was rebooted due to a power supply voltage failure).

**FALHA_RTC** = Indicates loss of the clock information.

**FALHA_RETENTIVAS** = Indicates a writing error in the retentive variables.

**WATCHDOG** = Indicates a watchdog in the previous execution.

**TAM_APLICATIVO** = Size of the user application (number of used bytes).

**SOFT_H** = Executive version (number before the point).

**SOFT_L** = Executive version (number after the point).

**MODELO =** Controller model (350 for DU350 and 351 for DU351).

The BROWNOUT, FALHA_RTC, WATCHDOG, TAM_APLICATIVO, SOFT_H and SOFT_L operands may be changed through the application and may be used as handled diagnostic signalization.

The names of the operands and how to access them are better described in the special operands list, located in the section "Diagnostics" - "Reserved Operands List".

### Troubleshooting

The next table contains the symptoms of some problems as well as their possible causes and solutions. If the problem persists, contact Altus Technical Support.

| Symptom | Possible Cause | Solutions |
|---|---|---|
| Do not turn on | Power supply failure or incorrect. | De-energize and energize the PLC.<br>Check the power supply source operation.<br>Check if the power supply voltage gets to the PLC terminal block with the correct polarization.<br>Check if the power supply voltage gets to the PLC.<br>Check if the application screen has proper contrast and backlight time. |
| Do not communicate | Connection or configuration | Check the communication cables connection.<br>Check the COM port settings in MasterTool IEC. |
| Do not switch screen | Application in Stop mode. Only one screen set up or keyboard not set up. | Check if the PLC is in Run mode.<br>Check if the application has more than one screen<br>Check if the keyboard is properly configured for the screen commutation. |

| | | |
|---|---|---|
| Do not respond to the keyboard | Keyboard not setup. | Check if the visualizations are with the function keys properly set up.<br>Check if the application uses the isKeyPressed() function, and the KeyPressed() properly, or if there is a conflict in the keyboard usage.<br>Check if the MAINTARGETVISU_INPUT_CODE function is called periodically by some task in the application.<br>Check if the shortcut keys are responding. |
| Do not display the user visualization | Application with improper contrast value or with no visualizations. | Check if the application has a proper value for the display contrast.<br>Check if there are configured visualizations for the current application.<br>Check if the MAINTARGETVISU_PAINT_CODE function is called periodically by some task in the application. |

**Table 6-1. Troubleshooting Table**

## Fast Inputs Diagnostics

DU350 and DU351 controllers have the following special reserved operands for fast input diagnostics, where “ x” is the fast input number which can vary from 0 to 3:

| | |
|---|---|
| **CNTx_OVERFLOW** | TRUE in case of overflow in counter x |
| **CNTx_UNDERFLOW** | TRUE in case of underflow in counter x |
| **CNTx_DIR** | Counter x direction<br>(FALSE- progressive / TRUE- regressive) |
| **CNTx_MAX_CMP0** | TRUE in case counter x is greater than CNTx_CMP0 |
| **CNTx_MAX_CMP1** | TRUE in case counter x is greater than CNTx_CMP1 |
| **CNTx_EQ_CMP0** | TRUE in case counter x is equal to CNTx_CMP0 |
| **CNTx_EQ_CMP1** | TRUE in case counter x is equal to CNTx_CMP1 |
| **CNTx_ZERO** | TRUE in case counter x is equal to ZERO |

**Table 6-2. Fast Inputs Diagnostics**

The special operands of fast input diagnostics are previously mapped into a specific area of memory. Thus, they may be employed as a global variable. The names of the operands are better described in the special operands list in the section “Diagnostics” - “Reserved Operands List”.

## Analog Outputs Diagnostics

DU350 and DU351 controllers have the following special reserved operands for analog output diagnostics:

| | |
|---|---|
| **AO0_DG** | Short-circuit diagnostics (voltage mode) or open loop (current mode) in channel 0 of analog output |
| **AO1_DG** | Short-circuit diagnostics (voltage mode) or open loop (current mode) in channel 1 of analog output |

**Table 6-3. Analog Outputs Diagnostics**

The special operands of analog outputs diagnostics are previously mapped into a specific region of memory. Thus, they can be used as a global variable. The names of the operands are better described in the special operands list in the section “Diagnostics” - “Reserved Operands List”.

## Analog Inputs Diagnostics

The current analog inputs may be configured to be used in the range of 4 mA to 20 mA or 0 mA-20 mA. When used as current analog input in the range of 4 mA - 20 mA, the inputs are diagnosed as open current input diagnostic (AIx_DG).

This diagnostics is enabled when the channel current input is less than or equal to 3.8 mA.

| | |
|---|---|
| **AI0_DG** | Channel 0 diagnostics of analog input in current 4-20 mA mode with open loop |

| | |
|---|---|
| AI1_DG | Channel 1 diagnostics of analog input in current 4-20 mA mode with open loop |
| AI2_DG | Channel 2 diagnostics of analog input in current 4-20 mA mode with open loop |
| AI3_DG | Channel 3 diagnostics of analog input in current 4-20 mA mode with open loop |

**Table 6-4. Analog Inputs Diagnostics**

The special operands of analog inputs diagnostics are previously mapped into a specific area of memory. Thus, they may be employed as a global variable. The names of the operands are better described in the special operands list in the section "Diagnostics" - "Reserved Operands List".

## Fast Outputs Diagnostics

### *PTO*

DU350 and DU351 controllers have the following special reserved operands for PTO diagnostics, where " x" is the fast output number which can vary from 0 to 1:

| Diagnostic | Description |
|---|---|
| Fx_PTO_DG Diagnostic Operand | |
| Fx_PTO_ON | Bit 0 - PTO output in progress |
| Fx_PTO_ACE | Bit 1 - PTO output in acceleration |
| Fx_PTO_REG | Bit 2 - PTO output in steady state |
| Fx_PTO_DES | Bit 3 - PTO output in deceleration |
| Fx_PTO_PRM | Bit 4 - PTO output not parameterized |
| Fx_PTO_ERR | Bit 5 - PTO output with parameterization error |
| Fx_PTO_CNT_DG Diagnostic Operand | |
| Fx_PTO_CNT_MAX_CMP0 | Bit 0 - PTO pulse counter greater than the comparator 0 |
| Fx_PTO_CNT_MAX_CMP1 | Bit 1 - PTO pulse counter greater than the comparator 1 |

**Table 6-5. Fast Outputs Diagnostics**

The Fx_PTO_ON, Fx_PTO_ACE, Fx_PTO_REG, Fx_PTO_DES, Fx_PTO_PRM and Fx_PTO_ERR diagnostic operands are of BOOL type and together make up the Fx_PTO_DG, which is of BYTE type

The Fx_PTO_CNT_MAX_CMP0 and Fx_PTO_CNT_MAX_CMP1 diagnostic operands are of BOOL type and together make up the Fx_PTO_CNT_DG, which is of BYTE type

The special operands of PTO diagnostics are previously mapped into a specific area of memory. Thus, they may be employed as a global variable. The names of the operands are better described in the special operands list in the section "Diagnostics" - "Reserved Operands List".

### *VFO/PWM*

DU350 and DU351 controllers have the following special reserved operands for VFO/PWM diagnostics, where " x" is the fast output number which can vary from 0 to 1:

| Diagnostic | Description |
|---|---|
| Fx_VFO_DG Diagnostic Operand | |
| Fx_VFO_ON | Bit 0 - Output in progress |
| Fx_VFO_PRM | Bit 1 - VFO/PWM output not parameterized |
| Fx_VFO_ERR | Bit 2 - VFO/PWM output with parameterization error |

**Table 6-6. VFO/PWM Diagnostics**

The Fx_VFO_ON,_Fx_VFO_PRM and Fx_VFO_ERR diagnostic operands are of BOOL type and together make up the Fx_VFO_DG, which is of BYTE type.

The special operands of VFO/PWM diagnostics are previously mapped into a specific area of memory. Thus, they may be employed as a global variable. The names of the operands are better described in the special operands list in the section "Diagnostics" - "Reserved Operands List".

### MODBUS Diagnostics

Each MODBUS relationship has an operand error counter and one operand which informs the communication status of the previous cycle.

If the communication port is configured as MODBUS Slave, the COMx_DE operand indicates the status of the last MODBUS communication performed at the x communication port. The COMx_CE operand indicates the number of errors that occurred at the x communication port.

If the communication port is configured as MODBUS Master, the COMx_Ry operand indicates the status of the last MODBUS communication of the y relation of x communication port. The COMx_Cy operand indicates the number of errors that occurred in the relation y of the communication port x.

By assigning TRUE for the CLR_ALL_COM operand, 0 will be assigned to all of the error counters (COMx_CE and COMx_Cy).

Table 6-7 describes the reserved operands related to the MODBUS Protocol diagnostics, where "x" is the COM port number and "y" is the relation number.

| | |
|---|---|
| **CLR_ALL_COM** | Clear all error counter of MODBUS communication |
| **COMx_DE** | Diagnostics in slave mode |
| **COMx_CE** | Error counter in slave mode |
| **COMx_Cy** | Error counter of relation y |
| **COMx_Ry** | Relation y diagnostics |

**Table 6-7. MODBUS Diagnostics**

Table 6-8 shows the error code that may be assumed by the communication status operands:

| Code | Description |
|---|---|
| **Master** | |
| **1** | Requested command does not accept broadcast |
| **2** | Communication failure |
| **3** | MODBUS address not found in the table |
| **4** | Sent command not implemented |
| **5** | Response receiving of another address |
| **6** | CRC Error Master response |
| **Slave** | |
| **7** | Received command not implementated |
| **9** | CRC error in the received packet |
| **10** | Address for another slave or broadcast |
| **11** | Invalid limit of the command address |
| **12** | MODBUS address not found in the table |
| **13** | Invalid host address |
| **General Communication Diagnostics (Master / Slave)** | |
| **0** | Indicates that the communication is ok |
| **14** | Reception error |
| **15** | Parity error |
| **16** | Framing error (baudrate, stopbits,...) |
| **17** | CTS timeout |

**Table 6-8. MODBUS Error Codes**

## Reserved Operands List

DU350 and DU351 controllers hold a special operands list employed for CPU diagnostics and configuration. The special operands are previously mapped into a specific area of memory. Thus, they may be used as a global variable.

To access the special operands on MasterTool IEC, double-click "DU35x.lib library..." located at the

"Resources" tab. Then, in order to show the special operands list, double-click "Global Variables..." In this list are described all the special operands. The following table presents these operands with their memory addresses and functionalities.

| | Name | Address | Type | Description |
|---|---|---|---|---|
| **MODEM** | | | | |
| **1** | **RTS** | %MB6373 | BYTE | RTS pin of COM0 port |
| **2** | **CTS** | %MB6374 | BYTE | CTS pin of COM0 port |
| **3** | **DCD** | %MB6375 | BYTE | DCD pin of Com0 port |
| **4** | **DTR** | %MB6376 | BYTE | DTR pin of Com0 port |
| **5** | **DSR** | %MB6377 | BYTE | DSR pin of Com0 port |
| **COM1** | | | | |
| **6** | **CLR_ALL_COM** | %MB6378 | BOOL | Clear COMs diagnostics |
| **7** | **COM1_DE** | %MB6379 | BYTE | Diagnostics in slave mode |
| **8** | **COM1_CE** | %MB6380 | BYTE | Counter in slave mode |
| **9** | **COM1_DR** | %MW3191 | WORD | Disable relations |
| **10** | **COM1_C0** | %MB6384 | BYTE | Counter of relation 0 |
| **11** | **COM1_R0** | %MB6385 | BYTE | Diagnostics of relation 0 |
| **12** | **COM1_C1** | %MB6386 | BYTE | Counter of relation 1 |
| **13** | **COM1_R1** | %MB6387 | BYTE | Diagnostics of relation 1 |
| **14** | **COM1_C2** | %MB6388 | BYTE | Counter of relation 2 |
| **15** | **COM1_R2** | %MB6389 | BYTE | Diagnostics of relation 2 |
| **16** | **COM1_C3** | %MB6390 | BYTE | Counter of relation 3 |
| **17** | **COM1_R3** | %MB6391 | BYTE | Diagnostics of relation 3 |
| **18** | **COM1_C4** | %MB6392 | BYTE | Counter of relation 4 |
| **19** | **COM1_R4** | %MB6393 | BYTE | Diagnostics of relation 4 |
| **20** | **COM1_C5** | %MB6394 | BYTE | Counter of relation 5 |
| **21** | **COM1_R5** | %MB6395 | BYTE | Diagnostics of relation 5 |
| **22** | **COM1_C6** | %MB6396 | BYTE | Counter of relation 6 |
| **23** | **COM1_R6** | %MB6397 | BYTE | Diagnostics of relation 6 |
| **24** | **COM1_C7** | %MB6398 | BYTE | Counter of relation 7 |
| **25** | **COM1_R7** | %MB6399 | BYTE | Diagnostics of relation 7 |
| **26** | **COM1_C8** | %MB6400 | BYTE | Counter of relation 8 |
| **27** | **COM1_R8** | %MB6401 | BYTE | Diagnostics of relation 8 |
| **28** | **COM1_C9** | %MB6402 | BYTE | Counter of relation 9 |
| **29** | **COM1_R9** | %MB6403 | BYTE | Diagnostics of relation 9 |
| **30** | **COM1_C10** | %MB6404 | BYTE | Counter of relation 10 |
| **31** | **COM1_R10** | %MB6405 | BYTE | Diagnostics of relation 10 |
| **32** | **COM1_C11** | %MB6406 | BYTE | Counter of relation 11 |
| **33** | **COM1_R11** | %MB6407 | BYTE | Diagnostics of relation 11 |
| **34** | **COM1_C12** | %MB6408 | BYTE | Counter of relation 12 |
| **35** | **COM1_R12** | %MB6409 | BYTE | Diagnostics of relation 12 |
| **36** | **COM1_C13** | %MB6410 | BYTE | Counter of relation 13 |
| **37** | **COM1_R13** | %MB6411 | BYTE | Diagnostics of relation 13 |
| **38** | **COM1_C14** | %MB6412 | BYTE | Counter of relation 14 |
| **39** | **COM1_R14** | %MB6413 | BYTE | Diagnostics of relation 14 |
| **40** | **COM1_C15** | %MB6414 | BYTE | Counter of relation 15 |
| **41** | **COM1_R15** | %MB6415 | BYTE | Diagnostics of relation 15 |
| **COM2** | | | | |
| **42** | **COM2_DE** | %MB6419 | BYTE | Diagnostics in slave mode |
| **43** | **COM2_CE** | %MB6420 | BYTE | Counter in slave mode |
| **44** | **COM2_DR** | %MW3211 | WORD | Disable relations |
| **45** | **COM2_C0** | %MB6424 | BYTE | Counter of relation 0 |
| **46** | **COM2_R0** | %MB6425 | BYTE | Diagnostics of relation 0 |
| **47** | **COM2_C1** | %MB6426 | BYTE | Counter of relation 1 |
| **48** | **COM2_R1** | %MB6427 | BYTE | Diagnostics of relation 1 |

| | | | | |
|---|---|---|---|---|
| **49** | **COM2_C2** | %MB6428 | BYTE | Counter of relation 2 |
| **50** | **COM2_R2** | %MB6429 | BYTE | Diagnostics of relation 2 |
| **51** | **COM2_C3** | %MB6430 | BYTE | Counter of relation 3 |
| **52** | **COM2_R3** | %MB6431 | BYTE | Diagnostics of relation 3 |
| **53** | **COM2_C4** | %MB6432 | BYTE | Counter of relation 4 |
| **54** | **COM2_R4** | %MB6433 | BYTE | Diagnostics of relation 4 |
| **55** | **COM2_C5** | %MB6434 | BYTE | Counter of relation 5 |
| **56** | **COM2_R5** | %MB6435 | BYTE | Diagnostics of relation 5 |
| **57** | **COM2_C6** | %MB6436 | BYTE | Counter of relation 6 |
| **58** | **COM2_R6** | %MB6437 | BYTE | Diagnostics of relation 6 |
| **59** | **COM2_C7** | %MB6438 | BYTE | Counter of relation 7 |
| **60** | **COM2_R7** | %MB6439 | BYTE | Diagnostics of relation 7 |
| **61** | **COM2_C8** | %MB6440 | BYTE | Counter of relation 8 |
| **62** | **COM2_R8** | %MB6441 | BYTE | Diagnostics of relation 8 |
| **63** | **COM2_C9** | %MB6442 | BYTE | Counter of relation 9 |
| **64** | **COM2_R9** | %MB6443 | BYTE | Diagnostics of relation 9 |
| **65** | **COM2_C10** | %MB6444 | BYTE | Counter of relation 10 |
| **66** | **COM2_R10** | %MB6445 | BYTE | Diagnostics of relation 10 |
| **67** | **COM2_C11** | %MB6446 | BYTE | Counter of relation 11 |
| **68** | **COM2_R11** | %MB6447 | BYTE | Diagnostics of relation 11 |
| **69** | **COM2_C12** | %MB6448 | BYTE | Counter of relation 12 |
| **70** | **COM2_R12** | %MB6449 | BYTE | Diagnostics of relation 12 |
| **71** | **COM2_C13** | %MB6450 | BYTE | Counter of relation 13 |
| **72** | **COM2_R13** | %MB6451 | BYTE | Diagnostics of relation 13 |
| **73** | **COM2_C14** | %MB6452 | BYTE | Counter of relation14 |
| **74** | **COM2_R14** | %MB6453 | BYTE | Diagnostics of relation 14 |
| **75** | **COM2_C15** | %MB6454 | BYTE | Counter of relation 15 |
| **76** | **COM2_R15** | %MB6455 | BYTE | Diagnostics of relation 15 |
| **General Configurations** | | | | |
| **77** | **STATUS_CP** | %MB6460 | BYTE | Brownout/RTC/Retain |
| **78** | **TAM_APLICATIVO** | %MD1616 | DWORD | User application size |
| **79** | **CONTRASTE** | %MB6468 | BYTE | Contrast percentage |
| **80** | **BACKLIGHT** | %MB6469 | BYTE | Backlight configuration |
| **81** | **SOFT_H** | %MB6470 | BYTE | Number before the dot |
| **82** | **SOFT_L** | %MB6471 | BYTE | Number after the dot |
| **83** | **MODEL** | %MW3231 | WORD | PLC model |
| **PLC Status** | | | | |
| **84** | **BROWNOUT** | %MX3230.0 | BOOL | Indicates the occurrence of a brownout in the PLC |
| **85** | **WATCHDOG** | %MX3230.1 | BOOL | Indicates the occurrence of a watchdog in the last execution |
| **86** | **FALHA_RETENTIVAS** | %MX3230.2 | BOOL | Indicates an error in the retain variables |
| **87** | **FALHA_RTC** | %MX3230.3 | BOOL | Indicates the loss of information from the RTC |
| **Unidirectional or Bidirectional - Counter 0** | | | | |
| **88** | **CNT_B0_EXT_EVENT_CNT** | %MB6472 | BYTE | Rising edge counter in the control pin of block 0 counter |
| **89** | **CNT0** | %MD1619 | DWORD | Counting value |
| **90** | **CNT0_PRESET** | %MD1620 | DWORD | Load PRESET value |
| **91** | **CNT0_HOLD** | %MD1621 | DWORD | Sampling value |
| **92** | **CNT0_CMP0** | %MD1622 | DWORD | Comparator 0 value |
| **93** | **CNT0_CMP1** | %MD1623 | DWORD | Comparator 1 value |
| **94** | **CNT0_CMD** | %MB6496 | BYTE | Commands |
| **95** | **CNT0_DG** | %MB6497 | BYTE | Diagnostics |
| **Counter 0 Commands** | | | | |
| **96** | **CNT0_CLR** | %MX3248.0 | BOOL | Clear counting register |
| **97** | **CNT0_STOP** | %MX3248.1 | BOOL | Disable counting |
| **98** | **CNT0_LOAD** | %MX3248.2 | BOOL | Load PRESET |

| | | | | |
|---|---|---|---|---|
| **99** | **CNT0_AMG** | %MX3248.3 | BOOL | Counting sampling (HOLD) |
| **100** | **CNT0_OVER** | %MX3248.4 | BOOL | Clear overflow and underflow bit |
| **Counter 0 Diagnostics** | | | | |
| **101** | **CNT0_OVERFLOW** | %MX3248.8 | BOOL | Counting overflow |
| **102** | **CNT0_UNDERFLOW** | %MX3248.9 | BOOL | Counting underflow |
| **103** | **CNT0_DIR** | %MX3248.10 | BOOL | Counting direction (0- progressive / 1- regressive) |
| **104** | **CNT0_MAX_CMP0** | %MX3248.11 | BOOL | Counter greater than CNT0_CMP0 |
| **105** | **CNT0_MAX_CMP1** | %MX3248.12 | BOOL | Counter greater than CNT0_CMP1 |
| **106** | **CNT0_EQ_CMP0** | %MX3248.13 | BOOL | Counter equal to CNT0_CMP0 |
| **107** | **CNT0_EQ_CMP1** | %MX3248.14 | BOOL | Counter equal to CNT0_CMP1 |
| **108** | **CNT0_ZERO** | %MX3248.15 | BOOL | Counter equal to ZERO |
| **Unidirectional – Counter 1** | | | | |
| **109** | **CNT1** | %MD1626 | DWORD | Counting value |
| **110** | **CNT1_PRESET** | %MD1627 | DWORD | Load PRESET value |
| **111** | **CNT1_HOLD** | %MD1628 | DWORD | Sampling value |
| **112** | **CNT1_CMP0** | %MD1629 | DWORD | Comparator 0 value |
| **113** | **CNT1_CMP1** | %MD1630 | DWORD | Comparator 1 value |
| **114** | **CNT1_CMD** | %MB6524 | BYTE | Commands |
| **115** | **CNT1_DG** | %MB6525 | BYTE | Diagnostics |
| **Counter 1 Commands** | | | | |
| **116** | **CNT1_CLR** | %MX3262.0 | BOOL | Clear counting register |
| **117** | **CNT1_STOP** | %MX3262.1 | BOOL | Disable counting |
| **118** | **CNT1_LOAD** | %MX3262.2 | BOOL | Load PRESET |
| **119** | **CNT1_AMG** | %MX3262.3 | BOOL | Counting sampling (HOLD) |
| **120** | **CNT1_OVER** | %MX3262.4 | BOOL | Clear overflow and underflow bits |
| **Counter 1 Diagnostics** | | | | |
| **121** | **CNT1_OVERFLOW** | %MX3262.8 | BOOL | Counting overflow |
| **122** | **CNT1_UNDERFLOW** | %MX3262.9 | BOOL | Counting underflow |
| **123** | **CNT1_DIR** | %MX3262.10 | BOOL | Counting direction (0- progressive / 1- regressive) |
| **124** | **CNT1_MAX_CMP0** | %MX3262.11 | BOOL | Counter greater than CNT1_CMP0 |
| **125** | **CNT1_MAX_CMP1** | %MX3262.12 | BOOL | Counter greater than CNT1_CMP1 |
| **126** | **CNT1_EQ_CMP0** | %MX3262.13 | BOOL | Counter equal to CNT1_CMP0 |
| **127** | **CNT1_EQ_CMP1** | %MX3262.14 | BOOL | Counter equal to CNT1_CMP1 |
| **128** | **CNT1_ZERO** | %MX3262.15 | BOOL | Counter equal to ZERO |
| **Unidirectional or Bidirectional - Counter 2** | | | | |
| **129** | **CNT_B1_EXT_EVENT _CNT** | %MB6474 | BYTE | Rising edge counter in the control pin of block 1 counter |
| **130** | **CNT2** | %MD1633 | DWORD | Counting value |
| **131** | **CNT2_PRESET** | %MD1634 | DWORD | Load PRESET value |
| **132** | **CNT2_HOLD** | %MD1635 | DWORD | Sampling value |
| **133** | **CNT2_CMP0** | %MD1636 | DWORD | Comparator 0 value |
| **134** | **CNT2_CMP1** | %MD1637 | DWORD | Comparator 1 value |
| **135** | **CNT2_CMD** | %MB6552 | BYTE | Commands |
| **136** | **CNT2_DG** | %MB6553 | BYTE | Diagnostics |
| **Counter 2 Commands** | | | | |
| **137** | **CNT2_CLR** | %MX3276.0 | BOOL | Clear counting register |
| **138** | **CNT2_STOP** | %MX3276.1 | BOOL | Disable counting |
| **139** | **CNT2_LOAD** | %MX3276.2 | BOOL | Load PRESET |
| **140** | **CNT2_AMG** | %MX3276.3 | BOOL | Counting sampling (HOLD) |
| **141** | **CNT2_OVER** | %MX3276.4 | BOOL | Clear overflow and underflow bits |
| **Counter 2 Diagnostics** | | | | |
| **142** | **CNT2_OVERFLOW** | %MX3276.8 | BOOL | Counting overflow |
| **143** | **CNT2_UNDERFLOW** | %MX3276.9 | BOOL | Counting underflow |
| **144** | **CNT2_DIR** | %MX3276.10 | BOOL | Counting direction |
| **145** | **CNT2_MAX_CMP0** | %MX3276.11 | BOOL | Counter greater than CNT2_CMP0 |

| | | | | |
|---|---|---|---|---|
| 146 | **CNT2_MAX_CMP1** | %MX3276.12 | BOOL | Counter greater than CNT2_CMP1 |
| 147 | **CNT2_EQ_CMP0** | %MX3276.13 | BOOL | Counter equal to CNT2_CMP0 |
| 148 | **CNT2_EQ_CMP1** | %MX3276.14 | BOOL | Counter equal to CNT2_CMP1 |
| 149 | **CNT2_ZERO** | %MX3276.15 | BOOL | Counter equal to ZERO |
| **Unidirectional – Counter 3** | | | | |
| 150 | **CNT3** | %MD1640 | DWORD | Counting value |
| 151 | **CNT3_PRESET** | %MD1641 | DWORD | Load PRESET value |
| 152 | **CNT3_HOLD** | %MD1642 | DWORD | Sampling value |
| 153 | **CNT3_CMP0** | %MD1643 | DWORD | Comparator 0 value |
| 154 | **CNT3_CMP1** | %MD1644 | DWORD | Comparator 1 value |
| 155 | **CNT3_CMD** | %MB6580 | BYTE | Commands |
| 156 | **CNT3_DG** | %MB6581 | BYTE | Diagnostics |
| **Counter 3 Commands** | | | | |
| 157 | **CNT3_CLR** | %MX3290.0 | BOOL | Clear counting register |
| 158 | **CNT3_STOP** | %MX3290.1 | BOOL | Disable counting |
| 159 | **CNT3_LOAD** | %MX3290.2 | BOOL | Load PRESET |
| 160 | **CNT3_AMG** | %MX3290.3 | BOOL | Counting sampling (HOLD) |
| 161 | **CNT3_OVER** | %MX3290.4 | BOOL | Clear overflow and underflow bits |
| **Counter 3 Diagnostics** | | | | |
| 162 | **CNT3_OVERFLOW** | %MX3290.8 | BOOL | Counting overflow |
| 163 | **CNT3_UNDERFLOW** | %MX3290.9 | BOOL | Counting underflow |
| 164 | **CNT3_DIR** | %MX3290.10 | BOOL | Counting direction |
| 165 | **CNT3_MAX_CMP0** | %MX3290.11 | BOOL | Counter greater than CNT3_CMP0 |
| 166 | **CNT3_MAX_CMP1** | %MX3290.12 | BOOL | Counter greater than CNT3_CMP1 |
| 167 | **CNT3_EQ_CMP0** | %MX3290.13 | BOOL | Counter equal to CNT3_CMP0 |
| 168 | **CNT3_EQ_CMP1** | %MX3290.14 | BOOL | Counter equal to CNT3_CMP1 |
| 169 | **CNT3_ZERO** | %MX3290.15 | BOOL | Counter equal to ZERO |
| **Fast Output - F0** | | | | |
| 170 | **F0_FREQ** | %MD1647 | DWORD | Frequency value for PTO/VFO/PWM |
| 171 | **F0_PLS_TOT** | %MD1648 | DWORD | Total pulses value for PTO |
| 172 | **F0_PLS_RMP** | %MD1649 | DWORD | Ramp pulses value for PTO |
| 173 | **F0_DUTY** | %MB6600 | DWORD | Duty Cycle value for VFO/PWM |
| 174 | **F0_DUTY_HR** | %MD1646 | REAL | Duty Cycle high-resolution value for VFO/PWM |
| 175 | **F0_PTO_CNT_CMP0** | %MD1570 | DWORD | Comparator 0 of F0 fast output |
| 176 | **F0_PTO_CNT_CMP1** | %MD1571 | DWORD | Comparator 1 of F0 fast output |
| **Fast Output- F1** | | | | |
| 177 | **F1_FREQ** | %MD1652 | DWORD | Frequency value for PTO/VFO/PWM |
| 178 | **F1_PLS_TOT** | %MD1653 | DWORD | Total pulses value for PTO |
| 179 | **F1_PLS_RMP** | %MD1654 | DWORD | Ramp pulses value for PTO |
| 180 | **F1_DUTY** | %MB6620 | DWORD | Duty Cycle value for VFO/PWM |
| 181 | **F1_DUTY_HR** | %MD1651 | REAL | Duty Cycle high-resolution value for VFO/PWM |
| 182 | **F1_PTO_CNT_CMP0** | %MD1572 | DWORD | Comparator 0 of F1 fast output |
| 183 | **F1_PTO_CNT_CMP1** | %MD1573 | DWORD | Comparator 1 of F1 fast output |
| **Fast Output - Commands and Diagnostics** | | | | |
| 184 | **PTO_CMD** | %MB6628 | BYTE | Operand mounted by the commands for PTO of F0 and F1 outputs. The commands are described below. |
| 185 | **VFO_CMD** | %MB6629 | BYTE | Operand mounted by the commands for VFO/PWM of F0 and F1 outputs. The commands are described below. |
| 186 | **F0_PTO_DG** | %MB6630 | BYTE | Operand mounted by the diagnostics for PTO of F0 output. The diagnostics are described below. |
| 187 | **F1_PTO_DG** | %MB6631 | BYTE | Operand mounted by the diagnostics for PTO of F1 output. The diagnostics are described below. |
| 188 | **VFO_DG** | %MB6632 | BYTE | Operand mounted by the diagnostics for VFO/PWM of F0 and F1 outputs. The diagnostics are described below. |

| | | | | |
|---|---|---|---|---|
| **189** | **PTO_CNT_CMD** | %MB6276 | BYTE | Commands for counters of PTO at F0 and F1 fast outputs. |
| **190** | **F0_PTO_CNT_DG** | %MB6277 | BYTE | Operand mounted by the diagnostics for PTO pulse counters of F0 output. The diagnostics are described below. |
| **191** | **F1_PTO_CNT_DG** | %MB6278 | BYTE | Operand mounted by the diagnostics for PTO pulse counters of F1 output. The diagnostics are described below. |
| **192** | **F0_PTO_CNT_REL** | %MD1565 | DWORD | PTO relative pulse counter at F0 fast output. |
| **193** | **F0_PTO_CNT_ABS** | %MD1566 | DWORD | PTO absolute pulse counter at F0 fast output. |
| **194** | **F1_PTO_CNT_REL** | %MD1567 | DWORD | PTO relative pulse counter at F1 fast output. |
| **195** | **F1_PTO_CNT_ABS** | %MD1568 | DWORD | PTO absolute pulse counter of F1 fast output. |
| **PTO Commands - F0** | | | | |
| **196** | **F0_PTO_START** | %MX3314.0 | BOOL | Start the train of pulses (PTO) in the F0 output |
| **197** | **F0_PTO_STOP** | %MX3314.1 | BOOL | Stop the train of pulses (PTO) in the F0 output |
| **198** | **F0_PTO_SOFTSTOP** | %MX3314.4 | BOOL | Start a soft stop at the F0 output |
| **199** | **F0_PTO_CNT_DIR** | %MX3138.0 | BOOL | Define the counting direction of the pulse counter of F0 output. |
| **200** | **F0_PTO_CNT_CLR** | %MX3138.1 | BOOL | Reset the absolute counter of F0 output. |
| **PTO Commands - F1** | | | | |
| **201** | **F1_PTO_START** | %MX3314.2 | BOOL | Start the train of pulses (PTO) in the F1 output |
| **202** | **F1_PTO_STOP** | %MX3314.3 | BOOL | Stop the train of pulses (PTO) in the F0 output |
| **203** | **F1_PTO_SOFTSTOP** | %MX3314.5 | BOOL | Start a soft stop at the F1 output |
| **204** | **F1_PTO_CNT_DIR** | %MX3138.2 | BOOL | Define the counting direction of the pulse counter of F1 output |
| **205** | **F1_PTO_CNT_CLR** | %MX3138.3 | BOOL | Reset the absolute counter of F1 output. |
| **VFO/PWM Commands - F0** | | | | |
| **206** | **F0_VFO** | %MX3314.8 | BOOL | 0 -> Disable VFO/PWM in the F0 output (stopped output) 1 -> Enable VFO/PWM in the F0 output |
| **207** | **F0_VFO_DUTY_SRC** | %MX3314.10 | BOOL | Select the duty source, F0_DUTY or F0_DUTY_HR |
| **VFO/PWM Commands - F1** | | | | |
| **208** | **F1_VFO** | %MX3314.9 | BOOL | 0 -> Disable VFO/PWM in the F1 output (stopped output) 1 -> Enable VFO/PWM in the F1 output |
| **209** | **F1_VFO_DUTY_SRC** | %MX3314.11 | BOOL | Select the duty source, F1_DUTY or F1_DUTY_HR |
| **PTO Diagnostic - F0** | | | | |
| **210** | **F0_PTO_ON** | %MX3315.0 | BOOL | PTO output in operation |
| **211** | **F0_PTO_ACE** | %MX3315.1 | BOOL | PTO output in acceleration |
| **212** | **F0_PTO_REG** | %MX3315.2 | BOOL | PTO output in target velocity |
| **213** | **F0_PTO_DES** | %MX3315.3 | BOOL | PTO output in deceleration |
| **214** | **F0_PTO_PRM** | %MX3315.4 | BOOL | PTO output not parameterized |
| **215** | **F0_PTO_ERR** | %MX3315.5 | BOOL | PTO output with parameterization error |
| **216** | **F0_PTO_CNT_MAX_CMP0** | %MX3138.8 | BOOL | Indicates that the pulse counter is greater than the comparator 0 at the F0 output. |
| **217** | **F0_PTO_CNT_MAX_CMP1** | %MX3138.9 | BOOL | Indicates that the pulse counter is greater than the comparator 1 at the F0 output. |
| **PTO Diagnostic - F1** | | | | |
| **218** | **F1_PTO_ON** | %MX3315.8 | BOOL | PTO output in operation |
| **219** | **F1_PTO_ACE** | %MX3315.9 | BOOL | PTO output in acceleration |
| **220** | **F1_PTO_REG** | %MX3315.10 | BOOL | PTO output in target velocity |
| **221** | **F1_PTO_DES** | %MX3315.11 | BOOL | PTO output in deceleration |
| **222** | **F1_PTO_PRM** | %MX3315.12 | BOOL | PTO output not parameterized |
| **223** | **F1_PTO_ERR** | %MX3315.13 | BOOL | PTO output with parameterization error |
| **224** | **F1_PTO_CNT_MAX_CMP0** | %MX3139.0 | BOOL | Indicates that the pulse counter is greater than the comparator 0 at the F1 output. |
| **225** | **F1_PTO_CNT_MAX_CMP1** | %MX3139.1 | BOOL | Indicates that the pulse counter is greater than the comparator 1 at the F1 output. |
| **VFO/PWM Diagnostic - F0** | | | | |
| **226** | **F0_VFO_ON** | %MX3316.0 | BOOL | Output in operation |
| **227** | **F0_VFO_PRM** | %MX3316.1 | BOOL | Output not parameterized |

| | | | | |
|---|---|---|---|---|
| **228** | **F0_VFO_ERR** | %MX3316.2 | BOOL | Output with parameterization error |
| **VFO/PWM - F1 Diagnostic - F1** | | | | |
| **229** | **F1_VFO_ON** | %MX3316.3 | BOOL | Output in operation |
| **230** | **F1_VFO_PRM** | %MX3316.4 | BOOL | Output not parameterized |
| **231** | **F1_VFO_ERR** | %MX3316.5 | BOOL | Output with parameterization error |
| **Analog Outputs Diagnostic** | | | | |
| **232** | **AO0_DG** | %MB6640 | BYTE | Channel 0 diagnostics of analog output |
| **233** | **AO1_DG** | %MB6641 | BYTE | Channel 1 diagnostics of analog output |
| **234** | **AO0_ERR** | %MX3320.0 | BOOL | Channel 0 in short-circuit (voltage) or open loop (current) |
| **235** | **AO1_ERR** | %MX3320.8 | BOOL | Channel 1 in short-circuit (voltage) or open loop |
| **Analog Inputs Diagnostic** | | | | |
| **236** | **AI0_DG** | %MB6648 | BYTE | Channel 0 diagnostics of analog input |
| **237** | **AI1_DG** | %MB6649 | BYTE | Channel 1 diagnostics of analog input |
| **238** | **AI2_DG** | %MB6650 | BYTE | Channel 2 diagnostics of analog input |
| **239** | **AI3_DG** | %MB6651 | BYTE | Channel 3 diagnostics of analog input |
| **240** | **AI0_OPN** | %MX3324.0 | BOOL | Analog input in current 4-20 mA mode with open loop |
| **241** | **AI1_OPN** | %MX3324.8 | BOOL | Analog input in current 4-20 mA mode with open loop |
| **242** | **AI2_OPN** | %MX3325.0 | BOOL | Analog input in current 4-20 mA mode with open loop |
| **243** | **AI3_OPN** | %MX3325.8 | BOOL | Analog input in current 4-20 mA mode with open loop |
| **Digital Inputs Block 0** | | | | |
| **244** | **I00** | %IX0.0 | BOOL | Digital Input I00 |
| **245** | **I01** | %IX0.1 | BOOL | Digital Input I01 |
| **246** | **I02** | %IX0.2 | BOOL | Digital Input I02 |
| **247** | **I03** | %IX0.3 | BOOL | Digital Input I03 |
| **248** | **I04** | %IX0.4 | BOOL | Digital Input I04 |
| **249** | **I05** | %IX0.5 | BOOL | Digital Input I05 |
| **250** | **I06** | %IX0.6 | BOOL | Digital Input I06 |
| **251** | **I07** | %IX0.7 | BOOL | Digital Input I07 |
| **252** | **I08** | %IX0.8 | BOOL | Digital Input I08 |
| **Digital Inputs Block 1** | | | | |
| **253** | **I10** | %IX1.0 | BOOL | Digital Input I10 |
| **254** | **I11** | %IX1.1 | BOOL | Digital Input I11 |
| **255** | **I12** | %IX1.2 | BOOL | Digital Input I12 |
| **256** | **I13** | %IX1.3 | BOOL | Digital Input I13 |
| **257** | **I14** | %IX1.4 | BOOL | Digital Input I14 |
| **258** | **I15** | %IX1.5 | BOOL | Digital Input I15 |
| **259** | **I16** | %IX1.6 | BOOL | Digital Input I16 |
| **260** | **I17** | %IX1.7 | BOOL | Digital Input I17 |
| **261** | **I18** | %IX1.8 | BOOL | Digital Input I18 |
| **Digital Inputs Block 2** | | | | |
| **262** | **I20** | %IX2.0 | BOOL | Digital Input I20 |
| **263** | **I21** | %IX2.1 | BOOL | Digital Input I21 |
| **Analog Inputs** | | | | |
| **264** | **AI0** | %IW4 | WORD | Channel 0 value |
| **265** | **AI1** | %IW5 | WORD | Channel 1 value |
| **266** | **AI2** | %IW6 | WORD | Channel 2 value |
| **267** | **AI3** | %IW7 | WORD | Channel 3 value |
| **Digital Outputs Block 0** | | | | |
| **268** | **Q00** | %QX0.0 | BOOL | Digital Output Q00 |
| **269** | **Q01** | %QX0.1 | BOOL | Digital Output Q01 |
| **270** | **Q02** | %QX0.2 | BOOL | Digital Output Q02 |

| | | | | |
|---|---|---|---|---|
| **271** | **Q03** | %QX0.3 | BOOL | Digital Output Q03 |
| **272** | **Q04** | %QX0.4 | BOOL | Digital Output Q04 |
| **273** | **Q05** | %QX0.5 | BOOL | Digital Output Q05 |
| **274** | **Q06** | %QX0.6 | BOOL | Digital Output Q06 |
| **275** | **Q07** | %QX0.7 | BOOL | Digital Output Q07 |
| **Digital Outputs Block 1** | | | | |
| **276** | **Q10** | %QX1.0 | BOOL | Digital Output Q10 |
| **277** | **Q11** | %QX1.1 | BOOL | Digital Output Q11 |
| **278** | **Q12** | %QX1.2 | BOOL | Digital Output Q12 |
| **279** | **Q13** | %QX1.3 | BOOL | Digital Output Q13 |
| **280** | **Q14** | %QX1.4 | BOOL | Digital Output Q14 |
| **281** | **Q15** | %QX1.5 | BOOL | Digital Output Q15 |
| **282** | **Q16** | %QX1.6 | BOOL | Digital Output Q16 |
| **283** | **Q17** | %QX1.7 | BOOL | Digital Output Q17 |
| **Analog Outputs** | | | | |
| **284** | **AO0** | %QW3 | WORD | Channel 0 value |
| **285** | **AO1** | %QW4 | WORD | Channel 1 value |

**Table 6-9. Reserved Operands**

The following reserved operands groups are declared in the configuration tree: Digital Inputs Block 0, Digital Inputs Block 1, Digital Inputs Block 2, Analog Inputs, Digital Outputs Block 0, Digital Outputs Block 1, and Analog Output.

# 7. Special applications with Serial RS-232

This chapter describes how the RS-232 serial interface (COM1) may be used in special applications requiring the use of control signals (RTS, CTS, DTR and DSR), as well as of normal data signals (TXD and RXD).

## RTS/CTS Hardware Handshake in Radio Modems

Usually, a radio has its carrier switched (on) only when it is transmitting; and off when it is not transmitting. This happens due to the following reasons:

- to save energy while the radio is not transmitting
- to avoid overheating transmitter
- to allow another radio to use the same frequency while it is not transmitting

In handheld radio transceivers (walkie-talkies), for example, usually there is a PTT button (push to talk) that must be pressed before the operator can talk (and released after that). In the event of data transmission via radio modems, the controller RTS output should be used to trigger the radio PTT and turn the carrier on. After the PTT connection, hypothetically, the controller may start the data transfer through its TXD output. However, in practice, there is a stabilizing delay. As a result, when triggering the RTS (PTT radio), the carrier is on, but only after a while it will be stable and recognized by the radio(s) receiver(s). This period of time varies according to each modem-radio model. Finally, in order forthat the controller to synchronize the time of the data transmission start up (TXD), the modem radio returns an output (CTS), which is turned on in the controllers CTS input.

Therefore, the data transmission protocol between the radio and the modem controller, called the RTS/CTS handshake, occurs as follows:

1. When the controller wants to transmit, the RTS output is turned on. This output must be connected to the RTS radio modem input. Note that in the controller, RTS is an output, and in the radio modem, RTS is an input.

2. When the radio modem RTS realizes that the RTS input was turned on, it turns the carrier on (PTT) and, after a typical time of this kind of radio, it turns the CTS output on.

3. When the controller realizes that the CTS is on, it starts the data transmission via TXD output, which is connected to the radio modem TXD input. Note that in the modem radio, CTS is an output and in the CTS controller it is an input, as well as the TXD, which is an output on the controller and an input in the radio modem.

4. If the driver does not receive the CTS radio modem return in 1 second after the RTS output was turned on and requesting the transmission, the transmission process is aborted and an error is signaled (CTS timeout).

5. If the PLC received the CTS return before 1 second, the data transmission is started via TXD. The RTS output is turned off as soon as the data transmission ends. When the radio modem realizes that, it turns off the carrier (PTT), and its CTS output.

The following figure shows the timing of the RTS, CTS, and TXD signals during data transmission. In addition, the DCD item (carrier detection) illustrates what happens in a radio DCD signal receiving that data transmission. The RXD item illustrates what happens in a radio signal RXD receiving that data transmission.

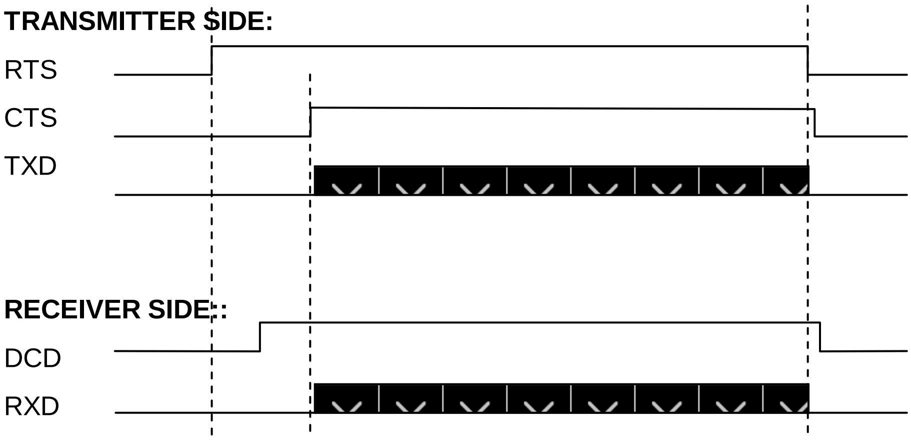


**Figure 7-1. RTS, CTS and TXD Signals Timing**

The following figure illustrates how the PLC and the modem radio must be connected.

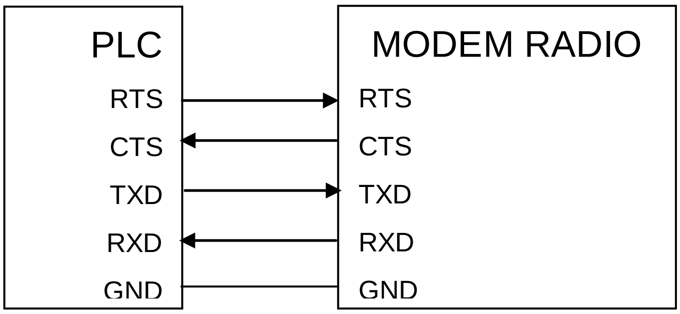


**Figure 7-2. PLC and Modem Radio Connection**

| |
|---|
| ATTENTION:<br>Some modern and smart radio modems do not require the RTS/CTS handshake, since their PLC interconnection cable renders both the RTS and the CTS signals unnecessary. These radio modems manage automatically the carrier activation / deactivation (PTT) and analyze the data signal (TXD). This management is done via microchip or another smart circuit installed in the radio modem. |

## RTS/CTS Hardware Handshake on RS-485 Converters

RS-485 converters are installed on buses where two or more pieces of equipment with an RS-485 interface can coexist sharing the same physical layer for transmission and reception. Thus, only one of the converters can perform the data transmission at any given moment in order to avoid colliding data.

Similarly to the radio modems situation, the RTS signal should be used to enable the RS-485 converter transmitter. When the RS-485 converter is not operating, the transmitter must be disabled or kept in “high impedance”.

Usually, the main difference between radio modems and RS-485 converters is the carrier stabilization time. In modems, time is considerable (around a few milliseconds). In RS-485converters, the data transmission via TXD may be started as soon as the RTS signal is activated. This action causes the

CTS test to be dismissed (whereas it is necessary in the case of radio modems).

However, to avoid the creation of another handshake type, the RTS/CTS handshake described for radio modems is used. Most of the RS-485 converters enable the CTS output immediately after receiving the RTS input, or they even interconnect their RTS and CTS pins. If the converter does not hold a CTS pin, a cable connection may be provided.

There are two recommended methods of interconnection between the controller and the RS-485 converter. The following figure exhibits these two methods.

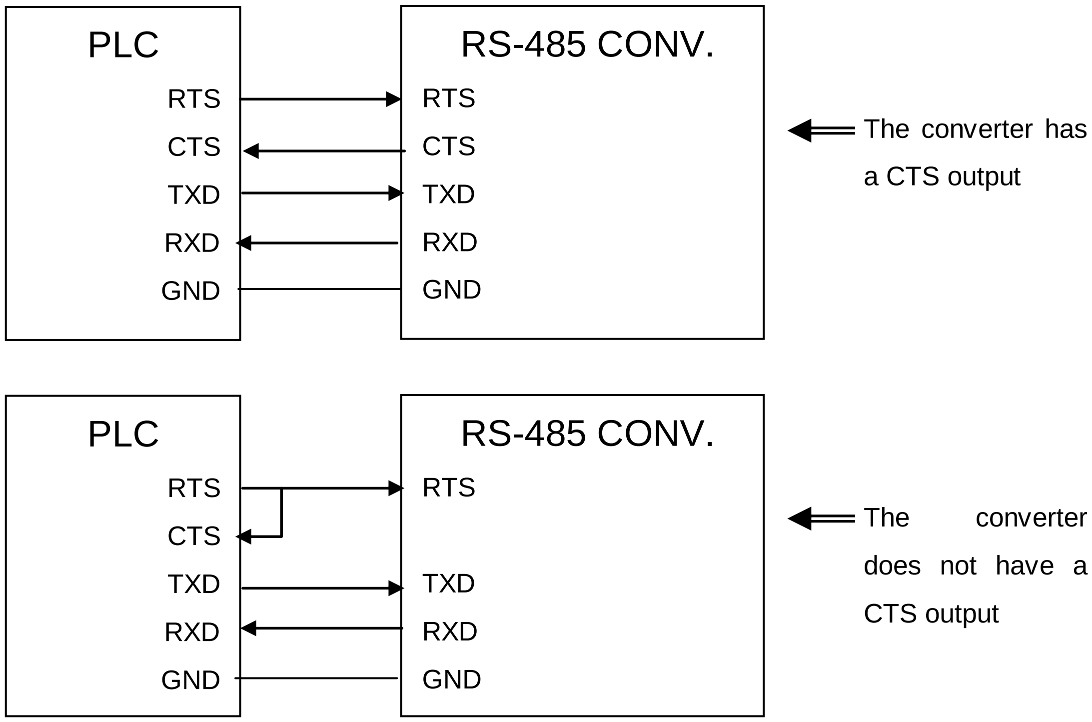


**Figure 7-3. PLC and RS-485 Converter Connection**

The figure below shows the timing of RTS, CTS, and TXD signals during data transmission.

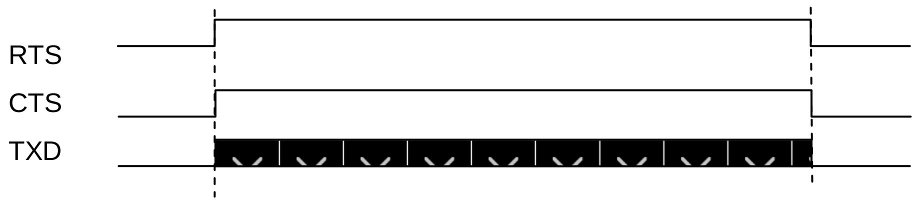


**Figure 7-4. RTS, CTS and TXD Signals Timing**

ATTENTION:
Some modern and smart RS-232 to RS-485 converters do not require the RTS/CTS handshake, since their PLC interconnection cable renders both the RTS and the CTS signals unnecessary. These converters manage automatically the carrier activation / deactivation and analyze the data signal (TXD). This management is done via microchip or another smart circuit installed in the RS-232 to RS-485 converter.

# 8. Glossary

| | |
|---|---|
| **AES** | Function for instantaneous updating of digital or analog inputs and outputs. |
| **Application Program** | Program downloaded into the PLC and has the instructions that define how the machinery or process will work. |
| **ARRAY** | Used to declare a data structure that holds a series of elements of the same size and type. |
| **Backlight** | Display background lighting. |
| **Bit** | Basic information unit, it may be at 1 or 0 logic level. |
| **Brownout** | Electronic circuit to check the integrity of the power supply operation. Used for triggering a security logic in case of power failures. |
| **Bus** | Set of electrical signals that are part of a logic group with the function of transferring data and control between different elements of a subsystem. |
| **Byte** | Information unit composed by eight bits. |
| **CPU** | Central Processing Unit. It controls the data flow, interprets and executes the program instructions as well as monitors the system devices. |
| **Crosstalk** | Is interference between two signals theoretically insulated. |
| **Cycle** | Complete application program implementation of a programmable controller. |
| **Diagnostics** | Procedures to detect and isolate failures. It also relates to the data set used for such tasks, and serves for analysis and correction or problems. |
| **Duty Cycle** | Percentage of time that a periodic square wave signal is at logical level 1 considering the same period of time. |
| **DWord** | Double Word. |
| **Encoder** | Normally refers to position measurement transducer. |
| **Firmware** | The operating system of a PLC. It controls the PLC basic functions and executes the application programs. |
| **Function block** | A function block is a POU that provides one or more values during its use. Unlike the function, the function block does not provide any return value. |
| **Handshake** | Process by which two machines report to each other that it was recognized and is ready to start the communication. |
| **Hardware** | Physical equipment used to process data where normally programs (software) are executed). |
| **HMI** | Human Machine Interface. |
| **IEC** | *International Electrotechnical Commission*. It is an international standards for preparing and publishing international standards for electrical, electronic and related technology. |
| **IEC 61131-3** | Third part of the generic standard for operation and usage of programmable controllers. |
| **Interface** | Normally used to refer to a device that adapts electrically or logically the transferring of signals between two equipments. |
| **Interruption** | Priority event that temporarily halts the normal execution of a program. The interruptions are divided into two generic types: hardware and software. The former is caused by a signal coming from a peripheral, while the later is caused within a program. |
| **Kbytes** | Memory size unit. Represents 1024 bytes. |
| **MasterTool IEC** | The Altus WINDOWS® based programming software that allows application software development for PLCs from the Ponto, Grano, Piccolo, AL-2000, AL-3000 and Quarks series. Throughout this manual, this software is referred by its code or as MasterTool Programming. |
| **Menu** | Set of available options for a program, they may be selected by the user in order to activate or execute a specific task. |
| **MODBUS** | Data communication protocol for industrial networks created for networks with master-slave architecture. |
| **Module (hardware)** | Basic element of a system with very specific functionality. It's normally connected to the system by connectors and may be easily replaced. |
| **Module (software)** | Part of a program capable of performing a specific task. It may be executed independently or in conjunction with other modules through information sharing by parameters. |
| **NAVIGATION** | Function block for browsing in the user screens via Up arrow and Down arrow keys. |
| **Operands** | Elements on which software instructions work. They may represent constants, variables or set of variables. |
| **Programmable Controller** | Also know as PLC. Equipment controlling a system under the command of an application program. It is composed of a CPU, a power supply and I/O modules. |
| **PLC** | See Programmable Controller. |
| **POU** | *Program Organization Unit*, is a subdivision of the application program that can be written in any of the available languages. |
| **Programming Language** | Set of rules, conventions and syntaxes utilized when writing a program. |
| **PTO** | Pulse Train Output. |

| | |
|---|---|
| **PWM** | Pulse Width Modulation. |
| **RS-232** | It is a standard for serial data change between two points (point to point). |
| **RS-485** | It is a standard for serial data change between two or more points (multipoint). |
| **RTC** | Real Time Clock. |
| **Safe state** | Condition in which the PLC changes its analog and digital outputs to a known state. |
| **Software** | Computer programs, procedures and rules related to the operation of a data processing system. |
| **Task** | A task is a unit of time in processing a IEC program. It is defined by a name, a priority and a given type of condition that will trigger its startup. |
| **Timeout** | Maximum preset time to a communication to take place. When exceeded, then retry procedures are started or diagnostics are activated. |
| **Upload** | Reading a program or configuration from the PLC. |
| **VFO** | Variable Frequency Output. |
| **Watchdog** | Electronic circuit that checks the equipment operation integrity. |
| **Word** | Information unit composed by 16 bits. |