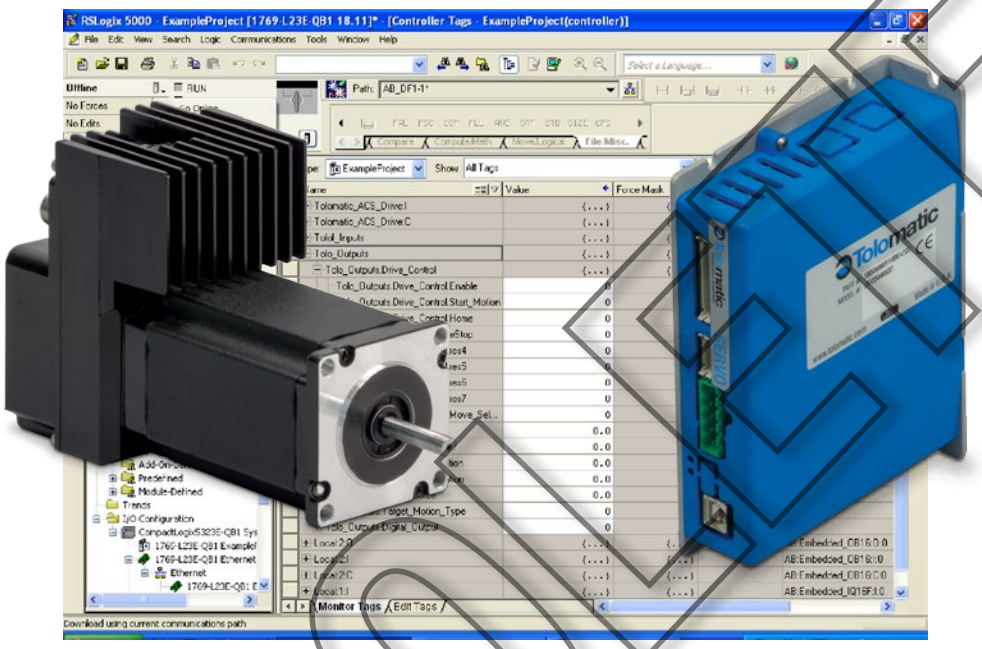
Tolomatic
EXCELLENCE IN MOTION

Technical Note

Setting Up the Allen Bradley RSLogix 5000 Software for EtherNet/IP Communication to Tolomatic's ACS Drive



Contents

1 System Requirements

HARDWARE	MINIMUM REQUIREMENTS
Processor	Intel Pentium II 450 MHz Min Intel Pentium III 733 MHz (or better) recommended
Software Requirements	Windows XP with Service Pack 2 (or above)
RAM	128 MB Minimum 256 MB Recommended
Disk Space	3 GB Free HDD space
PLC	Allen Bradley ControlLogix L3X or above
RSLogix	5000 version 18
Cables	USB CAT 5e

NOTE: Tolomatic's Motion Interface is dependent on .NET 4. Reference the TMI User Guide 3600-4167 for minimum requirements.

2 Cabling

Setup and establish communication between RSLogix and the controller. Connect an Ethernet cable to the PLC controller and to the ACS Drive.

3 Setting up the Tolomatic ACS Drive IP Address

Using the Tolomatic Motion Interface (TMI) program with the ACS, go to Tools -> Ethernet menu selection or press the Configure Ethernet button on the toolstrip. Assign the IP address of the drive (Figure 1). By default the ACS will be set for DHCP mode. A static IP address can be set using the "Ethernet Setup Tool" in TMI or the TCP/IP object (OXF5) in EtherNet/IP. For additional instruction, refer to the "Ethernet Setup Tool" section of the TMI User Guide 3600-4167

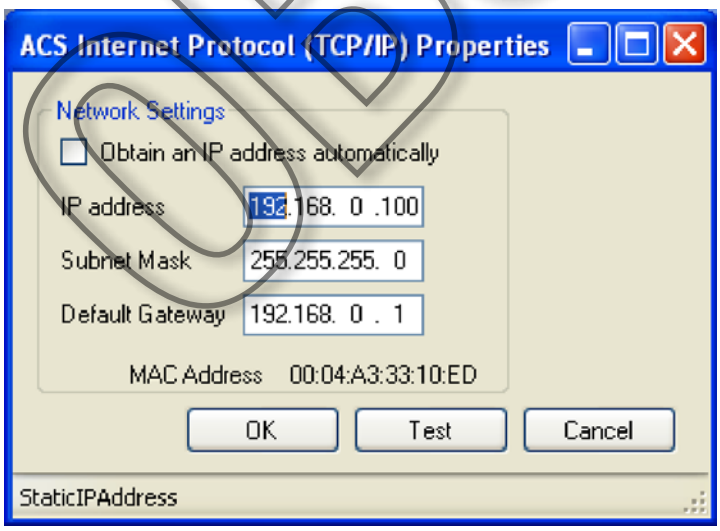


Figure 1: Assigning ACS Drive an IP Address.

4 Setting Up the Allen Bradley PLC Using the RSLogix 5000 Software

This instruction will walk through how to add an Ethernet module to a PLC controller, create data types, and download configurations and instructions to the controller.

4.1 Adding an Ethernet Module

Open RS Logix and select a 'New Project.'

Figure 2: Select New Project

In the 'New Controller' window, select controller, give project a name, and choose a directory to save the project.

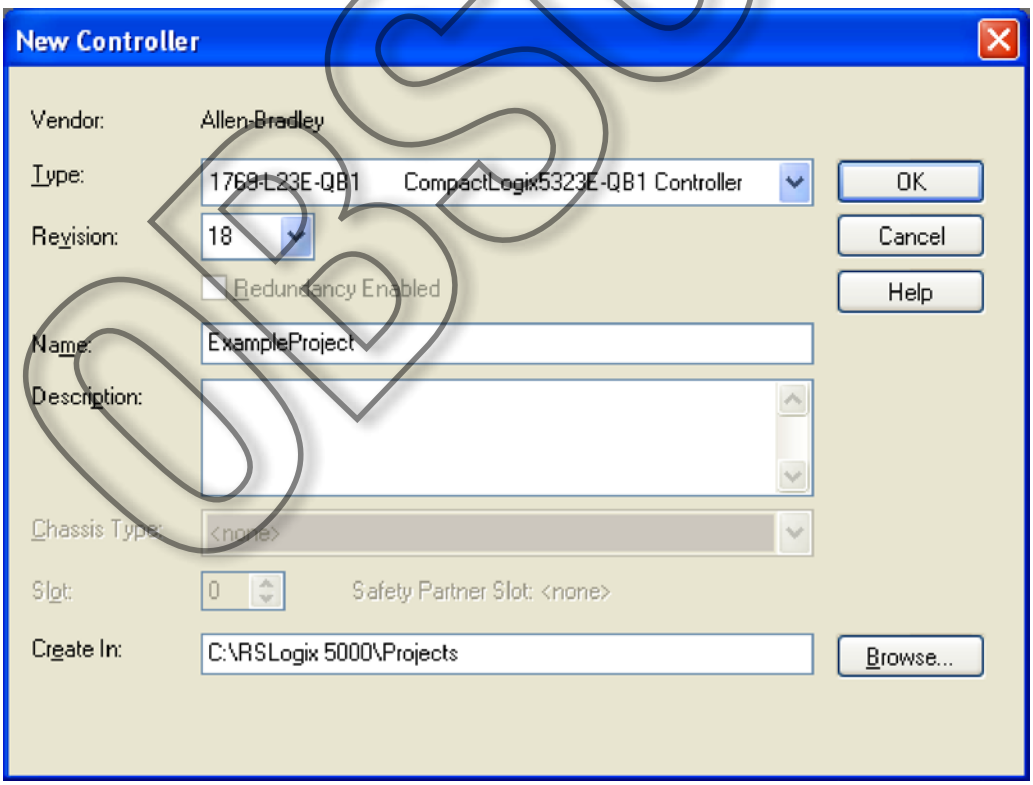


Figure 3: RSLogix New Project Window

Next, add a generic EtherNet/IP module; right click on 'Ethernet' and select 'New Module'.

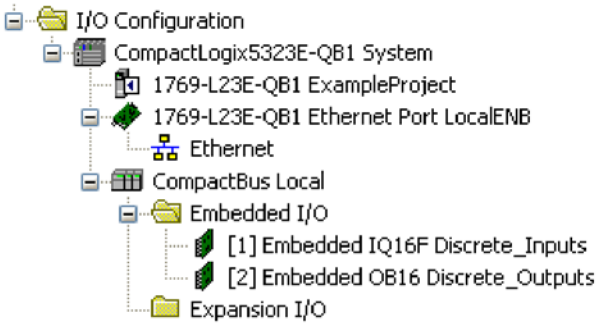


Figure 4: Adding a Generic EtherNet/IP Module

The 'Select Module' window will open. Choose the 'Generic Ethernet Module' and click 'OK". For details on how to use an Add on Profile & Instructions see document 3600-4188 "Using Add-On Instructions".

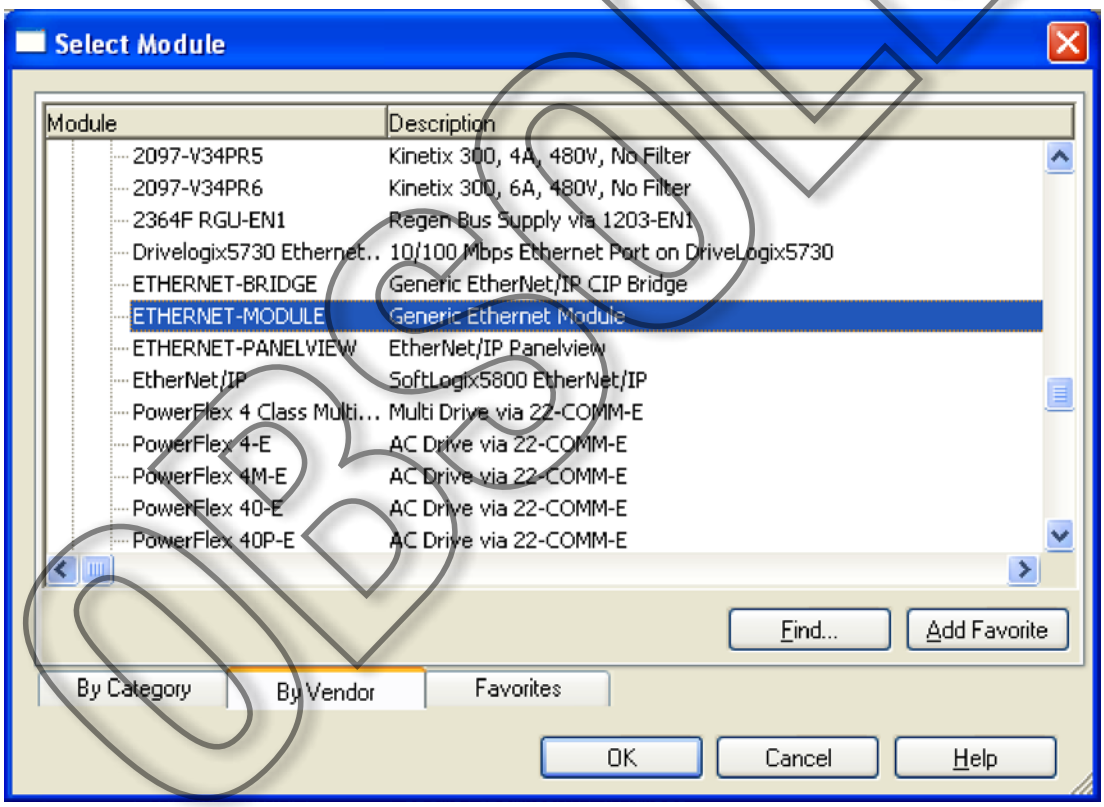


Figure 5: Selecting a Generic Ethernet Module

NOTE: An IP address can be obtained via DHCP by configuring the PLC. For more information on this procedure, please reference the the software/hardware manual for the PLC in use.

The new module properties window should have opened. Enter a name for the module, an IP address for the drive, and the assembly object parameters and click 'OK'. Set up Assembly Instances for the ACS drive as shown in Figure 6. The IP address must match the address configured in the Tolomatic Motion Interface Software (see section 3 of the TMI User Guide 3600-4167.

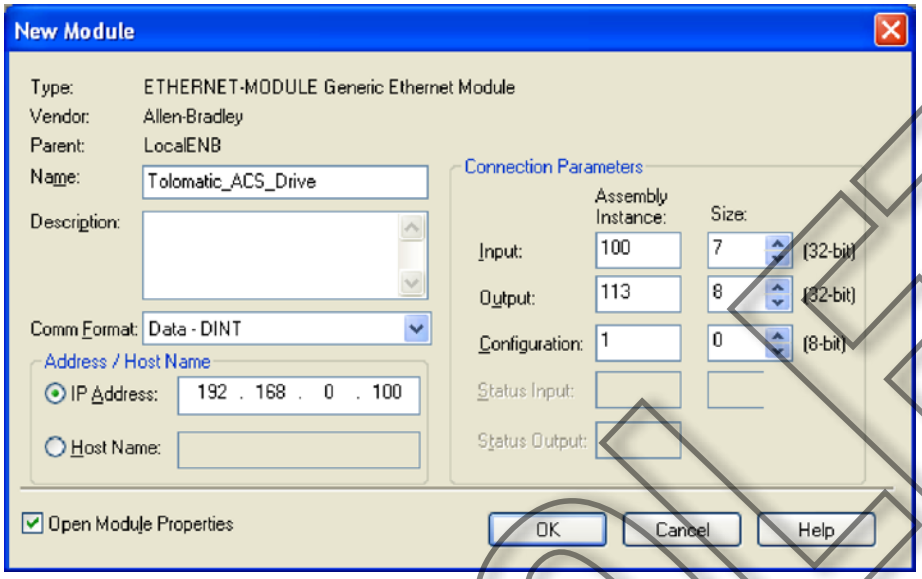


Figure 6: New Module Properties Window

In the connection tab, select these settings or other Requested Packet Interval (RPI). This value determines the interval the controller will use to send/receive data. To conserve bandwidth, use higher values. Click 'OK' when finished. **NOTE:** Setting the packet interval to low may result in erratic motion. Recommended RPI is 20.0 ms. Fastest RPI is 10.0 ms.

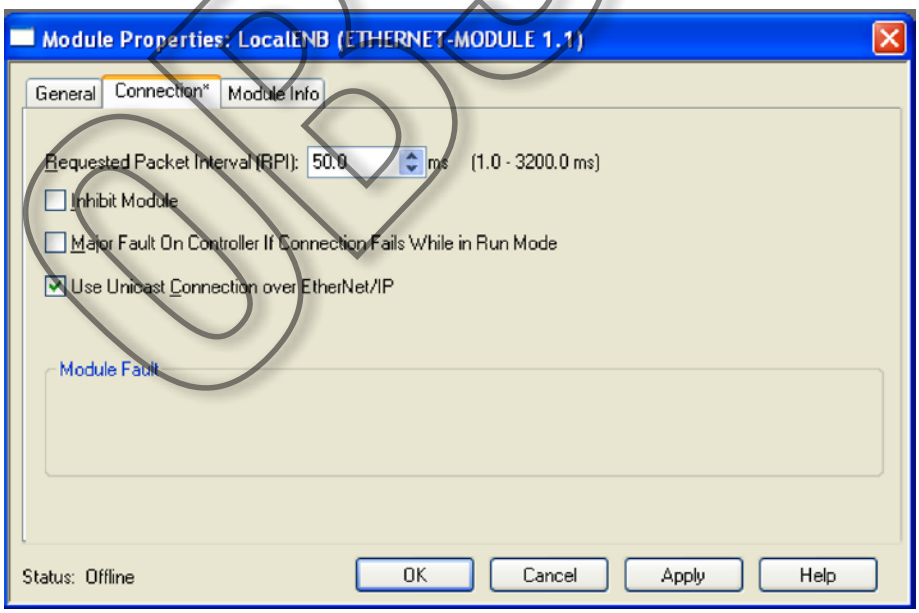


Figure 7: Module Properties Connection Tab

Now the module should have been automatically added in the organizer window.

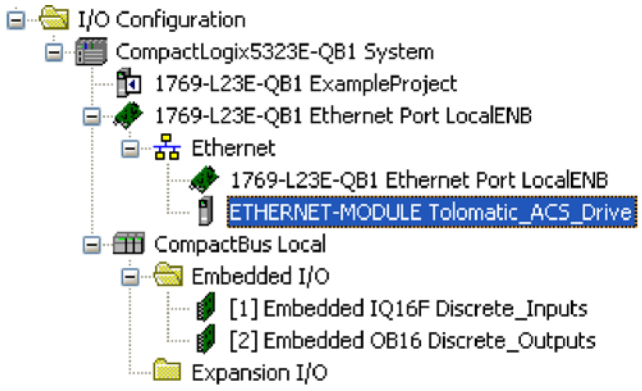


Figure 8: Ethernet Module Added to Organizer Window

A new node named Tolomatic_ACS_Drive now exists under I/O Configuration. Module-defined data types have also been created. These tags allow access to the Input and Output data of the ACS drive using the controller's ladder logic.

4.2 Download Configuration to Controller

Download the previous configurations to the controller and save the project.

4.3 Controller Tags

In the organizer window, expand 'Module-Defined' under 'Data Types'. Make sure Ethernet module data types are:

AB:ETHERNET_MODULE_DINT_28Bytes:I:0

AB:ETHERNET_MODULE_DINT_32Bytes:O:0

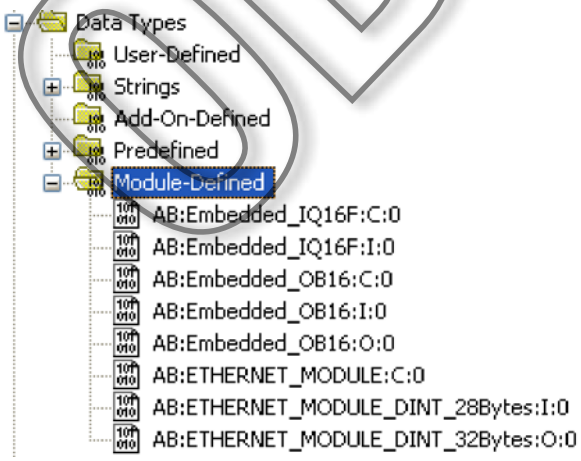


Figure 9: Verify Ethernet Module Data Types in Organizer Window

Next, double-click on 'Controller Tags' in the organizer window.

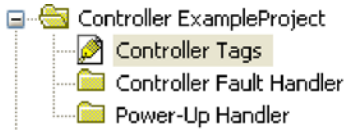


Figure 10: Controller Tags in Organizer Window

A window opens showing all of the existing controller tags. With the new Ethernet module, the controller tags were also created.

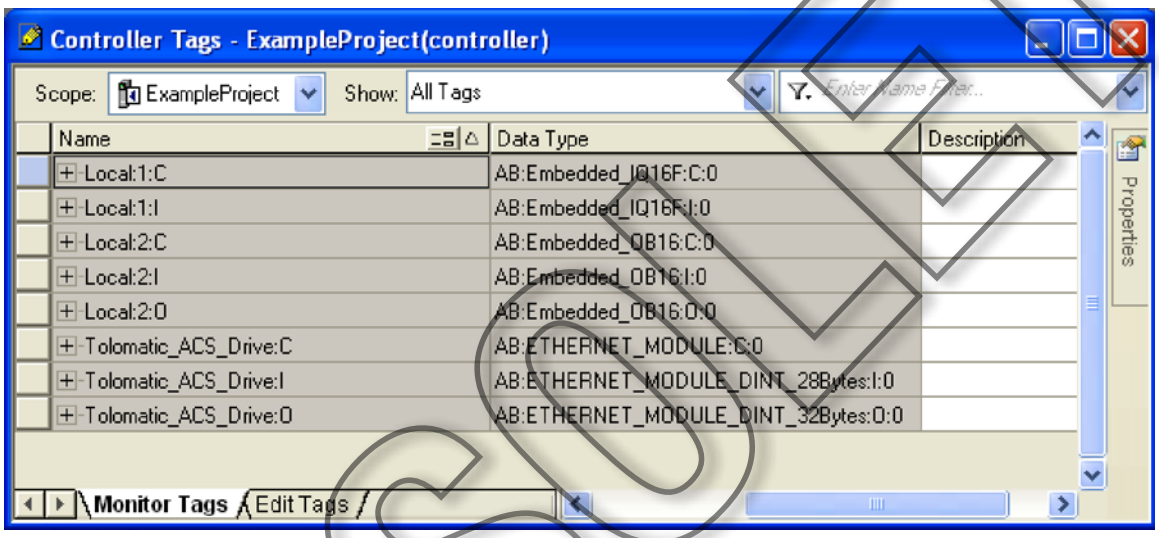


Figure 11: Controller Tags Window

This example uses the new Ethernet module: Tolomatic_ACS_Drive:C, Tolomatic_ACS_Drive:I, and Tolomatic_ACS_Drive:O.

Click the '+' next to each tag to expand the bytes. The data tags are listed numerically.

Name	Value	Force Mask	Style
+ Tolomatic_ACS_Drive:O	{...}	{...}	
− Tolomatic_ACS_Drive:I	{...}	{...}	
− Tolomatic_ACS_Drive:I.Data	{...}	{...}	Decimal
+ Tolomatic_ACS_Drive:I.Data[0]	0		Decimal
− Tolomatic_ACS_Drive:I.Data[1]	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].0	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].1	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].2	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].3	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].4	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].5	0		Decimal
Tolomatic_ACS_Drive:I.Data[1].6	0		Decimal

Figure 10: Expanded Controller Tags

Create user defined data types that reflect better naming conventions.

4.4 Creating Program Tags

Create two user defined data types called Tolo_Inputs, and Tolo_Outputs. To do this, Right click on 'User Defined' and select 'New Data Type'. Use the assembly object table as a reference to map the new program tags to each controller tag. Use naming conventions that will be easy to understand in the ladder logic. The following figures show each data type created.

Each bit in the 'Drive_Faults' register (bytes 8-11 of instance 100) represents a particular fault. The 'Drive_Faults' data type is created so each fault can be easily referenced by name instead of by bit number when a ladder logic program is defined.

Give each member a name, data type, and style to display a formatted number.

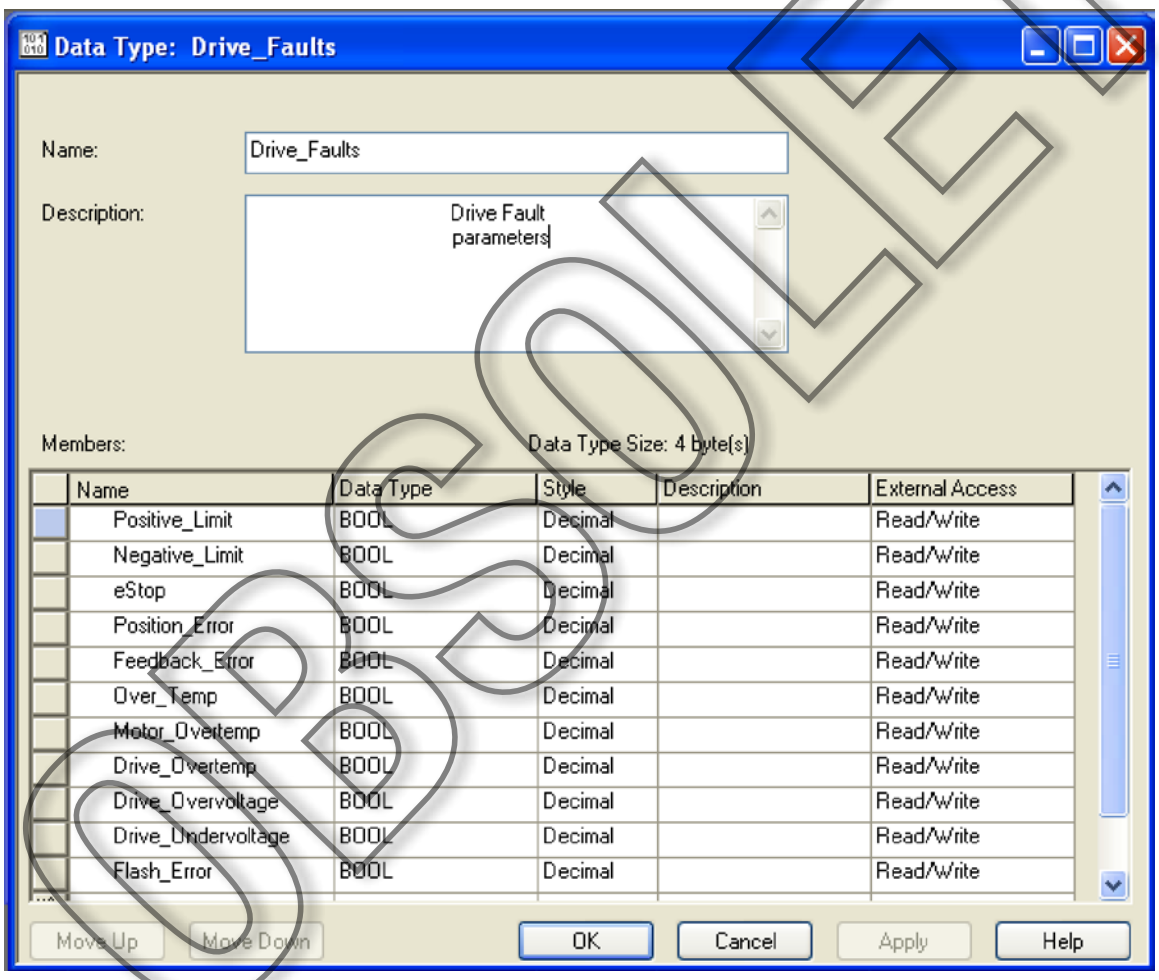


Figure 12: Creating a Drive Faults Data Type

Follow the same procedure for the drive 'Drive_Status' register. Some bits of the Drive_Status data type are not used in the ACS drive. Those are marked as reserved.

Members: Data Type Size: 4 byte(s)

Name	Data Type	Style	Description	External Access
Drive_Enable	BOOL	Decimal	Drive Enable Input	Read/Write
Drive_Homed	BOOL	Decimal	Drive Home Input	Read/Write
Drive_InMotion	BOOL	Decimal	Drive In Motion Input	Read/Write
Drive_EStop	BOOL	Decimal	Drive E-Stop Input	Read/Write
rsved_bit00	BOOL	Decimal		Read/Write
rsved_bit01	BOOL	Decimal		Read/Write
rsved_bit02	BOOL	Decimal		Read/Write
rsved_bit03	BOOL	Decimal		Read/Write
rsved_bit04	BOOL	Decimal		Read/Write
rsved_bit05	BOOL	Decimal		Read/Write
rsved_bit06	BOOL	Decimal		Read/Write
rsved_bit07	BOOL	Decimal		Read/Write
rsved_bit08	BOOL	Decimal		Read/Write
Drive_EthAvail	BOOL	Decimal	Drive Ethernet Online	Read/Write
rsved_bit09	BOOL	Decimal		Read/Write
rsved_bit10	BOOL	Decimal		Read/Write
rsved_bit11	BOOL	Decimal		Read/Write
rsved_bit12	BOOL	Decimal		Read/Write
rsved_bit13	BOOL	Decimal		Read/Write
rsved_bit14	BOOL	Decimal		Read/Write
Drive_BrakeOff	BOOL	Decimal	Brake Not Active	Read/Write
rsved_bit15	BOOL	Decimal		Read/Write
rsved_bit16	BOOL	Decimal		Read/Write

Figure 13: Drive Status Data Type

The previously created data types 'Drive_Faults' and 'Drive_Status' will now be used within our next data type. Create a data type called 'Tolo_Inputs'. Add these faults and status members using user defined data types. The other members can be created using the statndard data types DINT and REAL.

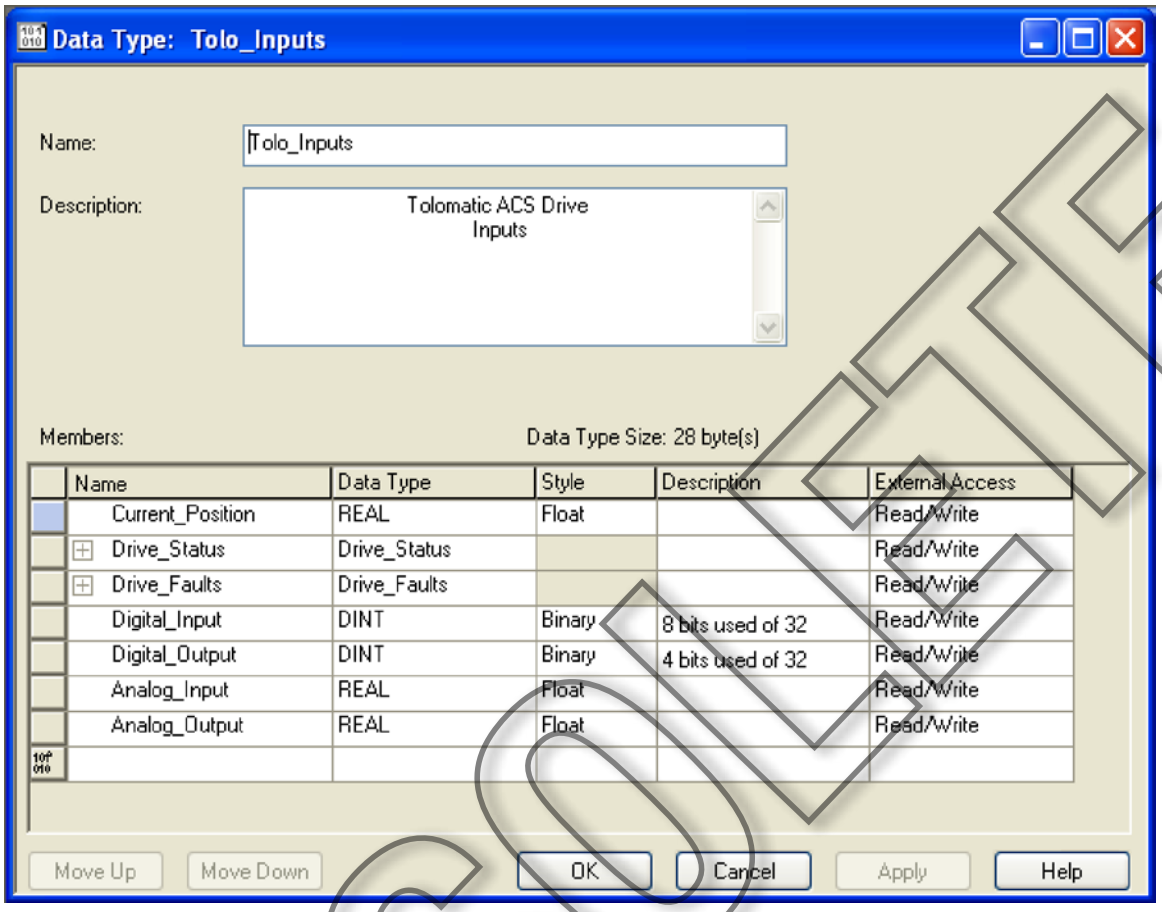


Figure 14: Creating a Tolo_Input Data Type

Next, create another data type called 'Network_Outputs' that will be used as a data type within the 'Tolo_Outputs' data type.

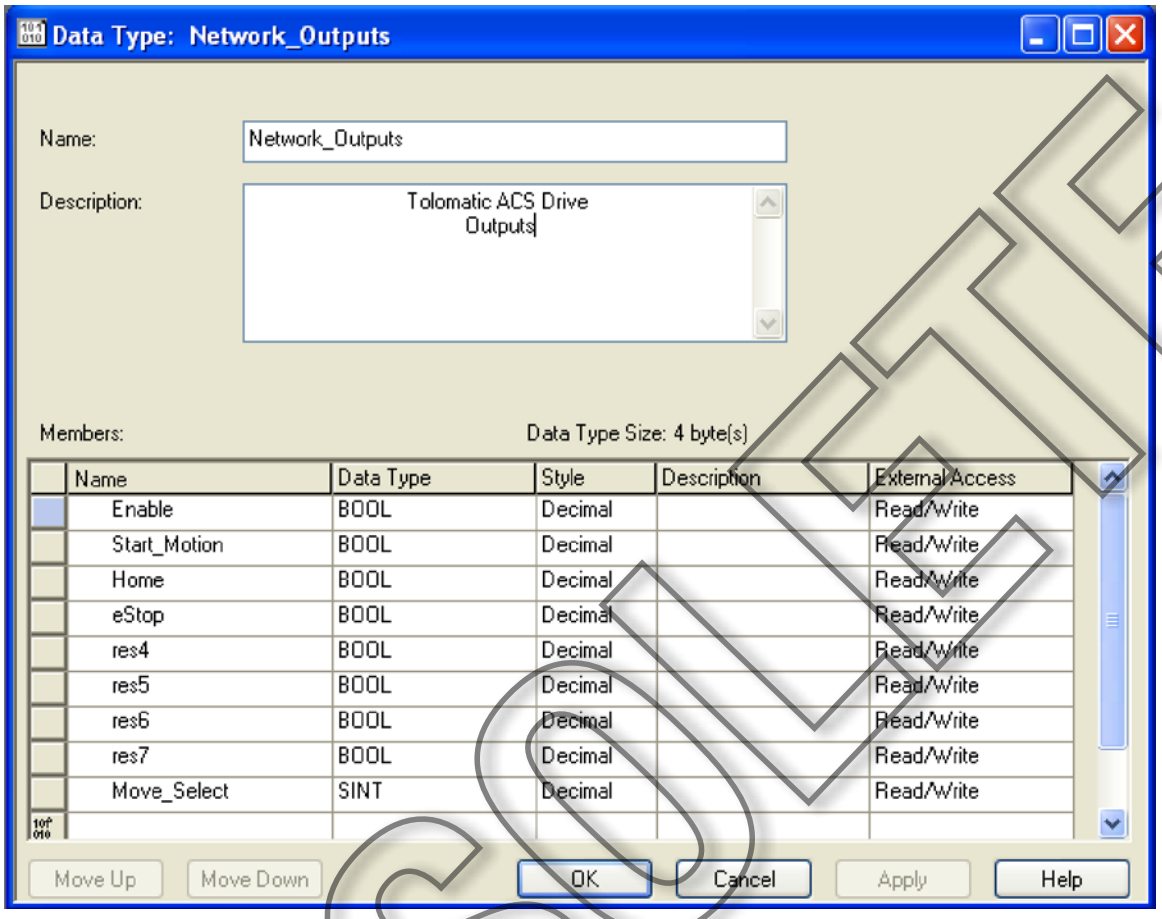


Figure 15: Creating a Network_Output Data Type

Next, create the 'Tolo_Outputs' data type as shown.

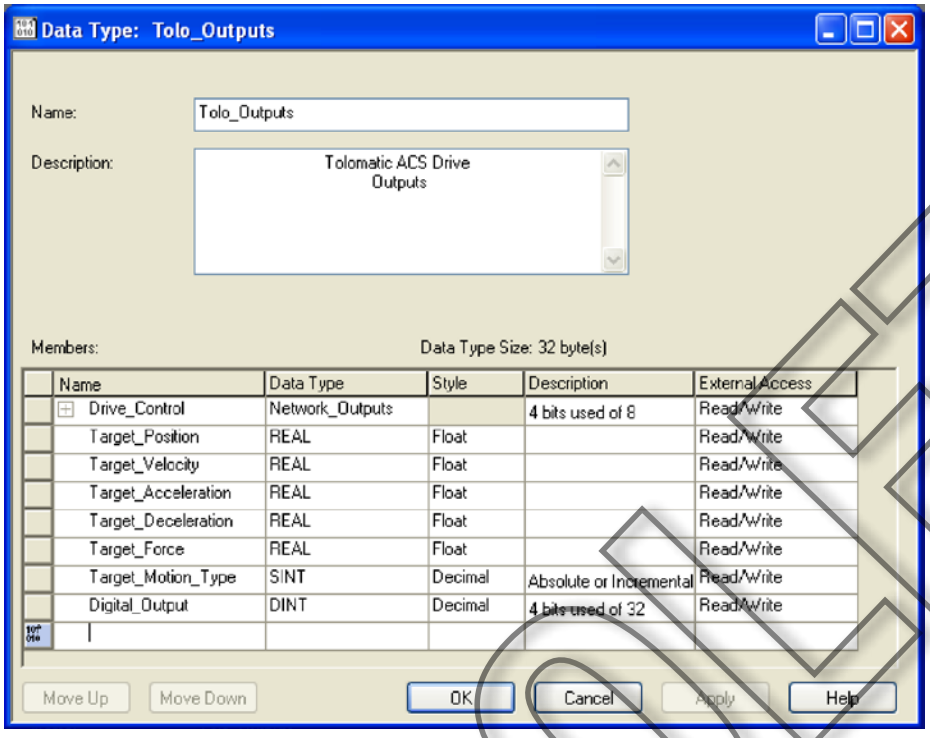


Figure 16: Create a Tolo_Outputs Data Type

Open the controller tags and click on the 'Edit' tab. Add Tolo_Inputs and Tolo_Outputs as the new data types just created.

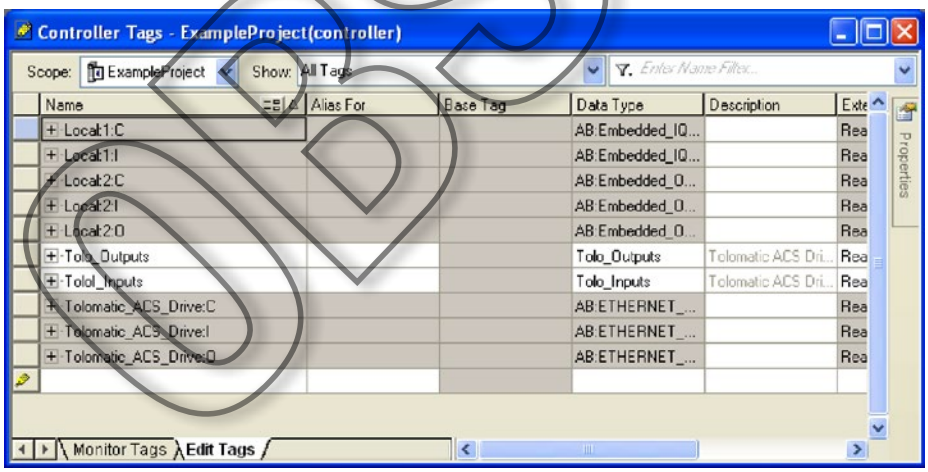


Figure 17: Add Tolo_Inputs and Tolo_Outputs as New Data Types

This completes setup of program tags with recognizable naming conventions.

4.5 Ladder Logic Instructions

Double-click on 'Main Routine' to enter the ladder logic program window.

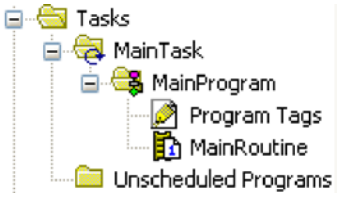


Figure 18: Main Routine in the Organizer Window

Insert 'Synchronous Copy File' instructions to copy the program tags to the controller tags.

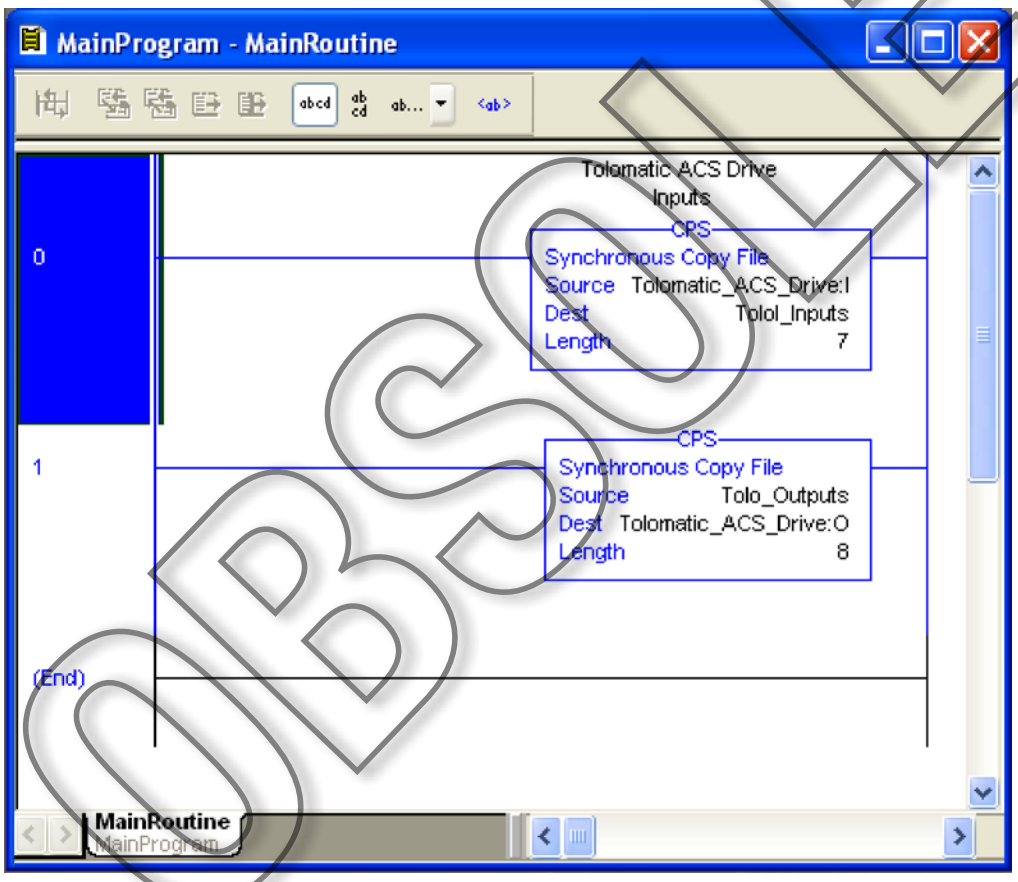


Figure 19: Inserting the Synchronous Copy File Instructions

Download the instructions to the controller and run the program. Turn the key on the PLC to RUN and the program should be online and running.

5 Make Motion

When the PLC is in RUN mode and RSLogix is online, manipulate the data tags in the controller tags window. If the proper bits are written, the drive will react as instructed.

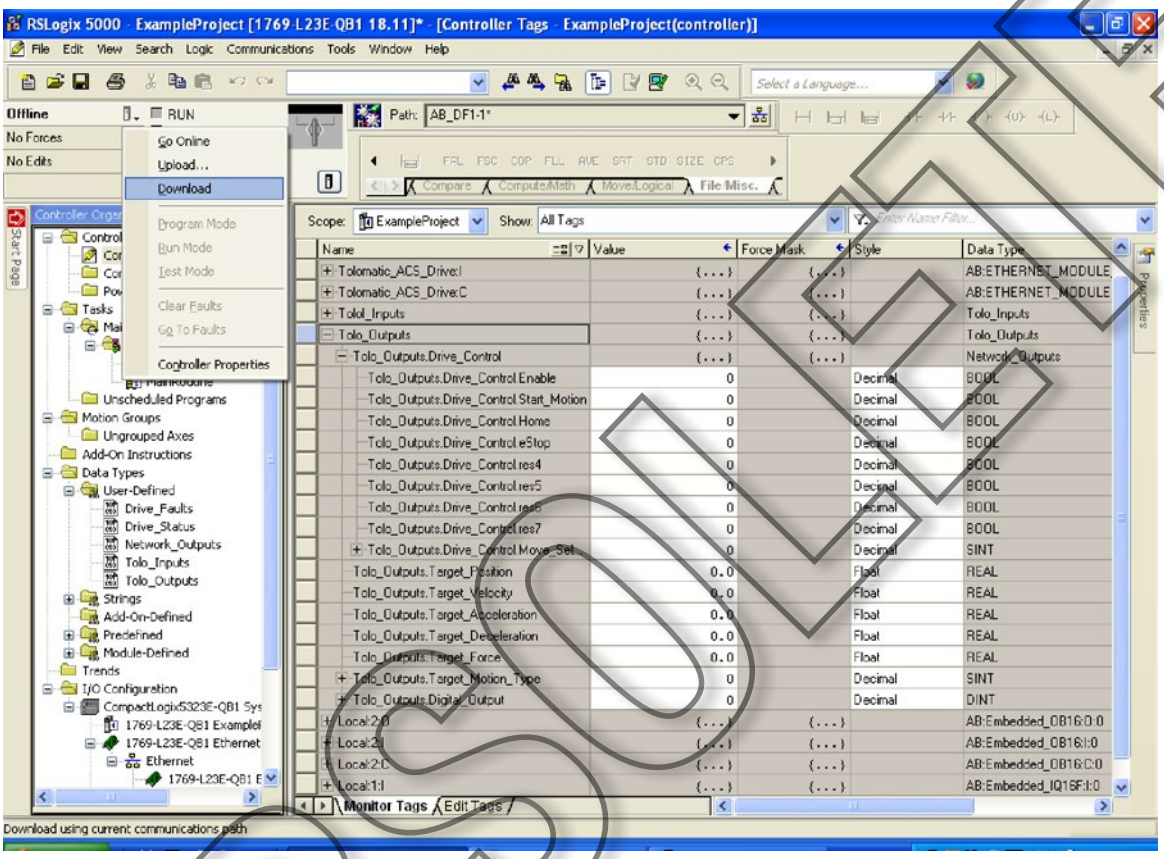


Figure 20: Manipulating Data Tags in Controller Tags Window

5.1 Home Move

The first step when bringing a drive online from power-up, is usually to have it find the actuator's home location. To do this, write a '1' to Enable and Home bits of Tolo_Outputs.Drive_Control. The drive should immediate begin the homing motion profile that was setup previously using TMI. When homing has successfully completed the tag Tolo_Inputs.Drive_Status.Drive_Home will turn to '1'.

Tolo_Outputs	{...}
Tolo_Outputs.Drive_Control	{...}
Tolo_Outputs.Drive_Control.Enable	1
Tolo_Outputs.Drive_Control.Start_Motion	0
Tolo_Outputs.Drive_Control.Home	1
Tolo_Outputs.Drive_Control.eStop	0

Figure 21: Finding the Home Postion

5.2 Absolute Move

To do an absolute move, set the Position, Velocity, Acceleration, and Force parameters of Tolo_Outputs.Drive_Control to desired values. Set Move_Select to '0' and Motion_Type to '0'. Then raise Start_Motion from '0' to '1'. Start_Motion begins the move only when its value cycles from '0' to '1' and Enable is also '1'. If the position has not changed since the last move there will be no motion because the actuator is already at the desired position.

Tolo_Outputs	{...}
Tolo_Outputs.Drive_Control	{...}
Tolo_Outputs.Drive_Control.Enable	1
Tolo_Outputs.Drive_Control.Start_Motion	1
Tolo_Outputs.Drive_Control.Home	0
Tolo_Outputs.Drive_Control.eStop	0
Tolo_Outputs.Drive_Control.res4	0
Tolo_Outputs.Drive_Control.res5	0
Tolo_Outputs.Drive_Control.res6	0
Tolo_Outputs.Drive_Control.res7	0
Tolo_Outputs.Drive_Control.Move_Select	0
Tolo_Outputs.Target_Position	55.2
Tolo_Outputs.Target_Velocity	25.0
Tolo_Outputs.Target_Acceleration	200.0
Tolo_Outputs.Target_Deceleration	200.0
Tolo_Outputs.Target_Force	100.0
Tolo_Outputs.Target_Motion_Type	0
Tolo_Outputs.Digital_Output	0

Figure 22: Making an Absolute Move

5.3 Increment Move

Set the same motion profile parameters as in the Absolute Move example in section 5.2. Change Target_Motion_Type to '1' and keep Move_Select at '0'. Set the position to the desired increment distance; in this example it is 10mm. Make sure Enable is '1'. Now toggle Start_Motion from '0' to '1' to initiate the move.

To perform an Incremental Negative Move, repeat the same procedure using a '2' for Target_Motion_Type.

Tolo_Outputs	{...}
Tolo_Outputs.Drive_Control	{...}
Tolo_Outputs.Drive_Control.Enable	1
Tolo_Outputs.Drive_Control.Start_Motion	1
Tolo_Outputs.Drive_Control.Home	0
Tolo_Outputs.Drive_Control.eStop	0
Tolo_Outputs.Drive_Control.res4	0
Tolo_Outputs.Drive_Control.res5	0
Tolo_Outputs.Drive_Control.res6	0
Tolo_Outputs.Drive_Control.res7	0
Tolo_Outputs.Drive_Control.Move_Select	0
Tolo_Outputs.Target_Position	10.0
Tolo_Outputs.Target_Velocity	25.0
Tolo_Outputs.Target_Acceleration	200.0
Tolo_Outputs.Target_Deceleration	200.0
Tolo_Outputs.Target_Force	100.0
Tolo_Outputs.Target_Motion_Type	1
Tolo_Outputs.Digital_Output	0

Figure 23: Increment Positive Move

5.4 Index Move

An Index Move uses the setting from the move definitions table put into the drive at setup. These move definitions can only be changed using TMI. It is not necessary to set any of the motion parameters in Drive_Control. These parameters are ignored the the drive uses the parameters from the move definition table. Set Enable to '1', and set Move_Select to any value between '1' and 16'. In this example index '4' is selected. Now toggle Start_Motion from '0' to '1' to initiate the move.

Tolo_Outputs	{...}
Tolo_Outputs.Drive_Control	{...}
Tolo_Outputs.Drive_Control.Enable	1
Tolo_Outputs.Drive_Control.Start_Motion	1
Tolo_Outputs.Drive_Control.Home	0
Tolo_Outputs.Drive_Control.eStop	0
Tolo_Outputs.Drive_Control.res4	0
Tolo_Outputs.Drive_Control.res5	0
Tolo_Outputs.Drive_Control.res6	0
Tolo_Outputs.Drive_Control.res7	0
Tolo_Outputs.Drive_Control.Move_Select	4
Tolo_Outputs.Target_Position	0.0
Tolo_Outputs.Target_Velocity	0.0
Tolo_Outputs.Target_Acceleration	0.0
Tolo_Outputs.Target_Deceleration	0.0
Tolo_Outputs.Target_Force	0.0
Tolo_Outputs.Target_Motion_Type	0
Tolo_Outputs.Digital_Output	0

Figure 24: Index Move

5.5 Other Supported Moves

Force_Move: Motion Type = '9'
Increment Move Positive Rotary: Motion Type = '11'
Increment Move Negative Rotary: Motion Type = '12'
Velocity Forward Rotary: Motion Type = '13'
Velocity Reverse Rotary: Motion Type = '14'