

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



# Contents

1	System	Requirements	2
2	Cabling		2
3	ACS Dri	ve IP Address Setup	2
4	Getting	Started in RSLogix 5000	3
	4.1	Add Ethernet Module	3
	4.2	Download Configuration to Controller	6
	4.3	Controller Tags	6
	4.4	Creating Program Tags	8
	4.5	Ladder Logic Instructions	13
5	Make M	lotion	14
	5.1	Home Move	14
	5.2	Absolute Move	15
	5.3	Incremental Move	15
	5.4	Index Move	16



# **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

ACS Internet Protocol (TCP/IP) Properties 💶 🗖 🔀				
Network Settings Dbtain an NP address automatically NP address Subnet Mask 255.255.255.0 Default Gateway 192.168.0.1				
MAC Address 00:04:A3:33:10:ED				
OK Test Cancel				
StaticIPAddress;				

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.'

0	Controller Projects
	Recent Projects
	Open Project
	New Project
	Open Sample 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.

New Controller		×
Vendor: <u>I</u> ype: Re <u>v</u> ision: Name: Description: <u>Chassis Type</u>	Allen-Bradley          1769:L23E-QB1       CompactLogix5323E-QB1       OK         18       Cancel         18       Cancel         Bedunkancy Enabled       Help         ExampleProject       Image: Control of the second sec	
Sl <u>o</u> t:	0 🗘 Safety Partner Slot: <none></none>	_
Ur <u>e</u> ate in:	C:\RSLogix 5000\Projects Browse	

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".

Select Module		
Module	Description	
2097-V34PR5	Kinetix 300, 4A, 480V, No Filter	<u> </u>
2097-V34PR6	Kinetix 300, 6A, 480V, No Filter	_
- 2364F RGU-EN1	Regen Bus Supply via 1203-EN1	
- Drivelogix5730 Etherne	10/100 Mbps Ethernet Port on DriveLogix573	0
ETHERNET-BRIDGE	Generic EtherNet/IR CIP Bridge	
ETHERNET-MODULE	Generic Ethernet Module	
ETHERNET-PANELVIEW	EtherNet/IP Panelview	
EtherNet/IP	SoftLogix5800 EtherNet/IP	
PowerFlex 4 Class Multi	Multi Drive via 22-COMM-E	
PowerFlex 4-E	AC Drive via 22-COMM-E	
PowerFlex 4M-E	AC Drive via 22-COMM-E	
PowerFlex 40-E	AC Drive via 22-COMM-E	
PowerFlex 40P-E	AC Drive via 22-COMM-E	<u>×</u>
		>
	<u></u>	<u>A</u> dd Favorite
By Category By Vend	or Favorites	
	OK Cance	elp

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 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.

New Module					
Type: Vendor: Parent:	ETHERNET-MODULE Generic Ethernet Allen-Bradley LocalENB	Module			$\langle \langle \rangle \rangle$
Na <u>m</u> e: Description:	Tolomatic_ACS_Drive	Lonnection Parameter Asser Instar Input: 100 Output: 113	nbly nce: Size:	(32-bit) 132-bit)	
Comm <u>F</u> ormal Address / F IP <u>A</u> ddre <u>H</u> ost Na	: Data - DINT   Iost Name  ss: 192 . 168 . 0 . 100  me:	Configuration: 1 Status Input: Status Output:		(8-bit)	
🔽 Open Mod	uļe Properties		Candel	Help	
Figure 6: Ne	w Module Properties Window	$\left( \right)$			

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.

Module Properties: LocalENB (ETHERNET-MODULE 1,1)
General       Connection*       Module Info         Hequested Packet Interval (BPI):       50.0       ms       (1.0 - 3200.0 ms)         Inhibit Module       Major Fault On Controller If Connection Fails While in Run Mode         Use Unidast Connection over EtherNet/IP         Module Facil
Status: Offline OK Cancel Apply Help

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:0:0



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.

	🖉 Controller Tags - ExampleProject(controller)						
S	cope: 🛐 ExampleProject 🔽 Show: All Tags	V. Enter Lame Fret					
	Name <u>18</u> 4	Data Type Description 🗠 🚗					
	⊞-Local:1:C	AB:Embedded_J@16F:C:0					
	⊞-Local:1:I	AB:Embedded_IQ16FvI:0					
	⊞-Local:2:C	AB:Embedded_QB16:C:0					
	⊞-Local:2:I	AB:Embedded_OB16i:0					
	⊞-Local:2:0	AB:Embedded_0816:0:0					
	⊞-Tolomatic_ACS_Drive:C	AB:ETHERNET_MODULE:C:0					
	⊞-Tolomatic_ACS_Drive:I	AB:ETHERNET_MODULE_DINT_288ptes:1:0					
	⊞-Tolomatic_ACS_Drive:0	AB:ETHERNET_MODULE_DINT_32Bytes:0:0					
•	Monitor Tags / Edit Tags / Development Page / Devel						

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 - Z	Value 🗲	Force Mask 🛛 🗲	Style
Tolomatic_AC6_Drive:0	{}	{}	
-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.

🚟 Data Type: Driv	e_Faults		ILA	
Name:	Drive Faults			$\mathbf{i}$
Description:	Di	ive Fault		
	pa	arameters		
		$\left( \cap \right)$		
			$\setminus$	
Marchan		Data Taka Cina Alta	-6	
Members:		Uata Type Size: 4 byt	e(s)	
Name	Data Type	Style Descri	ption External Acces	s 🔼
Positive_Limit	BOOL	Decimal	Read/Write	
Negative_Limit	BOOL	Decimal	Read/Write	
eStop	BOOL	Decimal	Read/Write	
Position Error	BOOL	Decimal	Read/Write	
Feedback_Bro	pr BOOL	Decimal	Read/Write	
Over_Temp	BOOL	Decimal	Read/Write	
Motor_Overten	np BOOL	Decimal	Read/Write	
Drive_Overten	BOOL	Decimal	Read/Write	
Drive Overvol	tage BOOL	Decimal	Read/Write	
Drive_Undervo	oltage BOOL	Decimal	Read/Write	
Elash_Error	BØOL	Decimal	Read/Write	~
Move Up Mov	e Down	OK (	Cancel Apply	Help

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 Ty	vpe 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	$\overline{)}$	Read/Write	
	rsved_bit12	BOOL	Decimal		Read/Write	
	rsved_bit13	BOOL	Decimal	$\overline{)}$	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.



101	)ata Type: Tolo	_Inputs					
Na	ame:	Tolo_Inpu	uts				
De	escription:		Tolomatic A Inpu	ACS Drive uts			
Me	embers:			Data Type Size	e: 28 byte(s)		
	Name Current Positio	n	Data Type BFAI	Eloat	Description	External Access Bead Write	
	⊡ Drive_Status		Drive_Status			Read/Write	
	 ⊞ Drive_Faults		Drive_Faults			Read/Write	
	Digital_Input		DINT	Binary	8 bits used of 32	Read/Write	
	Digital_Output		DINT	Binary	4 bits used of 32	Read/Write	
	Analog_Input		REAL	Float		Read/Write	
	Analog_Output		REAL	Float		Read/Write	
010				()			
	Move Up Move	e Down	$\square$	ОК	Cancel	Apply Hel	p

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.



101 D	ata Type: Netv	vork_Outputs					
Na	ime:	Network Outputs			/		
							$\wedge$
De	escription:	Toloma C	tic ACS Drive Dutputs				$\langle \rangle$
							~ /
					$\sim \sim \sim$		
							$\mathbf{v}$
Me	embers:		Data Type Size	e: 4 byte(s)	$\wedge$		
	Name	Data Type	Style	Description	External Access		
	Enable	BOOL	Decimal		Read/Write		
	Start_Motion	BOOL	Decimal		Read/Write		
	Home	BOOL	Decimal		ReadXWrite		
	eStop	BOOL	Decimal		Read/Write	=	
	res4	BOOL	Decimal		Read/Write		
	res5	BOOL	Decimal		Read/Write		
	res6	BOOL	Decimal		Read/Write		
	res7	BOOL	Decimal		Read/Write		
	Move_Select	SINT	Decimal		Read/Write		
10f <sup>4</sup> 010						<b>~</b>	
	Nove Up Mov	e Down	ОК	Cancel	Apply He	lp 📄	

Figure 15: Creating a Network\_Output Data Type





Next, create the 'Tolo\_Outputs' data type as shown.

🛗 Data Type: Tolo	_Output	s					$\land$
Name:	Tolo_Outp	puts					
Description:		Tolomatic AC	S Drive	~			$\land \land \lor$
		output	.5				
				<b>~</b>			
						$\langle \rangle$	
						$\bigvee$	
Members:			Data Type Siz	ze: 32 byte(s)			
Name		Data Type	Style	Description	External Access		
Drive_Control		Network_Outputs		4 bits used of 8	Read/Write		
Target_Positio	n	REAL	Float		Read/Write		
Target_Veloci	by .	REAL	Float		Read/Write		
Target_Acceld	eration	REAL	Float		Read/Write		
Target_Decel	eration	REAL	Float		Read/Write		
Target_Force		REAL	Float		Read/Write		*
Target_Motion	n_Type	SINT	Decimal	Absolute or Incremental	Read/Write		
Digital_Output	t	DINT	Decimal	4 bits used of 32	ReadAWrite		
10 <sup>4</sup>						Z >	
]				01			
Move Up Mov	/e Down		OK	Cancel	Apply H	elp	
							1

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.

					_
Controller Tags - ExampleProject(controlle	ar)				×
Scope: DExample roject Show: All Tags		🔽 🔽. Enter Nai	ne Filter		~
Name III Alias For	Base Tag	Data Type	Description	E ste	
+-Local1:C		AB:Embedded_IQ		Rea	P
+:Local1:1		AB:Embedded_IQ		Rea	ropo
+ Locat2:C		AB:Embedded_0		Rea	Intie
E-Lopat21		AB:Embedded_0		Rea	65
± Locat20		AB:Embedded_0		Rea	-
		Tolo_Outputs	Tolomatic ACS Dri	Rea	
		Tolo_Inputs	Tolomatic ACS Dri	Rea	
+ Tolomatic_ACS_Drive:C		AB:ETHERNET		Rea	
Tolomatic_AC9_Drive:l		AB:ETHERNET		Rea	
+ Tolomatic_ACS_Drive:D		AB:ETHERNET		Rea	
2					
				~	
▲ ► \ Monitor Tags \ Edit Tags /	<	ш		>	

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.

🗐 📇 Tasks	ė.
😑 🤯 MainTask	
😑 🚭 MainProgram	
🦳 📝 Program Tags	
🔂 MainRoutine	
🗀 Unscheduled Programs	

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
	Ø

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
	0
	7/1
Figure 24: Index Move	
5.5 Other Supported Moves	
Force_Move:	Notion type $=$
Increment Move Positive Botary	Motion Type = '
indicitional word in control flotary.	moutin type -
Increment Move Negative Rotary:	Motion Type = '
Valasty Farward Patary	Motion Tune
velocity Forward Rotary:	wouldn type = $\cdot$
Velocity Reverse Botary	Motion Type $=$ '
volocity notorios notary.	