Connected Component Workbench

Connected Component Workbench is the software used to program Allen Bradley Mico-810 PLC's. Copies can be obtained fom the following link:

http://www.rockwellautomation.com/global/products-technologies/connected-components/tools/workbench.page Note that if you are using a Windows XP machine, you need an older version from a CD ROM. Please see the instructor if this is you.

Notation

Inputs:

- Connect COM-0 to -DC24 (0V) to enable inputs 0..3
- Connect -DC24 to -DC24 to enable inputs 4..7
- Inputs 0..3 are digital (0V = logic 0, 24V = logic 1)
- Inputs 4..7 and digital (24V) or analog (0..10V), depending upon the input voltage

Constant Inputs:

Time:

- T#300x
- d = day
- h = hour
- m = minutes
- s = seconds
- ms = milliseconds

Binary: 2#

• 2#1001_0010 binary number 0x52

Hex: 16#

Decimal: 10#

Getting Started:

Start Connected Components Wizard. You should get the following screen

😫 Connected Components Wo	orkbench	
File Edit View Debug Took	s Communications Window Help	
🔁 📸 - 🖬 🐰 🖻 🛍	ヴ・ヴ・厚・思 💁 🔰 🔡 🖌 🖬 🖉 🦉 📳 🐘	🎚 🎲 🏫 🧼 💋 💋 Cycle Timing (ms): 🔹 👻
HMI Application Language:	·	Q Q .
Project Organizer 🛛 👻 🖡 🗙		Device Toolbox 👻 🕂 🗙
Name: Project11		Discover
Choose a device from the Device Toolbox using Discover or by dragging it from the		🖻 Catalog
Latalog.		2080-LC10-12DWD
		2080-LC10-12Q88
		2080-LC10-12QWB
		2080-LC30-10QVB
		2080-LC30-10QWB
		Toolbox 👻 🕂 🗙
		😑 General
		There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
	Output 🗸 🖡 🗡	
	Show output from:	
Ready		

Select the PLC we're using: 2080-LC10-12QWB, and draag it left to the Program window. Double click on Programs and select New Ladder Diagram:

Co 🖸	nnecte	d Compone	ents Wor	kbenc	h		
File	Edit	View Build	Debug	Tools	Communications	Window	Help
1	2	- 🛃 🐰	68	19 - 1	(*		
HMI A	pplicatio	on Language:	1			-	#
Project	Organiz	ter 🖣	ųΧ				
Name:	Project	12*					
	Micro81	0					
		Add	•	New 5	5T : Structured Tex	t	Т
	6	B Paste	00	New I	.D : Ladder Diagran	n	
		Propertie	s 🚺	New F	BD : Function Bloc	k Diagram	
	- -	User-Defined	Function				1

This should give you the following display: an empty program

Connected Components Workb	anch
File Edit View Build Debug To	ols Communications Window Help
🔁 😅 🗉 - 🖬 🔺 🖓 🖄 🖄	・0・四・四 10 10 10 10 10 10 10 10 10 10 10 10 10
HITLADORISTICAL LAND HIDLE	
B	Z U ち 人 唐 専 連 注 。
Project Organizer 🔹 🖗 🗙 👘	UntRiedLD-POU • ×
More Property Morosio Programs 	1

Program elements are shown in the lower right corner: these are drag and drop elements:



• Rung: Add a new line to the program

Outputs: These must be placed on the far right of the ladder diagram

- Direct Coil: Boolean variable. When energised, the variable becomes true. If the variable is an output relay, the relay closes. When de-energised, the coil opens. Can also be an internal state.
- Reverse Coil: Booleam vairable. Energised is open (true), de-energised is closed (false).
- Set Coil: When energised, the coil closes and remains closed.
- Reset Coil: When energised, the coil opens and remains open.
- Pulse Rising Edge Coil: The coil closes momentarilly when the output is energised (rising edge)
- Pulse Falling Edge Coil: The coil opens momentarilly when the output is de-energised (falling edge)

Inputs: These must be placed on the left side of the ladder diagram

- Direct Contact: True means the switch closes, false means the switch is open.
- Reverse Contact: Closed when false, open when true.
- Pulse Rising Edge Contact: Momentarilly closes when the signal goes high (rising edge)
- Pules Falling Edge Contact: Momentarilly closes when the signal goes low (falling edge)

Blocks:

• 100+ operations, such as a time delay. More on this later.

Example Code:

Write a prorgram so that the red LED turns on whenever you press button 0.

Function:

 $\mathbf{Y} = \mathbf{A}$

Inputs:

A = IN0

Output:

 $\mathbf{Y} = \mathbf{OUT0}$

Program:

Click on the Input Coil icon and drag it to the left of the ladder diagram. To define this input as IN0, double click on the coil and select I/O Micro 810

🕃 Connected Components Workbench	P Variable Selector	
File Edit Wew Build Debug Tools Communications Window Help	Name ID_EM_DI_DO Type Global Scope Micro810 Micro810	Local Scope N/A
Project Organizer + 4 × UntitledLD-POU	User clobal valiables - Micrositu Local valiables - N/A System valiables - Micrositu Vol - Micrositu Delined	Vords - Microol U
Name: Project12*	Name Data Type Dimension Alias	Initial Value
MicroB10 Programs UnRited.D Gobal Variables	off BOOL - - off JO_EM_DO_00 BOOL - - off JO_EM_DO_01 BOOL - - - JO_EM_DO_02 BOOL - - - JO_EM_DO_03 BOOL - - - JO_EM_DI_03 BOOL - - - JO_EM_DI_01 BOOL - - - JO_EM_DI_02 BOOL - - - JO_EM_DI_03 BOOL - - - JO_EM_DI_04 BOOL - - -	r pfr - ofr Read Read Read Read Read Read Read Read
DataTypes	_I0_EM_DL_05 BOOL - _I0_EM_DL_06 BOOL - _I0_EM_DL_07 BOOL -	Read Read Read

This lets you assign a coil to the input pins and output relays:

- _IO_EM_DO_0x Digital Output #x (relay output x)
- _IO_EM_DI_0x Digital Input #x (input pin x)

Select Digital Input 0 for pin #0. You can give it an Alian (name) if you like.

Click on the Output Coil icon and drag it to the right side of the display and assign it to Digital Output 0.

Name _IO_EM_DO_00	Global Scope Micro810		V Local	Local Scope N/A				
Jser Global Variables - Micro810 Local	Variables	• N/A Sys	tem Variable	s - Micro810	1/0 - Micro810	Defined Words - I	Micro810	
Name		Data	Туре	Dimension	2	Alias	Initial Value	
Name	- ₀ e*	Data BOOL	Гуре	Dimension		Alias • Al	Initial Value	
Name _IO_EM_DO_00	- dt+	Data BOOL BOOL	Type - 22	Dimension	Red Light	Alias * dl	InitialValue • A	Read/
	* d*	Data BOOL BOOL BOOL	Type 22 v	Dimension	Red Light	Alias • dt	Initial Value	Read/ Read/
✓ _I0_EM_D0_00 _I0_EM_D0_01 _I0_EM_D0_02	- A+	Data BOOL BOOL BOOL BOOL	Type 22 2	Dimension	Red Light	Alias • dt	Initial Value	Read/ Read/ Read/
Name .// _IO_EM_DO_00 _IO_EM_DO_01 _IO_EM_DO_02 _IO_EM_DO_03	* dł*	Data BOOL BOOL BOOL BOOL BOOL	Type - 22 - - -	Dimension	Red Light	Alias • Al	InitialValue	Read/ Read/ Read/ Read/

Your program should then look like the following:

🕄 Connected Components Workber	nch			
File Edit View Build Debug Tool	ls Communications Window Help			
🔁 📸 - 🔜 👗 🛍 🛍 🔊 -	· (* - 🖳 - 🖳 🔒	🔹 🖀 🏷 🖕 🛅 🕨 💷 🗇	91 (I 99 🗞 99 II I 68 6	h 🏟 🖉 💋 🛛 Cycle
HMI Application Language:		ᆃ 아 아 출 미 <mark>Build</mark> 및 먼	🔍 Q 🖕	
· · · B /	″ ⊻ 卷 ▲ = = = = := .			
Project Organizer 🗸 🗸 🗸 🗸	ntitledLD-POU*			▼ ×
Name: Project12*	1			
Micro810				
	_IO_EM_DI_00 Light On			_IO_EM_DO_00 Red Light
				O
Local Va				

Compilation:

Click on the Build icon. This translates the ladder diagram to machine code. With any luck. you'll get a successful message:

Program:

Connect the Micro-810 to a PC through the USB port. When you click on the Download icon, you'll get a message asking which device. Select the USB port and Micro810

kbench		
Tools Communications Window	Help	
り・ロ・思・思 🔒	- 2 2	BUU ON
	Connection Browser 📃 🗖 🔀	
UntitledLD-POU	Autobrowse Refresh Workstation, SURPLUS-4F58315 Refresh Ref	JO_EM_DO_00 Red Light

It will then ask if you want to halt the current program:



say yes. About 30 seconds later, after the program is downloaded, it will ask if you want to restart the (new) program. Say yes.



At this point, the program should be running: when you press button 0, the red light turns on.

Example 2: Two On-Off Switches

Write a program with the following functionality:

- IN0: Turn on the red LED (out 0)
- IN1: Turn off the red LED
- IN2: Turn on the yellow LED (out 1)
- IN3: Turn off the yellow LED

Create four rungs by dragging the rung icon left. Add four input coils and four output coils as follows:



Example 3: Logical AND and OR

Write a program which

- Turns on the red LED when button 0 and 1 are pressed, and
- Turns on the yellow LED when buttons 2 or 3 are pressed.

Program: To create the OR function, drag the Branch inco left and place in in the second rung. Add to Direct Contacts for input 2 and 3.

Note that

- Two switches in series is an AND function
- Two switches in parallel is an OR function



Combinational Logic on a PLC

From ECE 275, if you can implement AND, OR, and NOT functions, you can implement any truth table. One way to do this is using Karnough maps.

Example: Write a program which

- Turns off all LEDs when no buttons are pressed
- Turns on the blue LED (DO3) when one or more buttons are pushed
- Turns on the green LED (DO2) when two or more buttons are pressed, and
- Turns on the yellow LED (DO1) when three or more buttons are pressed

Assume only buttons 0 / 1 / 2 are used and defined as

- A = input 0
- B = input 1
- C = input 2

First, set up a Karnough map for each output:



Next, circle the ones in groups of 2n (shown in red above). This gives

Blue = A + B + CGreen = AB + AC + BCYellow = ABC

Implement this in Ladder Logic. In Connected Component Workbench, the code will look like the following:



Note that there are other solutions. For example, you could create a local variable for when one, two, or three buttons are pressed - with the following logic for each:

 $One = A\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C$ $Two = AB\overline{C} + A\overline{B}C + \overline{A}BC$ Three = ABC

• The blue LED (one or more) is on if any of these three are true

- The green LED (two or more) is on if Two or Three is true
- The yellow LED is on if three is true

Create a local variable

🔡 Variable Selector						
Name	Type BOOL	~	Glob	al Scope o830	Local Scope UntitledLD	~
User Global Variables - Micro830	Local Variables	- UntitledLD	System V	ariables - Microl	330 1/O - Micro830 Defined	Words - Mi 🔹
Name		Data T	уре	Dimension	Alias	Initial Valu
	- A*		× 22	- A*	- 6	#* - {
One_LED		BOOL	*			
Two_LEDs]	BOOL	τ.			
Three_LEDs		BOOL	*			
*			*			

Then use this local variable along with logic for one, two, or three buttons being pressed

UntitledLD-PO	n. Teorines periods	
1		
		0
	10.54,000 10.54,000 10.54,002	One_LED
2		
	าด"ยา้ย"ดา าด"ยา้ย"ดา	Two_LEDs
		$- \circ$
	J0_BM_DI_00 _J0_BM_DI_01 _J0_BM_DI_02	
3		
	JO. EM. DL 00 J. EM. DL 01 J. 0. EM. DL 02	Three LEOs
4		Ŭ
	One LED	10 EM DO 43
	Une_ceu	Bue
	Two_LEDs	U
	Three_LEDs	
L		
		Two_LEDs
2		
3		
	\$2,0,48,0L 10,18,48,0L 10,18,48,0L 10,10,48,0L	Three_LEDs
4		
	Ore_LED	_IO_EM_DO_03
		—Õ—
	100_1209	
	Three_LEDs	
5		
	Two_LEOs	_IO_EM_DO_02
	Three_LEDs	
6		
3		
	Three_LEDs	_IO_EM_DO_01 Yellow
		<u> </u>

Debug Feature:

Once you download your code, click the Debug button (F5)



This will then show you on the screen which lines of code are turned on as you press the buttons. It helps understand what is going on (and we'll use this feature a lot).

For example, when you press A and B, the display under Debug mode looks like the following:



Red shows which paths are turned on (true)