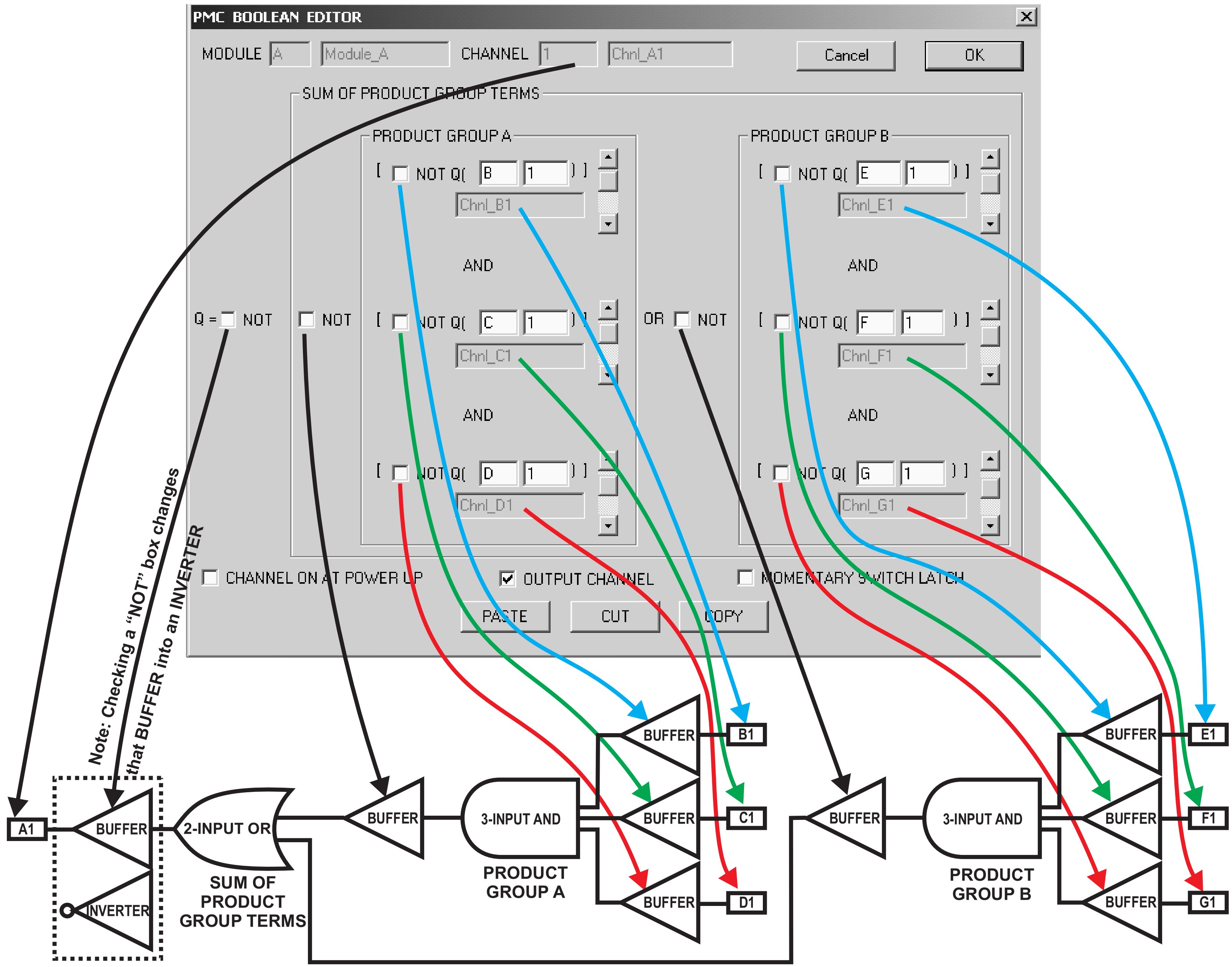
131 Eisenhower Ln. N.
Lombard, IL 60148

Chapter 5

Set UP Logic
Using PMC Software



SCHEMATIC LOGIC SYMBOL OF THE PMC BOOLEAN EDITOR SCREEN

Set Up Logic Using PMC Software

BOOLEANS

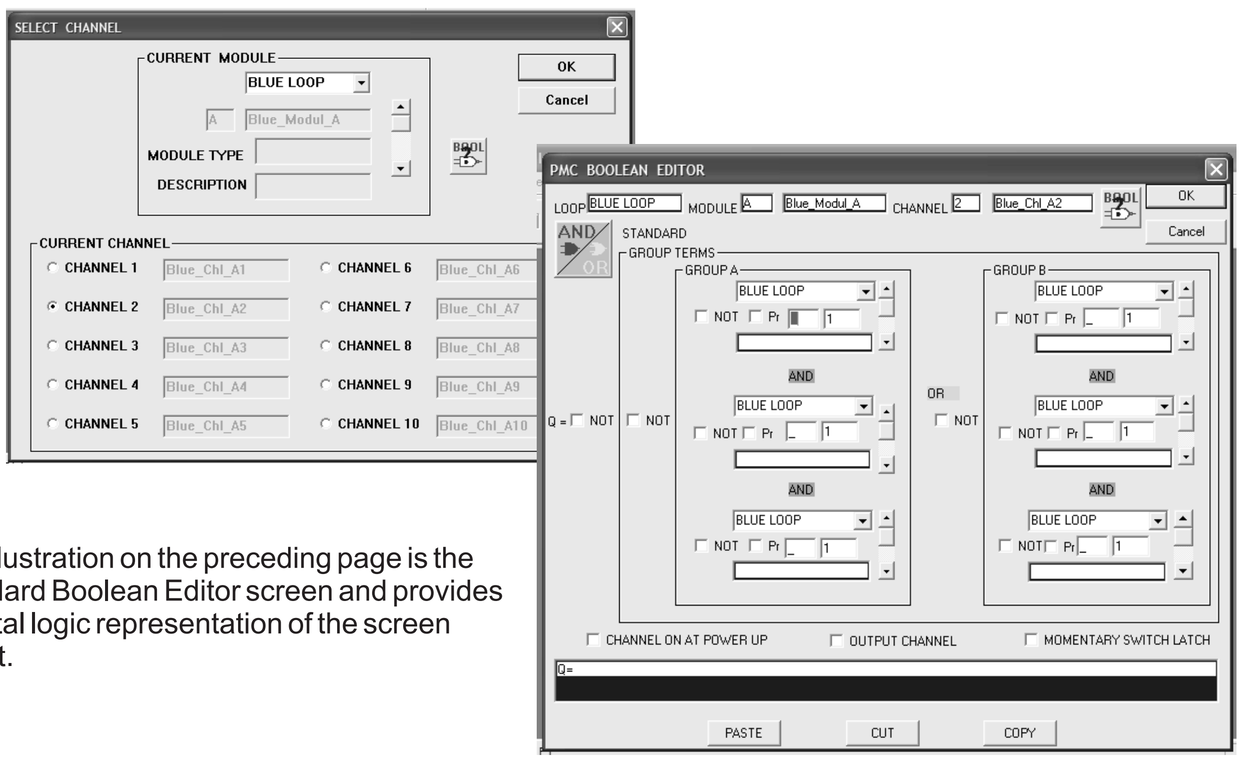
Once the module and channel labels are fully entered and saved, the actual programming can begin. In this step, you will enter the relationships between the inputs and outputs. This step is done through Boolean algebra expressions. ***Only outputs*** *are programmed.* **Boolean expressions must never be written for input channels.** Booleans are expressed as functions of the inputs. When writing a program, we should always ask ourselves “What turns the output on?” Never ask “What does the switch do?” This may seem to be the same thing, however; it will be helpful to you if this way of thinking is followed.

To begin, click on "PMC Setup" in the task bar to produce the flyout. Click on "Booleans". Another flyout will appear. Click on "Editor”, or simply click on the red Bool icon on the task bar. 

Using the up/down arrows, select the output module containing the output you wish to program. The Select Channel screen will come up. Select the channel you want to program by clicking in the circle in front of the channel you want to program. Then, click on the "OK" box.

The PMC Boolean Editor screen will come up. This screen is the area in which the actual relationship between inputs and the output will be determined through the Boolean equation.

The screens look like this:



The illustration on the preceding page is the Standard Boolean Editor screen and provides a digital logic representation of the screen layout.

This general set up allows you to write an expression that includes a three input AND, OR'ed with another three input AND. Using DeMorgans theorem this single screen can also be used to write a 6 input OR, or a 6 input AND *(See Chapter 9 for examples and the Chapters on Boolean Algebra for further explanation).* The screen shown above is from WinPMC II and the 320 channel CPU. You will note the AND/OR Icon. 

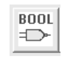
By repeatedly clicking this Icon, the screen can be changed to a 6 input AND, 6 input OR, 3 input OR AND’ed with a 3 input OR. With the 160 channel system, only the 3 input AND, Or’ed with a 3 input AND is available. DeMorgans theorem must be used to write a 6 input AND or OR.

Set Up Logic Using PMC Software

The simplest equation that can be written, is one in which the output is a direct function of the input; such as, the light goes on when the switch is turned on. Let's put the output or light on channel BA1 and the input or switch on channel YC1. The equation for this function is written as:

Light = Switch
BA1 = YC1

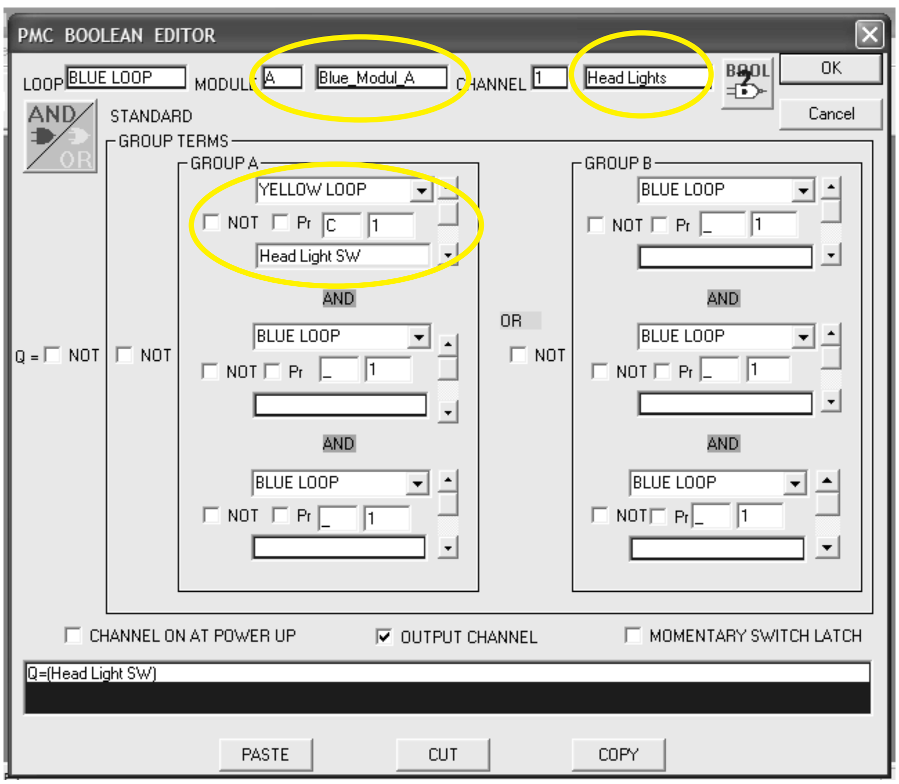
B = Blue Loop Y = Yellow Loop.
Note that the160 channel CPU has only one loop, it's addresses do not include a B or Y.
With this equation, the output to the light on channel BA1 will go on when the switch YC1 goes on.

To input this equation, click on the red Bool icon in the task bar. 

Select the loop and module whose output you want to program; in this case, "Blue Loop and Module A", clicking the up or down boxes until the loop and module you want is shown in the box. *Remember, <u>only</u> the outputs need to be programmed.*

Next, select the channel you want to program; in this case, "channel 1", by clicking in the small circle in front of the desired channel. Then click on "OK". The Boolean Editor screen will come up, with Module "A" and it's name and "Channel A1" and it's name will appear at the top of the screen.

This screen is the general input screen. It is a three input AND, OR'ed with another three input AND.



There are "NOT boxes" next to each entry at the beginning of the expression and between the three input AND's. Checking these boxes inverts that term in the expression. For example, the screen above says, that when the light switch YC1 is on, the light output BA1 will be on. If the "NOT" box were to be checked it would change the operation so that when the switch is off, the light is on.

At the bottom of the screen, there are a few more useful tools. The first small box, labeled "CHANNEL, ON AT POWER UP", is used to set the output of a channel to ON, when the power is first applied.

Set Up Logic Using PMC Software

The second box, labeled "OUTPUT CHANNEL" indicates to the system that this channel is an output, as opposed to an input channel. **This box should never be checked for an input channel.** If you write a boolean and forget to check the box, the system will remind you.

The third small box, labeled "MOMENTARY SWITCH LATCH", indicates to the system that this is a momentary switch input channel. **This is the *ONLY* time that the Boolean Editor screen will be used for an input channel.** This is useful for systems requiring momentary switch inputs. The system will respond as though the pushbutton is a toggle switch, by latching the switch on and off alternately on each press of the switch. **Again, this is the *ONLY* entry that is made for an input channel.** Selecting this for a switch input makes the switch act as a push/on- push/off switch. Depending upon the requirements of the system you may wish to write a boolean that will latch an output on, instead of latching the switch on.
We will discuss latching outputs later in this chapter.

The "PASTE, CUT, and COPY" commands are the same as used in other WINDOWS™ applications. They can be used to program multiple output channels with the same or similar inputs.

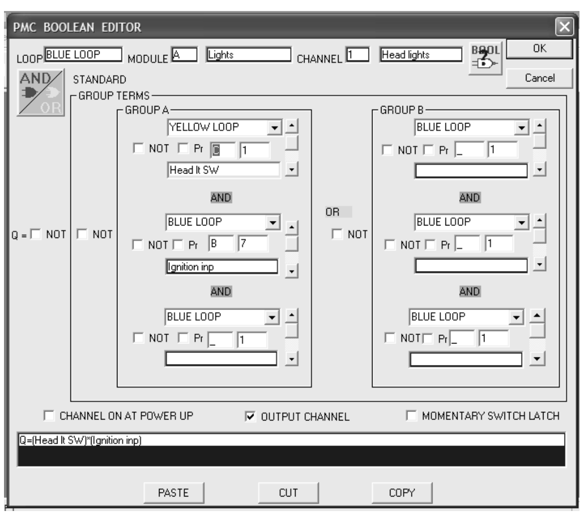
EXAMPLE

Lets look at an example. Suppose that you would like to turn the head lights on, but only when the ignition is on. If the headlights are located on output channel BA1, the head light switch on input channel YC1 and an ignition input on channel B B7, we would write the following Boolean.

Headlight = Headlight switch AND Ignition
BA1=YC1 AND BB7, or in short hand BA1=YC1*BB7

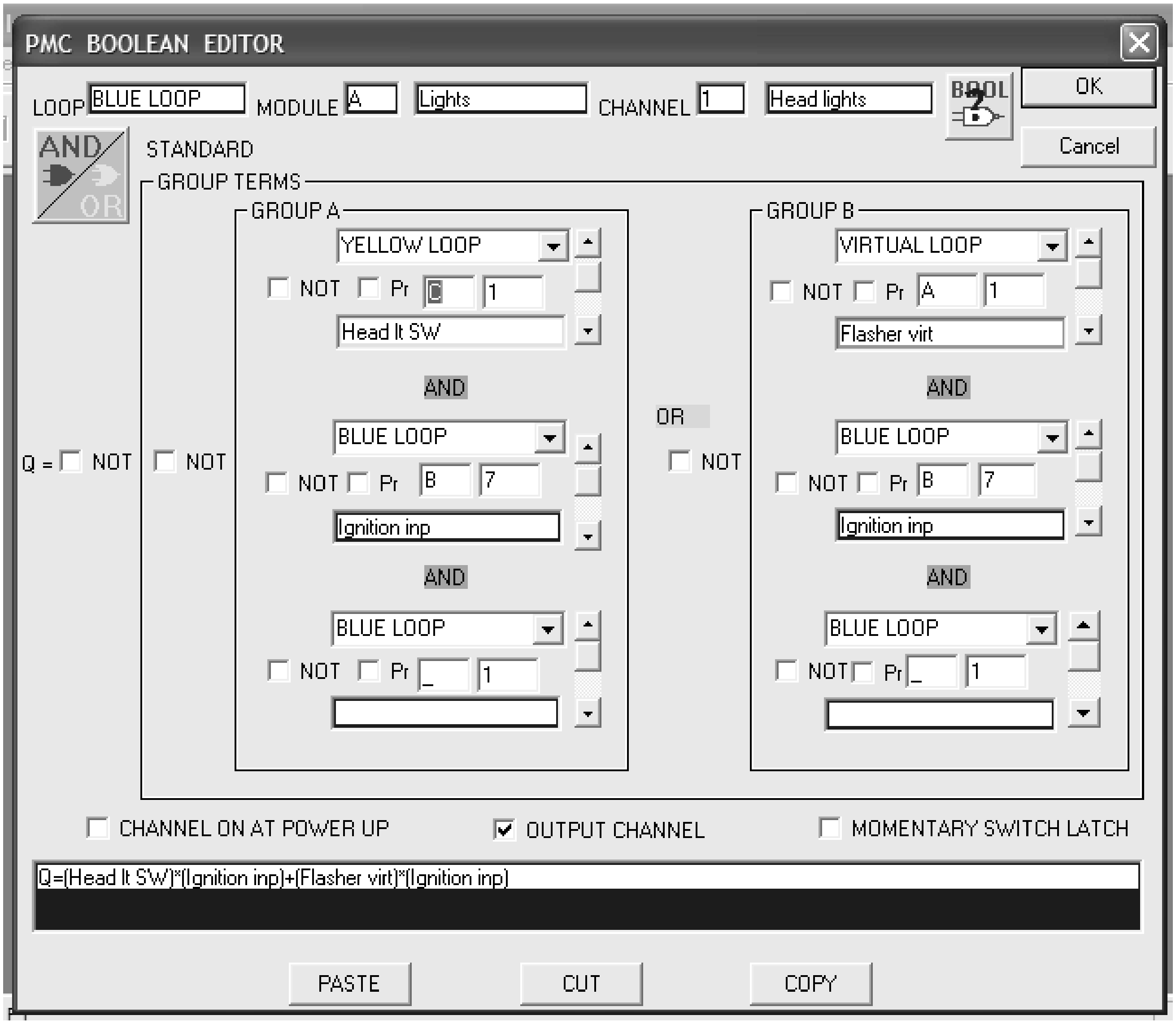
Let's see how we can use this screen to input the data of our example. Click on the Bool icon and navigate to the blue loop channel A1. In the top line of Product Group A, select yellow loop and the input C1, by clicking on the up or down boxes on the slide bar. The Tab key will step you through to the next box. In the second line of product group A, select the blue loop and B7, then click on the "Output Channel" at the bottom of the screen to designate this channel to be an output. If you forget to click on output channel, the program will ask you if you want it to be an output after you click on OK. When asked click "yes".

The screen will look like this:



As you write Boolean equations, remember that it is acceptable to set one output equal to another. It is not necessary to have a switch input turn an output on. For example, you may wish to have the tail lights on whenever the headlights are on. If the headlights are located on output channel A1 and the tail lights are on output channel D1, you could write the following statement: **D1 = A1.**

Set Up Logic Using PMC Software



THE EXAMPLE ABOVE SAYS

Headlights are ON, when the headlight switch and ignition is ON, or when the flash virtual channel is on and ignition is on.

Other ways to write this are:

Headlight = headlight switch AND Ignition OR flasher virt AND ignition
or
BA1=YC1*BB7+VA1*BB7

Assuming that virtual channel VA1 is programmed to flash, the headlight will flash when ignition is on and will burn steady when ignition is on and the headlight switch is on.

Set Up Logic Using PMC Software

VIRTUAL MODULES

On the Boolean Editor screen, the three input AND, OR'ed with another three input AND, allows a lot of flexibility in creating expressions for functions of the system. The system is not limited to just these equations. Any channel can be used as an input for an expression. The system includes virtual channels at address Q for the 160 channel system. On the 320 channel CPU, there are 160 virtual channels in the virtual loop. These channels exists only in the software. In other words, there is no physical module with the address of Q, or on the virtual loop.

As in the example on the previous page, an expression can be written in a virtual channel. Then the virtual channels output can be used as an input expression for another channel. This allows the "stacking" of expressions to provide almost limitless factors in the expression. In addition to the use of the virtual channels, any nonexistent hardware module can be used as a virtual module. Any unused channel of an existing module can be used the same way. For example, every module address has 10 channels. If you were using a six rocker switch adapter, there would be 4 unused channels available to be used as virtual channels. This is most important when using the 160 channel CPU, since it only has 10 virtual channels in the Q module.

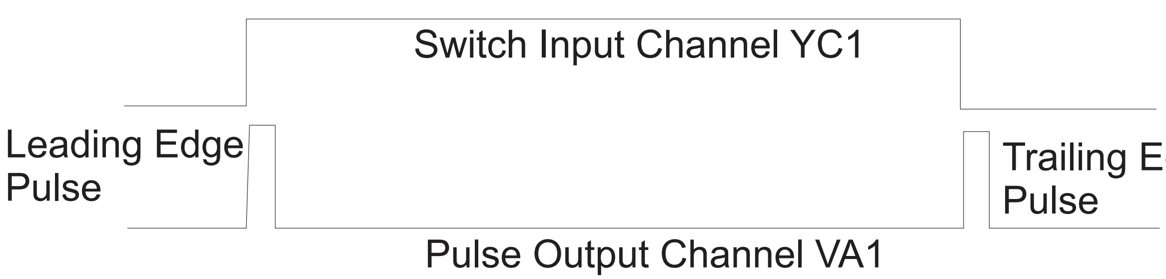
PRIOR STATE- EDGE DETECTION

You may have noticed check boxes on the Boolean Editor screen labeled PR. Prior State is most often used to create a pulse on another channel when a switch is first operated, or released. This is often referred to as edge detection. *Refer to the picture on the previous page and the description of the multiplex signal. As previously discussed, the multiplex signal consists of a sync pulse, followed by 160 clock pulses.* This is a cycle that occurs every .040 seconds. The CPU also evaluates the position of every input and output during this cycle. When the prior state box is checked, the CPU remembers the state of the channel in the prior cycle and compares it with the present cycle. Edge detection is most often used in Booleans to latch outputs on and off. Examples of edge detection and latched outputs follow.

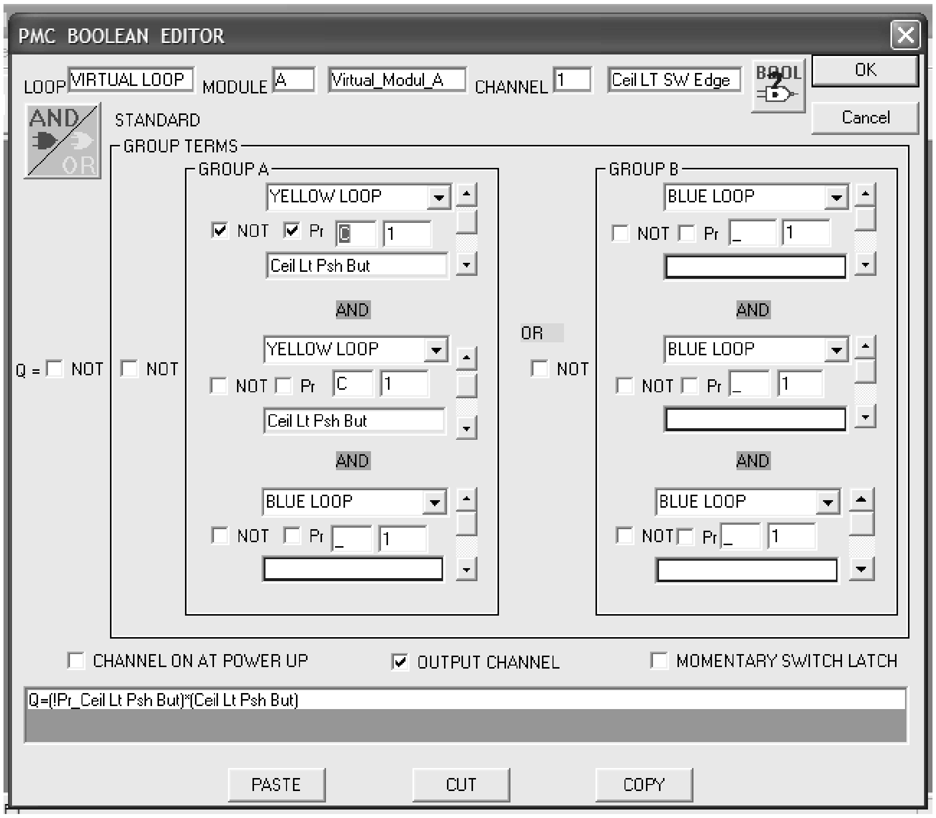
EDGE DETECTION 320 CHANNEL CPU

The purpose of edge detection is to create a short pulse on a channel when a switch is opened or closed. *If the pulse occurs when the switch is closed, we refer to this as the* **leading edge**. *If the pulse occurs when the switch is opened, we refer to this as the* **trailing edge**. The pulse channel will be on for only 0.040 seconds until the boolean for the pulse channel is evaluated again.

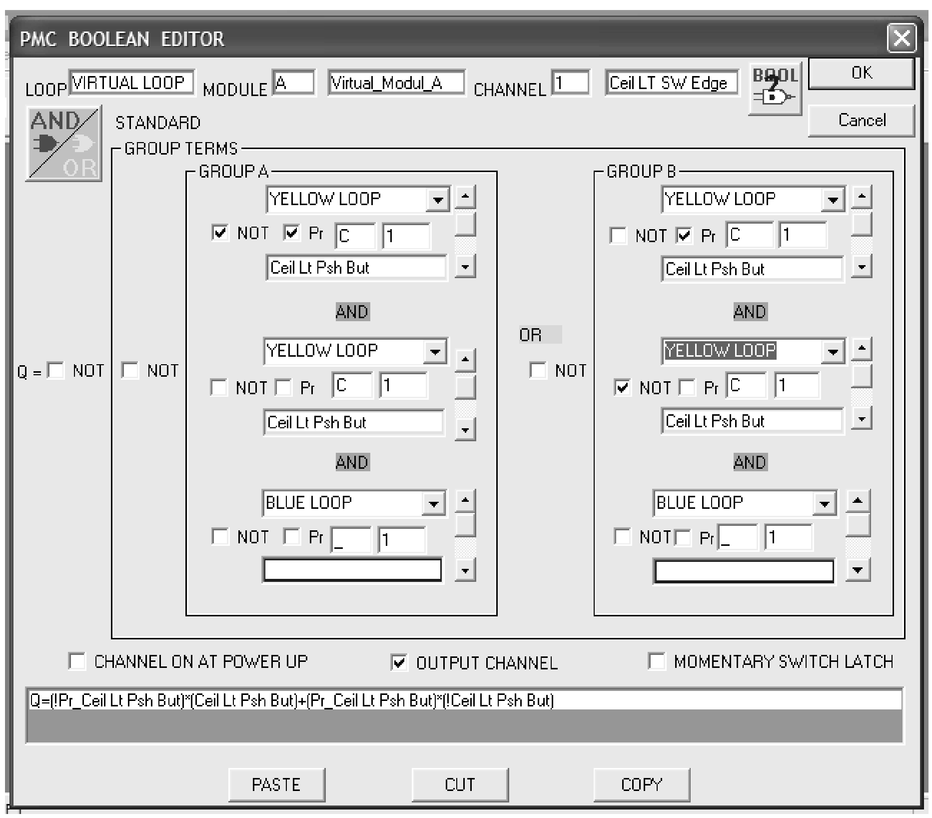
EDGE DETECTION 160 CHANNEL CPU *(See page 9-19)*



Creating an Edge Pulse



This Boolean will create a **leading edge** pulse, on virtual channel VA1, when the switch YC1 is pressed.



The Boolean above will create a **leading edge** pulse, on virtual channel VA1, when the switch is pressed and a **trailing edge** pulse when the switch is released.

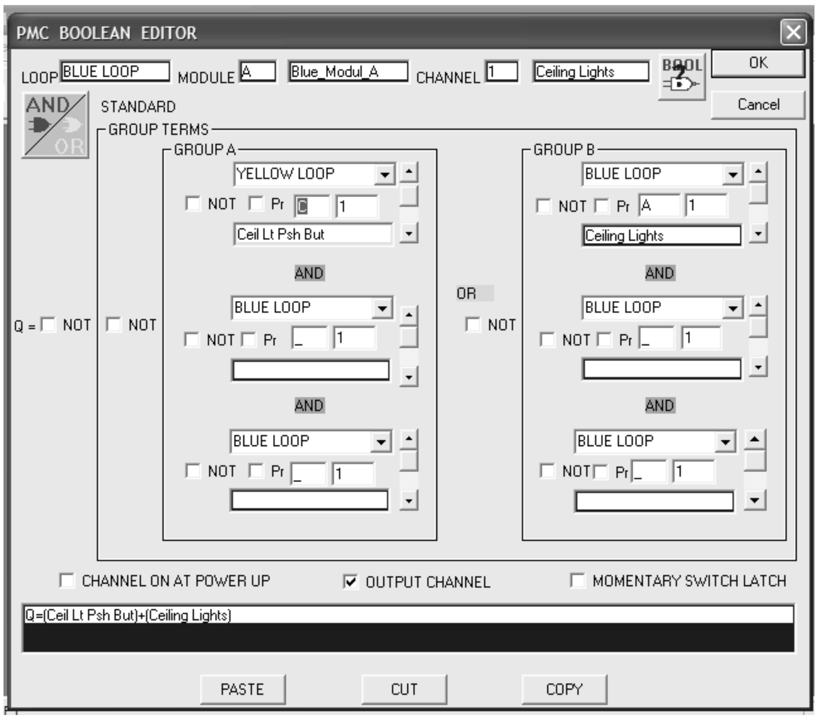
To create a single pulse that occurs only when the switch is released, enter a boolean as shown in Group B, to the right on the screen.

Latching an Output

To latch an output, simply set the output equal to itself.

For example: BA1 = YC1 OR BA1

BA1 turns on when the switch YC1 is on. Since the program says OR BA1, it keeps itself on. With this configuration, BA1 will turn on and will not turn off unless you turn power off to the CPU.



To latch an output on, simply set the output equal to itself.

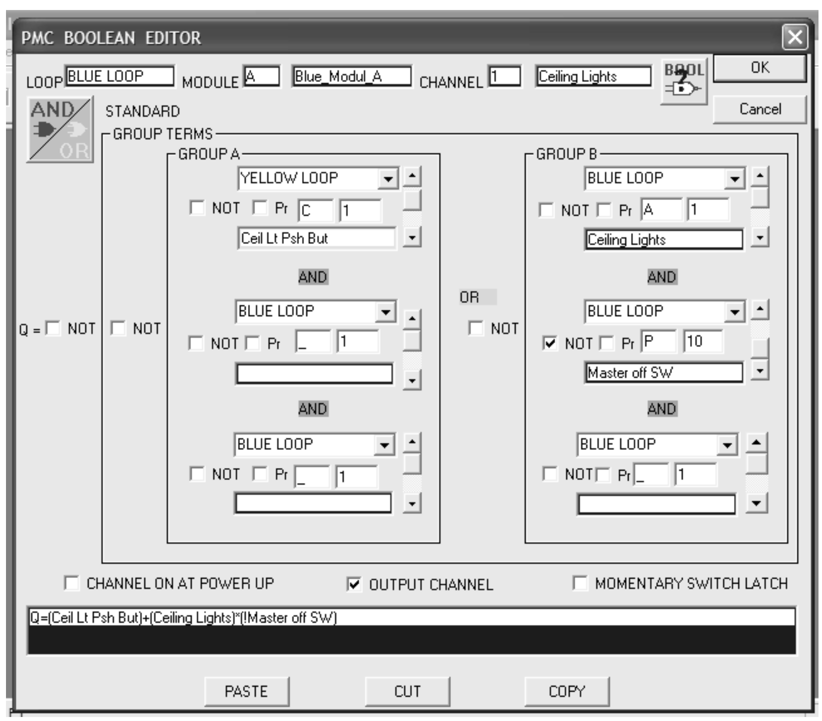
For example:
BA1=YC1 OR BA1

BA1 turns on when the switch YC1 is on. Since the program says OR BA1, once it has turned on, it will keep itself on.

With this configuration BA1 will turn on and will not turn off unless you turn power off to the CPU.

In the the Boolean Editor screen to the right, BA1 will latch on when YC1 turns on. Turning YC1 off will have no effect. You will notice that in Group B, on the right side of the screen, it says BA1 AND Not BP10.

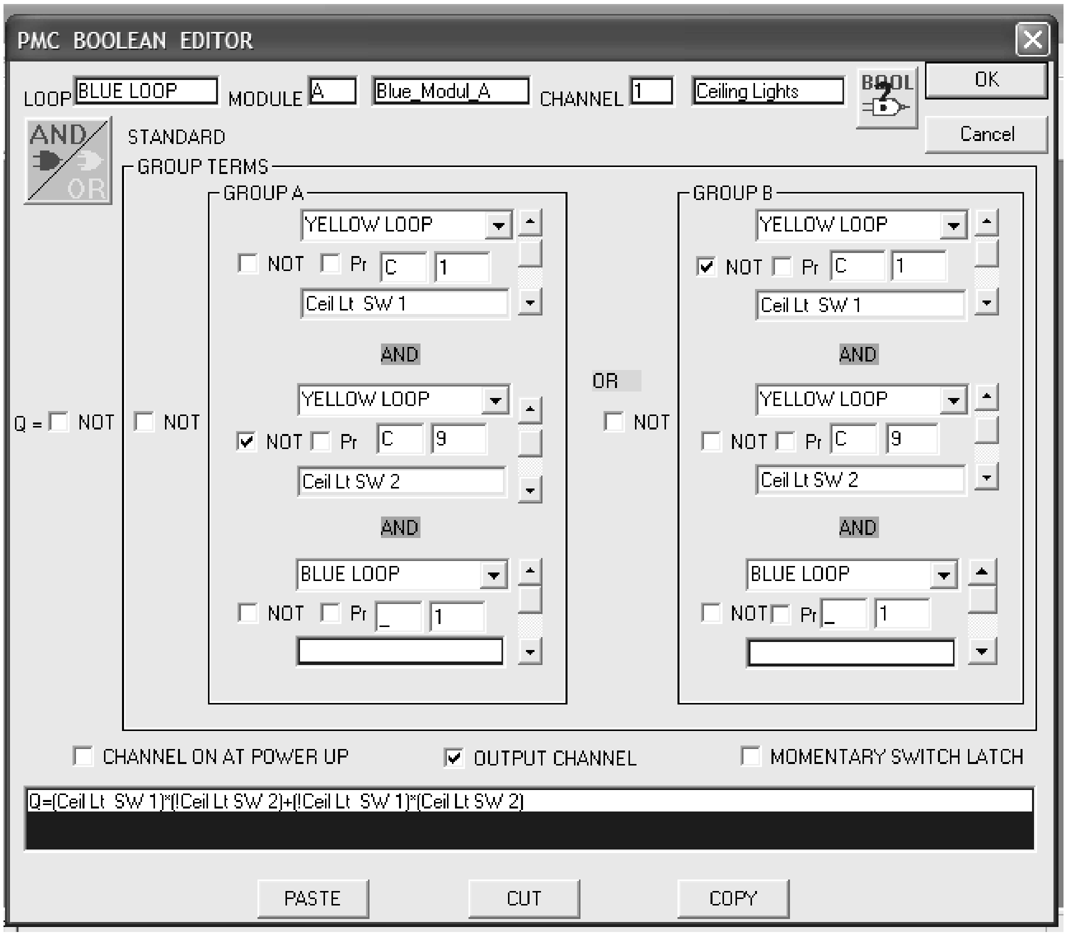
Once BA1 is latched on, it will stay latched until channel BP10 comes on. Momentarily turning BP10 on will cause BA1 to turn off until YC1 is operated again.



Latching an Output

EXCLUSIVE OR

An exclusive OR is a special kind of OR that works with two or more inputs. In a two input Exclusive OR arrangement, the output will only be on if one of the inputs is on. The output will be off, if both inputs are off, or if both are on. An example of this is a switch at each end of a hallway.



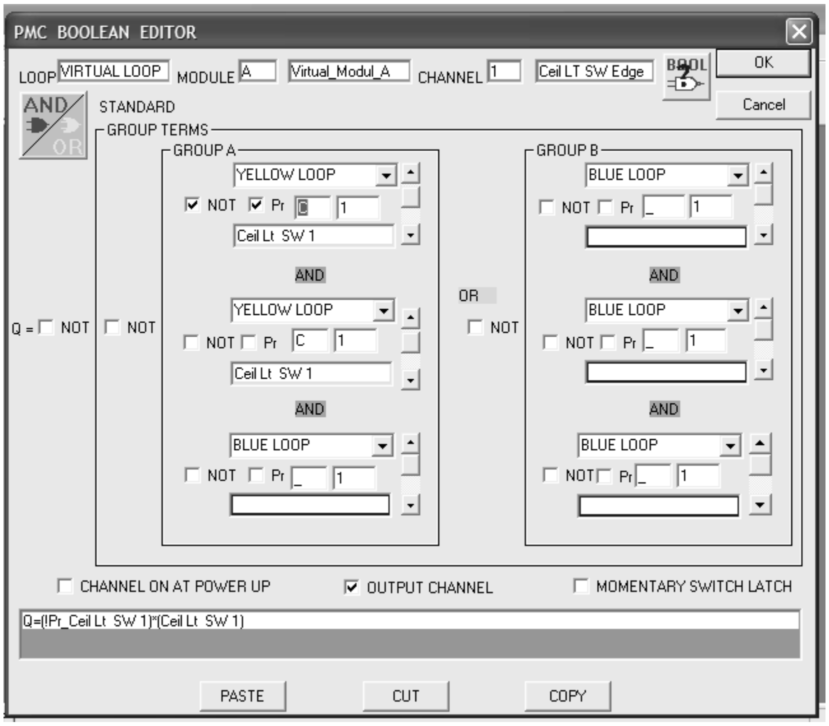
In the example above, YC1 and YC9 are toggle switches at either end of the hall. To read the screen above we need to break the screen into two parts, Group A and Group B. If we look at Group A first, we find that Group A will be true and turn output BA1 on, when YC1 is on, and YC2 is off. If we look at Group B, output BA1 will be on when YC1 is off and YC2 is on.

The truth table looks like this:

IF	YC1	OFF	ON	OFF	ON
IF	YC9	OFF	ON	ON	OFF
Then	BA1	OFF	OFF	ON	ON

Next, let's explore using **edge detected pulses** and the **Exclusive OR** to latch a channel on and off with each edge pulse.

Latching ON, Latching OFF using a Pulse



The Boolean on VA1 to the left, creates a short leading edge pulse each time switch YC1 turns on.

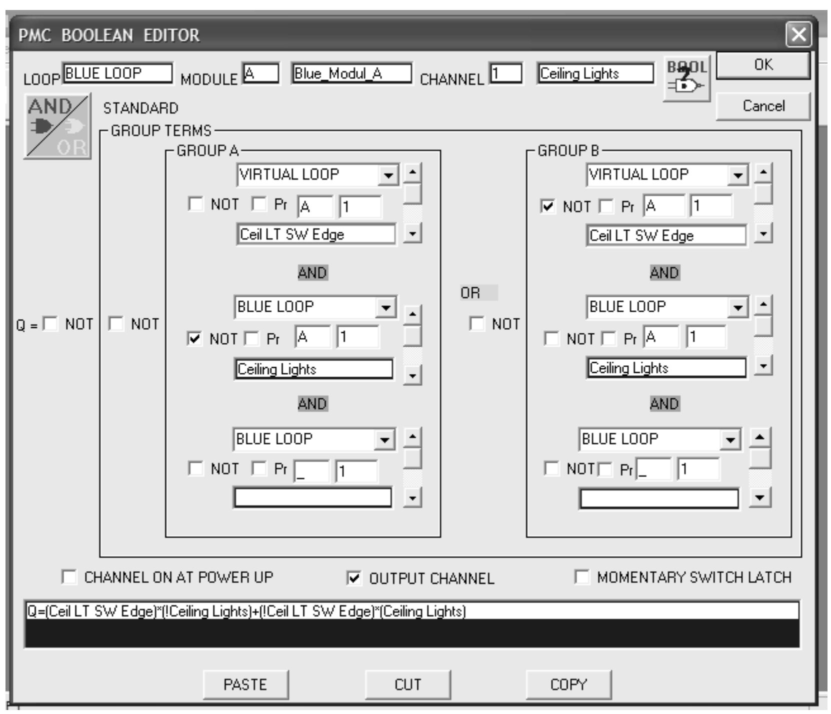
The pulse can be created on any channel in the virtual, Blue Loop, or Yellow Loop.

We can use the pulse we created on VA1 in a boolean to latch BA1 ON and OFF.

This is particularly useful when using momentary push buttons.

We will again look at Group A and Group B separately. To understand how this boolean works, it is important to remember that booleans are processed every communication cycle. Lets look at the first cycle and assume that the edge pulse occurs during this cycle. During the first cycle, BA1 is OFF and VA1 is ON. This makes Group A true and causes BA1 to turn ON. In the second cycle, since the pulse only lasts for 1 cycle, we see that VA1 is OFF and BA1 is ON, which makes Group B true and Group A false. If you study Group B, you will see that it creates a latch because BA1 = BA1. Channel BA1 will now stay on until another pulse comes along. If the edge pulse occurs again, it makes Group B false which unlatches BA1.

With a boolean such as this, a channel will latch on and then off upon alternate pulses.



In the example section of this manual you will see how we can use timer pulses to create a flasher.

Using Timers

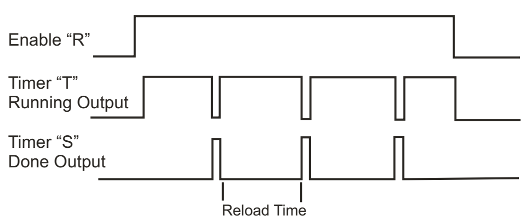
TIMER FUNCTIONS

There are ten timers included in the 160 channel system. Seven of these timers can be set in 0.1 to 25.4 second intervals and are used when that kind of precision is needed. The other three timers can be set in 10 to 2540 second intervals, or 42.33 minutes. The 320 channel system has 16 timer modules, each having 10 timers. The first 6 timers in each module can be set from .1 to 25.4 seconds, and the last 4 can be set up to 2540 seconds in 10 second intervals.

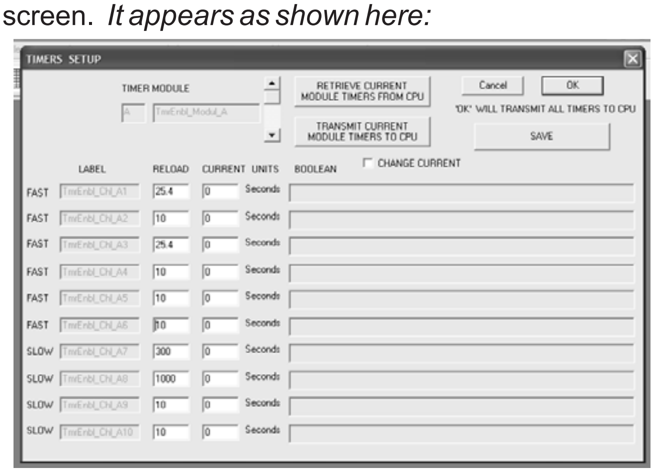
Each timer has two outputs (channels). In the 160 channel system, Timer Running Output "T" and the Timer Done Output "S" and one input, Enable "R". In the 320 channel system, there are timer modules A through P each having 10 timers. In this case you will have timer channels such as Timer enable A1, Timer run A1 and Timer Done A1.

A timer starts to run when the Enable input goes high (is on) and will continue to run as long as it is high. Once the enable input is gone, the timer stops.

The input and two output signals appear as shown here:



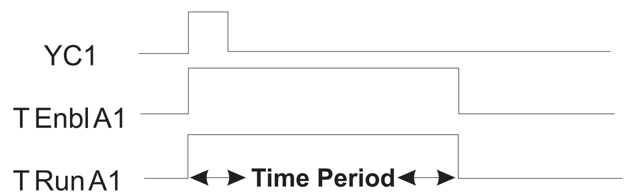
The Timer Done channel, "S", is present for only 40 milliseconds and is high when the Timer Running channel is low. To set up the time intervals of the timers, click on "PMC Setup". Move the cursor to "Timer Setup" and click to bring up the Timer Setup screen. *It appears as shown here:*



Timers can be used for either a repetitive function such as a flasher, or a lamp delay. Let's look at how each of these functions are used. First, let's look at using a timer in a **delay function**.

In this application, the timer is started by an input switch and it will stay on for a fixed period of time. If the input switch is used to initiate the timer, the timer will have to be latched on to keep it going. A latching equation needs to be used. *The equations would be:*

T Enbl A1 = THE TIMER ENABLE (Timer Loop)
YC1 = THE SWITCH INPUT
T Run A1 = THE TIMER'S RUNNING OUTPUT
T Enbl A1 = YC1+ T Run A1



Making the Enable R1 equal to the input switch YC1, starts the timer. Then, the timer keeps itself running (enabled) for the run period TrunA1. At the end of the time period, neither YC1 or TrunA1 is present and the timer is no longer enabled. The input, YC1 could be a momentary pushbutton, or a pulse created at the edge of switch. This pulse would be 40 milliseconds long. That is long enough to start the timer, which may be minutes long.

TIMED OUTPUT A typical application for a timed output is a check out light on an ambulance. This is an output that is turned on with a momentary switch, latches on for 15 minutes and then turns off after the time period is up. Lets look at the booleans to make this happen.

TmrEnblChl_A8 = THE TIMER ENABLE CHANNEL
YC1 = A SWITCH INPUT CHANNEL
TmrRun_Chl_A8 = THE TIMER'S RUNNING OUTPUT
BA1 = THE OUTPUT CHANNEL

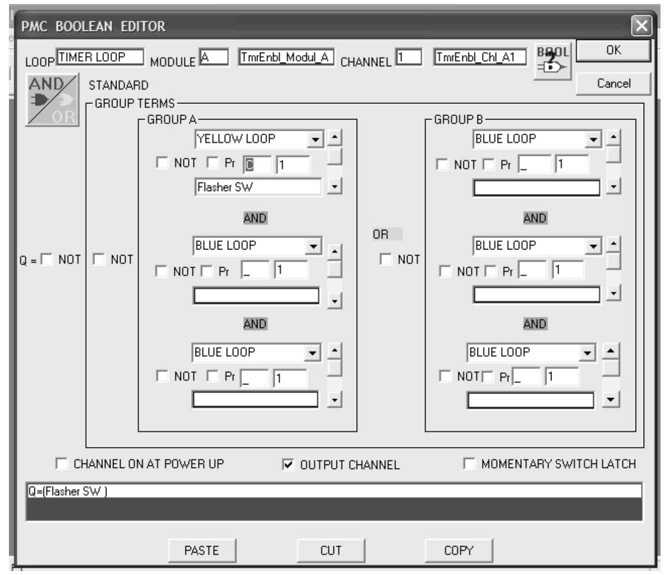
TmrEnblChl_A8 = YC1+ TmrRun_Chl_A8
(The timer is started with the momentary switch and runs until the run pulse goes low)

BA1 =TmrRun_Chl_A8
(The output BA1 will be on, as long as the timer running output is on) In the PMC setup/ timer set up menu, set the time to 15 minutes menu.

A Flasher

Using A Timer Pulse And Latched Output

FLASHER



A boolean for the timer enable must be created to create a flasher. Let's say that we want the timer to operate when a switch on YC1 is ON. In the PMC setup, timer set up menu, enter the time of the ON/OFF periods in timer A1. A good setting for a turn signal is 0.3 sec.

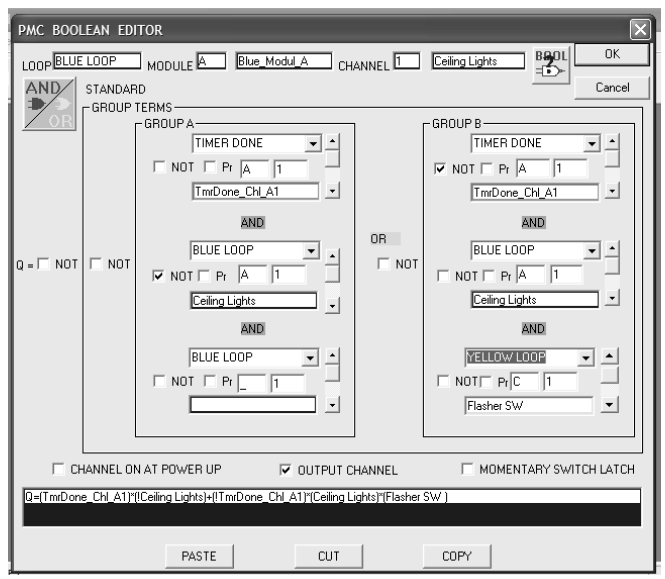
The first Boolean will be:

T Enbl A1 = YC1

This means that the timer will be running as long as the switch on input YC1 is on. To create the flash, we create an Exclusive OR with the timer done pulse and the output to a light, which here we will call BA1. An exclusive OR means that the output will be ON only when one input is on, but not both.

This equation will be:

BA1 = (BA1 * !T done A1) + (!BA1 * T done A1)



One slight problem with this equation, is that the output BA1 is indeterminate when the switch YC1 is turned off. It may be on or off, depending on exactly when the switch YC1 is turned off. To be sure the output is off when the switch is off, the input YC1 should be AND'ed with the timer equation for BA1. Then, BA1 can only be on when the switch is on.

The equation will then be:

BA1 = (Tmr doneA1* !BA1)+(!Tmr doneA1* BA1* YC1)

With each timer done pulse, the output will flash on and off.

Many people find this logic difficult to follow. Do not be concerned. There is a chapter in this manual with examples of common Boolean Logic statements used in vehicles.

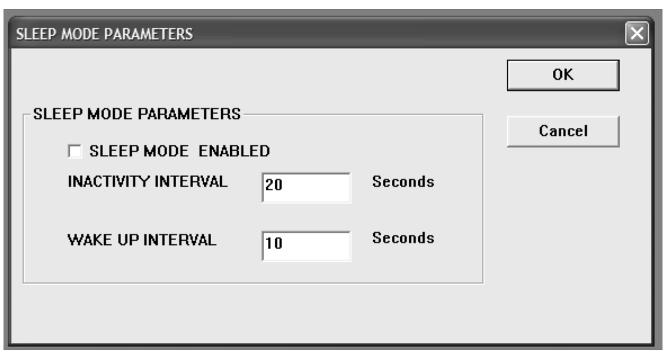
Sleep Mode

SLEEP MODE

The system includes a sleep mode, which puts it to "sleep" to save battery power when the vehicle is not in use. When the system goes to sleep, any output that is on will turn off. There are two inputs on the CPU Module that are used to signal the system to enter into "Sleep Mode." These would typically be connected to an ignition source. As long as the ignition is on, the system will remain on and will not enter sleep mode. Once the ignition is turned off and the system does not sense any activity from any inputs, the system will go to sleep after a period of time. When the system is asleep, it will turn itself back on periodically and check if any of the inputs have changed. If it senses a change in any input, it will wake up and resume normal operation. If there are no changes in inputs, it will go back to sleep.

To set up the Sleep Mode Timer, click on "PMC Setup." Move the cursor to Sleep mode and click. This will bring up the Sleep Mode Timer screen.

It will look like this:



There are three settings to be made on this screen: Sleep Mode Enabled, Inactivity Interval, and Wake Up Interval. The Inactivity Interval timer can be set from 10 to 2540 seconds, which is the time the CPU will use to decide that there hasn't been any activity on the system, allowing it to shut itself down. *If this time is set too long, the system will be wasting battery power. If it is set too short, it may go off before the driver has left the vehicle.* To set this timer, move the cursor to the box, click on it and type in the new value. If you input a value out of the range, the program will prompt you to correct it.

The Wake Up Interval Timer can be set from 10 to 254 seconds, which is the amount time the CPU will wake up for to determine if there is any activity on the system. *If this interval is set too long, the CPU may take a long time to see that an input has changed. If it is set too short, it will be waking up too often and waste battery power.* **Suggested settings are 240 seconds for inactivity and 5 seconds for wake up.**

To set this timer, move the cursor to the box, click on it and type in the new value. If you input a value out of the range, the program will prompt you to correct it.

The CPU will provide power during sleep mode to back light switches via pins J2-1, J3-1 and J4-1. If you are using switch adapters or the lighted rocker switch module, consideration should be given to the current draw from the backlit switches.

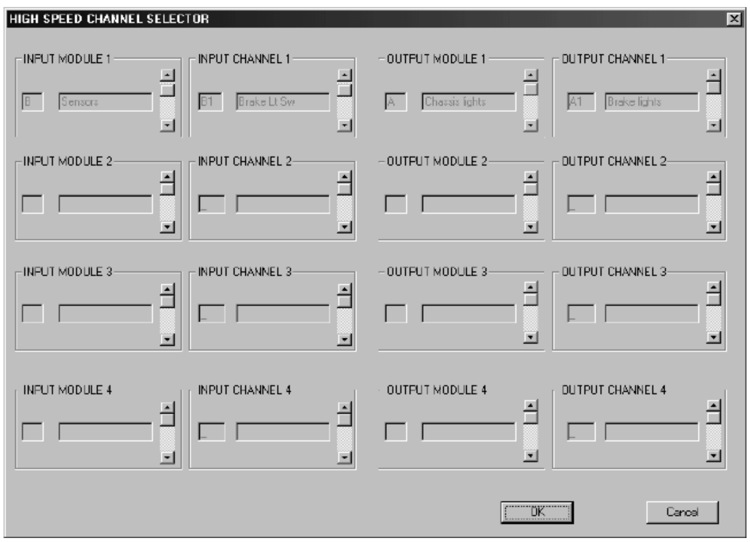
High Speed Channels

There are 4 high-speed channels available, on both the 160 channel system and the 320 channel system. The 320 channel system has high-speed channels on the Blue Loop only. These channels are to be used for functions that demand very short delays through the system, such as brake lights. These channels have direct relationships between the inputs and outputs. The CPU does not spend any time evaluating a Boolean. Booleans cannot be written for a channel that has been defined as a high-speed channel.

The relationship between the input and output is defined on the high-speed channel selector screen.

Any output channel on any module may be defined as a high-speed channel. To define a channel as a high-speed channel, select "PMC SET UP" from the task bar, then click on high-speed channels.

The screen below will appear:



Using the Up/Down arrow buttons for "Input Module 1", select which module, A-P, containing the input channel you would like to use to operate the high-speed channel. Move to the next box “Input Channel 1" select the input channel (switch) that will control the output. In the next box, “Output Module 1", select the module that contains the output you would like to operate. In the final box, select the output channel you wish to control. Click on “OK” and the channel is set up as a high-speed channel.

As you may recall from previous pages, the CPU evaluates when to turn a channel on and does it in order from channel A1 to channel P10. The system takes .040 seconds to scan from A1-P10. In most cases, when an input switch changes state, the system sees it in the first cycle, looks to see if it is still present in the second cycle, and then processes the boolean to turn the appropriate output on. Depending upon the channels involved, it can take from a little more than .040 seconds to .080 seconds for the output channel to come on after the input appears. With a high-speed channel, the delay is .040 maximum, depending upon the channels involved. *If this delay is still too long for your specific application, see the data sheets on I/O module 00-00846-506 and 516.* These modules have two inputs and outputs that can be tied together by a dip switch setting and will function as a solid-state relay. In this case, there is practically no delay.