Chapter 10

Edge Detection On the 160 Channel CPU

System Timing

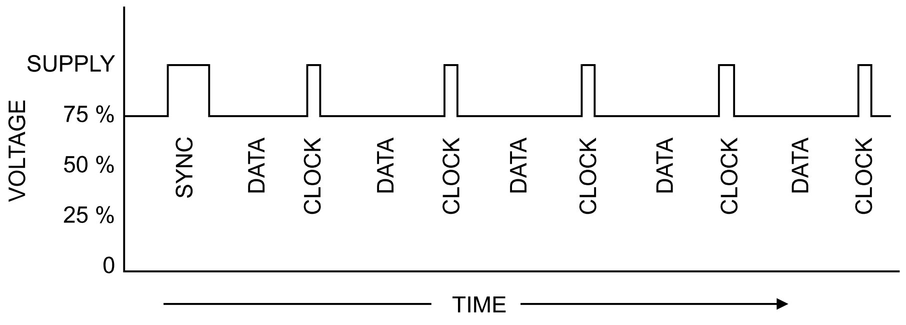
The PMC system has a total of 160 channels for information. These channels appear serially on the PMC multiplex bus, every 40 milliseconds.

The multiplex signal is divided into 16 module groups of ten channels each. At the beginning of a module data group, there is a system reset pulse. Channel A1 occurs at the end of this reset pulse. At the end of the first data window, a clock pulse is sent.

Channel A2 follows A1, followed by another clock pulse, and so forth. At the end of ten channel A group, there is a synchronization pulse, which signals the system that the next group of ten channels is beginning. This sync pulse is shorter than the system reset pulse. The sync pulses also act as clock pulses. *The signal appears as shown below.*

The CPU acts as a Master, sets up these timing signals and puts them on the PMC multiplex bus for use by all the slave input and output modules. The signal is generated by the microprocessor in the CPU module.

In addition to the timing function, the processor is performing the Boolean logic that has been programmed into the system. The Booleans are calculated in order, starting with the earliest equation. In other words, the Boolean for channel A1, if there is one, will be the first to be calculated. Then, the one for channel A2, if there is one, will be calculated. *This is an important point to remember when writing certain Booleans*, as the value of the earliest channels may change before the later channels are calculated. This can also be of help in performing certain latching and timing functions and when programming the timers. These calculations are not synchronous with the system timing.



CREATING AN EDGE PULSE ON THE 160 CHANNEL CPU

Creating and using an edge detected pulse has been demonstrated for the 320 Channel CPU in previous chapters. *The following example is the use of the timing aspect to create a pulse when a switch is turned ON/OFF.*

First, a channel of a Virtual Module is made equal to an input. Then, this channel is made to equal another *earlier* channel.

Here is an example **Switch = B1**

Virtual Module = Q

Q2 = B1

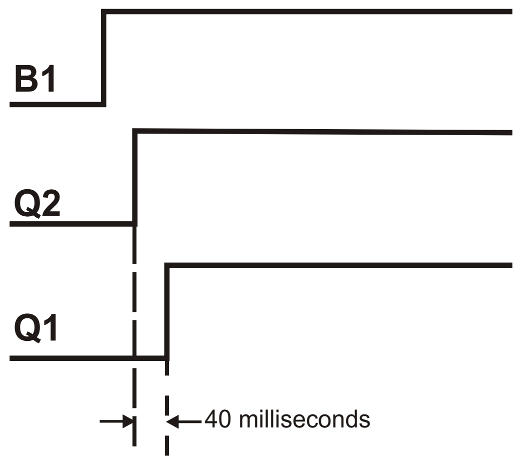
Q1 = Q2

With these equations, Q2 equals the input B1, and Q1 equals Q2. Q1=Q2 will not be calculated until it comes around again, after having calculated all therefore low. At some point in time, input B1 is switched high. Since Q1 is calculated before Q2, it will be a 0 or low, since Q2 has not been calculated yet. When Q2 is calculated, it will be set equal to B1, making it a 1 or high. The next time Q1 is calculated and made equal to Q2, it will then become a 1 or high. This has happened approximately 40 milliseconds later.

System Timing

CREATING AN EDGE PULSE ON THE 160 CHANNEL CPU *(Continued)*

Therefore, the edges of the two signals are delayed by the 40 milliseconds, as shown here.



By creating a Boolean with these two signals, pulses can be formed.

A Boolean that will create a pulse when the input switch goes high, or at the ***leading edge***, would be the following:

Q3 = Q2 * !Q1

This will create a pulse, which starts when Q2 goes high and ends when Q1 goes high. Similarly, a pulse can be created between the two signals when the switch goes OFF instead of going ON. This would be done by inverting Q2 instead of inverting Q1.

The Boolean would be:

Q3 = !Q2 * Q1

By creating an Exclusive OR between these two signals, a pulse will be created each time the input switch changes states.

This equation would be:

Q3 = Q1 * !Q2 + !Q1 * Q2

This technique can also be used to create longer pulses by adding additional steps in the process, such as starting with B1 = Q10.

Then

Q9 = Q10
Q8 = Q9
Q7 = Q8

Finally

Q3 = Q10 * !Q4

This step by step process adds 40 milliseconds for each time through the Boolean processor. This process can be made quite long by continuing this process through a number of modules.

This is not the only way time delays can be produced. The system includes ten, programmable timers, whose *function and use is explained in chapter five.* The technique of creating a pulse at the edge of a input switch function can be useful in starting a timer.