Emulation Code User's Manual - the 68230 PI/T

Information on the 68230 PI/T can be found in the text book on pages 399-415. The information presented here is intended only to make writing code that uses the 68230 emulation easier.

What do I need to use the 68230 emulation?
You will need to load the file called "68230.com" into the debugger. This file can be loaded by typing, from the debugger command line: File Command 68230.com

I have put it into /home/courses/282/pub

How can I use the timer?
You can use the timer in two modes: interrupt mode or polled mode. The polled mode works similarly to the polled I/O from the keyboard. First, the timer is set up by writing to the counter preload registers and the timer control register in that order. Then the timer can be used by starting the timer, checking the timer status register until it signals the end of the given time, and then clearing the timer status register with a direct clear to reset the timer.

The interrupt mode operates much differently. To use this mode, the timer interrupt vector register must be properly set first. Then the counter preload registers can be set, and finally, the timer control register can be initialized. The interrupt vector register should not be changed from its initial value without stopping the time and starting the entire process from scratch.

In summary,
polled I/O mode

1. initialize counter preload registers
2. initialize timer control register
3. start the timer
4. poll the timer status register
5. when finished, direct clear the TSR. YOU MUST WRITE TO TRSW!!
6. to time the same period again, goto step 3.

interrupt I/O mode

1. initialize timer interrupt vector register
2. initialize counter preload registers
3. initialize timer control register
4. start the timer
5. The timer will continue timing the same period until stopped.
6. When finished, stop the timer by writing to the timer control register.

Steps 2 and 3 for the polled I/O case, or steps 3 and 4 for the interrupt case can be done together because they both involve writing to the timer control register.

How do I set the period I want to time?

You always set the period you want to time by writing to the three counter preload registers. As detailed in the text book, all three registers can be set at once using a single MOVEP instruction, although three MOVE.B instructions will also work. The timer ticks 125,000 times per second, so the proper number to load into the registers can be calculated by multiplying the number of seconds by 125,000.

How do I initialize the timer control register?
You can initialize the timer control register by doing a MOVE.B command. The number written to the control register determines the mode that the timer operates in. The settings should be made as follows:

action on zero detect	first three bits	should be either 100 for polled or 101 for interrupt
counter load	fourth bit	can be 1 to rollover or 0 to reset from preload
clock control	sixth,seventh bits	should always be 00
timer enable	eighth bit	can be 1 to start or 0 to stop timer

Additional Guidelines
The operations should always be done in the order specified. This means that the timer interrupt vector register should never be initialized after you start the timer. Also, anytime you want to change a parameter, the timer should be stopped, the parameter changed, and then the timer restarted. Changing a parameter while the timer is running could do anything from having no effect to sending the 68000 into random code sections. it is very important that you stop and restart the timer when changing parameters to give the emulation code a chance to recognize your changes.

In addition to the above guidelines, it is important to keep in mind that this timer is emulated. Therefore, the timed periods will not always be accurate. Due to overhead involved in running the debugger, the timer may run slightly fast or slow. Thus, it is usually best to try and minimize overhead if you want the timer to approximate the real device. In other words, if you are trying to run a clock, timing every tenth of a second might create excessive overhead and force the timer to ruin very slowly, with each tenth actually taking about a quarter of a second. However, timing every minute would create very little overhead, so the timer might actually run faster than a real clock. The moral of the story is: set the counter preload registers as if there were 125,000 ticks per second and expect the timer to go a little faster or slower than you set it to.

Timer register addresses
The addresses of the emulated timer registers are:

TCR	timer control register	$10021
TIVR	timer interrupt vector register	$10023
CPRH	counter preload-high	$10027
CPRM	counter preload-medium	$10029
CPRL	counter preload-low	$1002B
TSR	timer status register	$10035

```
TSRW    timer status register (write only)      $10037
```

If you copy the clocklab.com file to your directory, you will be able to use an easy command to start up the debugger without needing the File Command 68230.com discussed above, i.e., by typing:

db68k -c clocklab <your_prog_name>

This means all of the following files should be in your directory:
- 68230.com
- acia.com
- iodorm.com
- clocklab.com
- extra.com

The extra.com file just sets the usp and ssp. If you want to do this tyourself, you can skip this file.