EE 308 – LAB 7

HCS12 Timer Input Capture and Output Compare

Last week you wrote programs using the HCS12 Timer Overflow Interrupt and Real Time Interrupt. This week you will work with the Timer Input Capture and Timer Output Compare functions.

1. In this lab, it will be convenient for our program to have the HCS12 print out some information to the computer screen. You can use the standard C function `printf()` to do this. Here is a program `hello.c` which uses `printf()`:

```
#include <iodp256.h>
#include <stdio.h>

#define TRUE 1

main()
{
    printf("Hello, world!\r\n");
}

int putchar(char c)
{
    while ((SCI0SR1 & 0x80) == 0) ;  // Wait until okay to xmit
    SCI0DRL = c;                     // Send byte
}
```

   To use this program, you will need to modify you `lkf` file to include the C libraries which include `printf()`:

```
# link command file
#
+seg .text -b  0x1000 -n .text  # program start address
+seg .const -a .text            # constants follow code
+seg .data  -b 0x2C00           # data start address
crts.o                          # startup routine
hello.o                         # application program
c:\cx32\lib\libi.h12            # include integer library
c:\cx32\lib\libm.h12            # include machine library
+def __memory=@.bss             # symbol used by library
+def __stack=0x3C00             # stack pointer initial value
```

   Run the program `hello.c` on your HCS12 board. Verify that you can print to the computer screen.

2. Start with the following program, which is just a do-nothing infinite loop:

```
#include <iodp256.h>

#define TRUE 1

main()
{
    DDRA = 0xff;        /* Make all bits of PORTA output */
    PORTA = 0x00;
    while (TRUE)
   {
       _asm("wai");
   }
}
```

   Connect your debounced switch to Input Capture 2 (`PORTT`, Bit 2), as shown below:

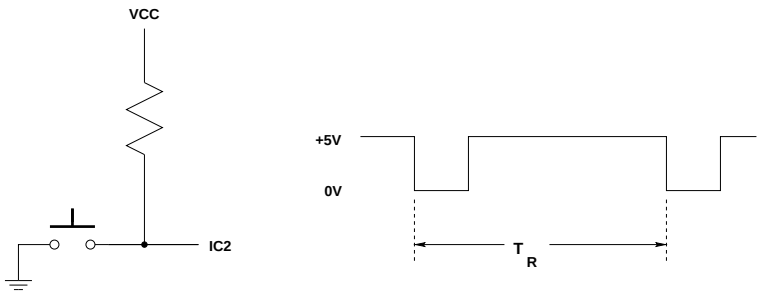

Figure 1: Circuit to measure speed of button pushing.

   The right part of Figure 1 is what the signal to IC2 will look like if you push the pushbutton twice.

3. Add to your program code to measure the time T_R between the two falling edges of the signal in Figure 1. Set the prescaler so you can unambiguously measure time differences of at least 250 ms.

   Use the `printf()` function to print out the result – the number of timer ticks between pushes of the button. (Do not try to calculate the actual time as a floating point number. When you use floating point numbers the programs will become very large, and the floating point library for our compiler appears to be buggy.) You should write your program as an infinite loop so that after pressing the button twice your program will print out the result, the go wait for the next two presses.

4. Test your program on your EVBU. See how fast you can push the switch twice. Record several values in you lab notebook, and convert the times to seconds.

5. Add an Output Compare function on Bit 3 of `PORTT` to generate a 1 kHz square wave. Use a logic probe to verify that Bit 3 of `PORTT` is toggling.

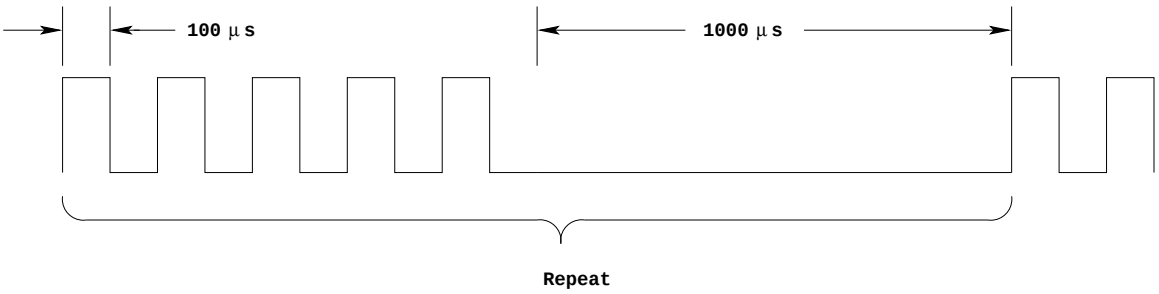

Figure 2: Friend signal for Part 6.

6. Add an Output Compare function on Bit 4 of `PORTT` to generate the following signal: The signal consists of five pulses which are high for 100 μs and low for 100 μs, followed by a 1000 μs low signal. This signal then repeats.

7. Connect your output compare signals to a logic analyzer. Verify that the square wave has a 1 kHz frequency and a 50% duty cycle, and that the other signal looks like the signal of Figure 2.