EE 308 – Homework 5

Due Feb. 23, 2009

For the homework problems which follow assume you have included the file `hcs12.h` in your C program. Thus, you can refer to `PORTB` when you want to access a byte at address `0x0001`. Where I ask for "some code" just write that part of a C program which will do the task. Where I ask for "a program" write a complete program, include the `#include "hcs12.h"` line, the declaration of variables, the `main()` function, etc.

1. Write some C code which will set bits 3 and 1 of the eight-bit register at address `0x0075` while leaving the other bits unchanged.

2. Write some C code which will clear bits 13, 11, 6 and 0 of the sixteen-bit register at address `0x0076` while leaving the other bits unchanged.

3. Write some C code which will set bit 0 of PORTB if the 16-bit value at address `0x0099` is less than 72, and will clear bit 0 of PORTB if the value is greater than or equal to 72.

4. Write some C code which will wait until Bit 7 of the eight-bit number at address `0x00cc` becomes set.

5. Consider an array of 8-bit data located in memory with a starting address of `$1000` and an ending address of `$101F`. Write a C program which will swap the first element of the array with the last element; the second element with the next-to-last element, etc.

6. An MC9S12 has the following in some of its registers:

TSCR1	TSCR2	TFLG2
0x80	0x04	0x80

   (a) Is the timer subsystem enabled? How can you tell this?

   (b) How long will it take (in ms) for the timer to overflow? How can you tell this?

   (c) Has the timer overflowed since the last time the TOF flip-flop was reset? How can you tell this?

7. Write a C function to convert an eight-bit binary number to its Gray code equivalent. The procedure to do this is as follows: Exclusive OR the number with the same number shifted right by one bit. The function should have one argument (the eight-bit number), and should return the eight-bit Gray code value.

8. Write a C function to generate the next pattern in the sequence for an eight-bit Johnson counter. The procedure to do this is as follows: Shift the present pattern to the right by one bit. The most significant bit of the next pattern is the inverse of the least significant bit of the present pattern. The function should have one argument (the curent pattern, an eight-bit number), and should return the eight-bit value of the next pattern.

9. Write some C code to take the next entry out of a table, write it to `PORTB`, and update the index into the table. Here is an example of what the table might look like:

   ```
   const char table[] = {0x00, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80};
   ```

   Use a variable called `table_index` to hold the index of the pattern to write to PORTB. Your code should write the table entry corresponding to the index to PORTB. (For example, if `table_index` were 5, you would write the fifth element of the table, 0x10, to PORTB.) It should then update `table_index` to point to the next element of the table. Make sure that `table_index` stays between 0 and `sizeof(table)-1`.