EE 308

Exam 2

March 27, 2009

Name: _____

You may use any of the Freescale data books, the class lecture notes, and a calculator. Show all work. Partial credit will be given. No credit will be given if an answer appears with no supporting work.

For all the problems in this exam, assume you are using an MS9S12 with an 8 MHz crystal, resulting in a 24 MHz bus clock.

Also, assume that `hcs12.h` has been included, so you can refer any register in the MC9S12 by name rather than by its address in any C code you write.

1. The following questions concern writing C code.

   (a) Write some C code which will read the 16-bit unsigned number at addresses `0x3000` and `0x3001`, and will store that number into memory locations `0x1010` and `0x1011`.

   ```
   *(unsigned int *) 0x1010 = *(unsigned int *) 0x3000;
   ```

   (b) Write some C code which will set bits 3 and 5 and clear bits 2 and 4 of the byte at address `0x0049`, and leave all the other bits of that byte unchanged.

   ```
   *(char *) 0x0049 = *(char *) 0x0049 | 0x28 & ~0x14;
   ```

   or

   ```
   char *ptr;

   ptr = (char *) 0x0049;

   *ptr = *ptr | 0x28 & ~0x14;
   ```

   (c) Write some C code which will do the following: Wait until the TOF bit of the TFLG2 register is set. It will then read the contents of address 0x0060 (as an eight-bit number), write that number to PORTB, and then clear the TOF flag. (Assume that all bits of PORTB have been set up for output.)

   ```
   while ((TFLG2 & 0x80) == 0) ;
   PORTB = *(char *)0x0060;
   TFLG2 = 0x80;
   ```

2. Below are the contents of the memory of an MC9S12:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
FFC0	27	05	9F	CD	99	03	84	9C	01	9B	CC	90	66	FC	93	30
FFD0	7E	E3	4B	7E	E5	38	21	54	05	83	09	34	2A	38	3C	03
FFE0	41	38	66	F2	7C	13	37	0C	25	F2	0C	38	5F	1B	42	1A
FFF0	7A	26	21	13	6A	AA	20	1F	4B	38	33	38	45	38	08	00

(a) What is the address of the first instruction the MC9S12 will execute when coming out of reset?

The reset vector is at address 0xFFFE, so the first instruction is at address 0x0800

(b) What is the address of the first instruction of the Timer Overflow interrupt service routine?

The TOF interrupt vector is at address 0xFFDE, so the first instruction of the TOF ISR is at address 0x3C03

(c) What is the address of the first instruction of the IIC Bus interrupt service routine?

The IIC interrupt vector is at address 0xFFC0, so the first instruction of the TOF ISR is at address 0x2705

3. Below are the contents of the memory of an MC9S12:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
FFC0	27	05	9F	CD	99	03	84	9C	01	9B	CC	90	66	FC	93	30
FFD0	7E	E3	4B	7E	E5	38	21	54	05	83	09	34	2A	38	3C	03
FFE0	41	38	66	F2	7C	13	37	0C	25	F2	0C	38	5F	1B	42	1A
FFF0	7A	26	21	13	6A	AA	20	1F	4B	38	33	38	45	38	08	00

The MC9S12 registers have the following values:

Reg	–	–
	S X H I N Z V C	
CCR	1 1 0 0 0 1 0 1	
A:B	A3	92
X	82F2	
Y	12F7	
SP	1CE7	
PC	2024	

Assume that the Timer Overflow interrupt has been enabled.

(a) Describe what happens to the MC9S12 between the time that the Timer Overflow occurs and the MC9S12 starts executing the first instruction of the Timer Overflow interrupt service routine.

The MC9S12 completes the instruction it is working on, pushes registers CCR, B, A, X, Y and PC onto the stack (which decrements the SP by 9), sets the I bit of the CCR, then loads the program counter with the address from the TOF interrupt vector.

(b) Fill in any values in the registers which have changed in the MC9S12 by the time it gets to the first instruction in the timer overflow interrupt service routine. (You can leave blank any register which has not changed its value.)

Reg	–	–
	S X H I N Z V C	
CCR	1 1 0 1 0 1 0 1	
A:B		
X		
Y		
SP	1CDE	
PC	3C03	

Below, show the contents of the stack which were changed due to the interrupt process:

Address	Value	What
1CDE	C5	CCR
1CDF	92	B
1CE0	A3	A
1CE1	82	X High
1CE2	F2	X Low
1CE3	12	Y High
1CE4	F7	Y Low
1CE4	20	RTN High
1CE6	24	RTN Low

(c) From the time the timer overflows to the time the MC9S12 starts executing the first instruction in the interrupt service routine, about how much time (in microseconds) has elapsed?

It takes 9 clock cycles to push push the registers onto the stack and load the interrupt vector into the PC (see the SWI instruction, which also generates an interrupt). This takes 9/24,000,000 = 0.375 μs.

4. Below are the values of some timer registers in the MC9S12:

TIOS	TSCR1	TCTL1	TCTL2	TCTL3	TCTL4	TIE	TSCR2	TFLG1	TFLG2
32	80	A4	C2	5F	76	2C	84	52	00

(a) Which timer channels are being used for output compare?

TIOS is 0x32 = 00110010_2, so channels 5, 4 and 1 are set up for output compare.

(b) Which timer channel interrupts are enabled?

TIE is 0x2c = 00101100_2, so channels 5, 3 and 2 interrups are enabled. Also, TSCR2 = 0x84, so the Timer Overflow Interrupt is enabled.

(c) What action is timer channel 2 set to perform? (I.e., if it is set up as input capture, which edge will it capture; if it is set up as output compare ; if it is set up as output compare what action will occur when `TCNT` equals `TC2`?)

Channel 2 is set up for input capture. In input capture mode, bits 5 and 4 of TCTL4 control its function. Bits 5 and 4 of TCTL4 are 11, so channel 2 will capture the time of any edge (rising or falling).

(d) What action is timer channel 3 set to perform?

Channel 3 is set up for input capture. In input capture mode, bits 7 and 6 of TCTL4 control its function. Bits 7 and 6 of TCTL4 are 01, so channel 3 will capture the time of a rising edge.

(e) What action is timer channel 4 set to perform?

Channel 4 is set up for output compare. In output compare mode, bits 1 and 0 of TCTL1 control its function. Bits 1 and 0 of TCTL1 are 00, so channel 4 is disconnected from the output pin. Channel 4 can be used to generate an interrupt after a specified time to control the timing of a program.

(f) What action is timer channel 5 set to perform?

Channel 5 is set up for output compare. In output compare mode, bits 3 and 2 of TCTL1 control its function. Bits 3 and 2 of TCTL1 are 01, so channel 5 will toggle the output pin on a compare.

(g) Which timer flags are set?

TFLG1 = 0x52 = 01010010_2, so flags 6,4 and 1 are set. TFLG2 = 0x00, so the TOF flag is not set.

(h) What is the timer prescaler set at – i.e., by what factor will the processor clock be divided before driving the `TCNT` register?

The timer prescaler is bits 2-0 of TSCR2. TSCR2 is 0x84, so the timer prescaler is 0x4. The processor clock is divided by $2^4 = 16$.

(i) How long (in seconds) will it take for the `TCNT` register to overflow?

With a prescaler of 0, it takes 2^{16} cycles/24,000,000 cycles/sec = 2.73 ms for the timer to overflow. A prescaler of 4 increases this time by 16, for an overflow time of 16 $\times 2^{16}$ cycles/24,000,000 cycles/sec = 43.69 ms.

5. You are tasked with the design of a system to control the amount of light used to illuminate the stage of a microscope. You need to adjust the light intensity with a resolution of at least 0.5%. You do this by using the PWM of a MC9S12 to control the duty cycle of the lightning system.

   (a) Write some C code to provide a pulse-width-modulated signal on Bit 5 of Port P of the MC9S12. Use a PWM frequency of 200 Hz. Be sure to explain the values you use.

   To get at least 0.5% control, the PWM Period has to be 200 or greater. I'll let PWMPER5 = 200. The period of a 200 Hz frequency is 1/200 = 5 ms. To get 5 ms, we need the total number of cycles to be 5 ms $\times$ 24,000,000 cycles/sec = 120,000. We need PWMPER5 $\times 2^N$ = 120,000 (clock mode 0) or PWMPER5 $\times 2^{N+1} \times M$ = 120,000 (clock mode 1). Clock mode 0 won't work, so with PWMPER5 = 200, we have $2^{N+1} \times M = 600$. Let $N = 1$, $M = 150$. (Will also work with $N = 2$, $M = 75$.) Need to choose clock mode 1, PCKB = 1, PWMSCLB = 150.

```
/* Choose 8-bit mode */
PWMCTL = 0x00;
/* Choose left-aligned */
PWMCAE = 0x00;
/* Choose high polarity on all channels */
PWMPOL = 0xFF;
/* Select clock mode 1 for Channel 5 */
PWMCLK = PWMCLK | 0x20;
/* Select PCKB = 1 for Channel 5 */
PWMPRCLK = (PWMPRCLK & ~0x60) | 0x10;
/* Select period of 200 for Channel 5 */
PWMPER5 = 200;
/* Enable PWM on Channel 5 */
PWME = PWME | 0x20;
```

   (b) Write some C code to set the duty cycle of the PWM to 57.5%.

   The duty cycle is 57.5% of PWMPER5, or 0.575 $\times$ 200 = 115.

```
PWMDTY5 = 115;      /* 57.5% duty cycle on Channel 5 */
```