Lab 9: On-Board Time Generation and Interrupts

Summary:

Develop a program and hardware interface that will utilize externally triggered interrupts and the on-board timer functions of the 68HC12.

Learning Objectives:

- Learn to use the on board timer system of the 68HC12.
- Practice the use of interrupts.

Resources and Supplies:

- CML12S-DP256 development board
- CML12S-DP256 MON12 Manual
- PC with WinIDE Development Environment and MON12 monitor program

Important Reminders:

- It is your responsibility to save the programs you create.
- Pre-lab assignments must be completed <u>before</u> coming to the lab.

Background:

68HCS12 Free Running Counter System

The free running counter of the 68HCS12 provides timing functions. The MCU configuration registers relevant to the free running counter system are listed in the table below along with their register addresses. TCNT is the 16-bit counter. Bit 7 of the TSCR register holds the timer enable (TEN). Setting TEN to '1' turns on the timer and resets the value in TCNT. Bit 7 of the TFLG2 register contains the time overflow (TOF), which goes high when TCNT rolls over, changing from $FFFF to $0000. Bits 2-0 of the TSCR2 register contain the 3-bit binary-encoded counter clock prescale factor (PR) that reduces the timer clock frequency by the encoded PR value. PR = 001 means scale by 2, 010 scale by 4, 011 scale by 8, etc. ***For this lab, you should set PR to %101*** which will scale the clock by a factor of 32 and should results in a timer clocking frequency of 8MHz/32 = 256kHz. Remember, it is best to use BSET/BCLR instruction to set/clear control bits rather than *store* instructions.

Register Name	68HCS12 Register Address	Important Bits Name (bit #)
TCNT	$0044-45	all
TSCR	$0046	TEN (7)
TFLG2	$004F	TOF (7)
TSCR2	$004D	PR (2:0)

IRQ Interrupt Configuration.

Master control of all maskable interrupts (like IRQ') is provided by a master mask (switch) that is by ASM instructions. The status of this mask is indicated by the CCR bit I. Instruction **SEI** sets the master mask (I = 1) turning off the maskable interrupt system (engaging the mask). **CLI** clears the master mask (I = 0) turning on the maskable interrupt system (disengaging the mask). When the maskable interrupt system is enabled (I = 0), individual maskable interrupts can then be enabled and used as interrupt sources. During coding and debugging, pay careful attention to the status of the CCR-I bit and turn the system on/off as needed.

The maskable interrupt signal, IRQ, is controlled by the IRQCR register. Bit 7 (IRQE) is the select edge sensitivity bit. By default it is ‘0’ for a negative-edge triggered interrupt. Setting it to ‘1’ would make IRQ positive edge triggered. Bit 6 (IRQEN) enables the IRQ interrupt when set to ‘1’. The remaining bits are unused. ***Storing a value of #$40 to IRQCR would enable an active low IRQ interrupt***. You can also use BSET/BCLR instructions.

Register Name	68HCS12 Register Address	Important Bits Name (bit #)
IRQCR	$001E	IRQE (7), IRQEN (6)

Note you must use the clear interrupt mask (CLI) instruction to enable before turning on any individual maskable interrupts, like the IRQ. The SEI instruction will allow you to set the CCR-I bit.

The interrupt vector address for the IRQ service routine is ($3FF2, $3FF3). This is remapped from the standard address ($FFF2, $FFF3) by the MON12 monitor program. The address for your IRQ service routine must be stored to this interrupt vector. You can achieve this using labels in your code without knowing the actual address of the service routine, or you can assign your service routine to a specific address and then store this address to the IRQ interrupt vector.

Port K: LEDs and Buzzer

As used in previous labs, four LEDs are attached to the lower nibble of Port K (PK0 – PK3). To control the LEDs, you must set these pins as outputs on the data direction register. Setting a value of ‘1’ to any of these port pins will turn on the respective LED.

An audio buzzer on the project board is attached to the 6th bit of Port K (PK5). If Port K is set as an output port, then setting PK5 will turn on the buzzer and clearing PK5 will turn it off. In order to hear the buzzer, a small delay will be needed between turning it on and turning it off.

Be sure your software properly manages the fact that Port K is attached to both the LEDs and the Buzzer and only turns on what you want on at any given time.

The configuration registers relevant to Port K are:

$0032	Port K data register
$0033	Port K data direction register

Connection to IRQ’ Pin

The HC12 active-low maskable interrupt pin, IRQ’, is wired to pin 32 on the BUS_PORT of the CML12S-DP256 development board. The AXM-0295 project board includes a push button switch that generates a logic low when pushed and is well suited to create an interrupt signal. The output of this switch is available on pin 15 of the AUX1-PORT of the AXM-0295 project board. To use this switch as the IRQ interrupt, you must wire these two pins together.

BUS_PORT	32	IRQ’
AUX1_PORT	15	push button switch

Pre-lab Assignment:

- Read this entire lab assignment so you know what to expect in the lab.
- Complete the steps described in the Pre-lab sheet near the end of this document. Each student must complete his/her own pre-lab before coming to the lab and hand it in to the lab TA at the beginning of the lab.

Laboratory Assignment:

This lab consists of two parts. A check-off sheet is included at the end of this lab document.

- Print the check off sheet. Where indicated, you must record your results on the check-off sheet. After you successfully finish each part of the lab, show the TA your results and ask him to sign the check-off sheet.
- Ask the TA to explain how the MON12 program expects you to initialize the stack pointer. Do you need to initialize it or does MON12 take care of it? Where should you set it (or where does MON12 set it)?

Part 1: On-board time generation.

This part of the lab is similar to the counter program you developed in lab 7. The program must sequentially count from $0 to $F and then restarts from $0 again. The binary count value (0000 to 1111) must be displayed on the four LEDs of the project board. The program must provide a time delay of approximately 1 second between each count so that you can visually observe the LEDs at each count value. Unlike Lab 7, here your delay subroutine must use the 68HC12 free running counter timer system, which can be polled within your subroutine. The program should begin at $4000. The Background section contains information you will need to complete this task.

<u>Preparation:</u>

It is assumed that after the previous labs you can complete the process of developing, uploading, and testing an ASM program on the development board without step-by-step instructions. If you require more detailed instruction, please look back at labs 6-8.

1. Write an ASM program to complete the task described above. Be sure to include a delay subroutine that utilizes the timer system. Assemble and check the program. Save the program once it is complete and operational. Remember to print the .lst file once you have tested it and proven it works correctly.

<u>Testing:</u>

2. Upload your program to the CML12S-DP256 development board and run it to test operation.
3. If your program is correct and your delay is sufficiently long, you should now see the count being displayed on the LEDs. If not, debug your program and repeat. Once your program is correctly counting, <u>record</u> the count sequence in the check off sheet.
4. When you are satisfied that the program is operating correctly, ask the TA to check a demonstration of your program. *Be sure to demonstrate that you have used the timer system for your delay subroutine*. Ask the TA to check off Part 1 on your lab check-off sheet.

Part 2: Interrupts

Starting with the code of part 1, add the ability to handle an externally triggered interrupt coming from the push button switch on the AXM-0295 project board. Your code should implement an IRQ' interrupt service routine (ISR) that sounds the buzzer three times with (approximately) 1 second delay between each buzzer transition, i.e., three buzzer cycles of 1 second 'on' and 1 second 'off'. End your ISR with the RTI instruction. Be sure to set the IRQ interrupt service vector in the location remapped by the MON12 program ($3FF2, $3FF3).

<u>Hints</u>: Start your main code at $4000 and put your ISR at $3000.

Because you will be programming the controller through the MON12 program, it will initialize the stack pointer for you.

Normally, during software development you need to carefully plan use of the stack to insure the stack will never collide with the program code. This is especially true when you are nesting subroutine calls/returns as well as interrupt service routines, as you will do here. In debugging this program, monitoring the stack contents may be necessary but is complicated because of the MON12 program. The SP of MON 12 automatically decreases by one for every instruction. Thus, when pushing something, it thus will subtract one more value from the SP, which is SP=SP-2. When pulling something, the SP will remain the same (-1 for instruction and +1 for pull).

<u>Preparation:</u>

1. Wire pin 15 of AUX1-PORT on the project board to pin 32 of BUS_PORT on the MCU development board so your program can control the buzzer.
2. Write an ASM program to complete the task described above. Be sure to include a delay subroutine that utilizes the timer system. Assemble and check the program. Save the program once it is complete and operational. Remember to print the .lst file once you have tested it and proven it works correctly.

<u>Testing:</u>

3. Press the reset button and then press ENTER when you see the monitor startup message. Upload your program to the CML12S-DP256 development board and run it to test operation.
4. Verify the program generates the proper counting sequence as in Part 1 and <u>record</u> a comment. If the program is not operating as expected, debug and repeat before proceeding.
5. While the program is running, press the push button switch on the project board to initiate an interrupt. Does the program generate the buzzer sound as expected? If not, debug your program and repeat until it is working correctly. <u>Record</u> a comment of your observations.
6. Either during debugging or afterward, use the MON12 breakpoint **BR** command to stop your program at various locations and observe the contents of your stack. See if you can identify the values that are on the stack and confirm subroutine/ISR return addresses are stored correctly. You may be asked to demonstrate this ability during your TA check off.
7. Ask the TA to check a demonstration of your program and check off Part 2 on your lab check-off sheet.

Final Tasks and Notes

- Turn off the power supply and anything else that you might have turned on.
- Disconnect and return the microcontroller interface board to the lab closet.

Discussion Points

As explained in the *Lab Report Guide*, you should address these discussion points in a designated section of your report.

1. After performing Part 1 of this lab, you learned how to use the HC12's free running counter. List some advantages of using the free running counter for delay loops compared to software delay loops that execute a series of instructions.
2. What modifications to your program from Part 2 would be necessary in order to utilize the XIRQ interrupt instead of the IRQ interrupt? Refer to MON12 Manual.
3. What modifications would you need to make if you wanted to use the timer overflow as your delay interrupt?

PRE-LAB 9

Due: At the beginning of lab.

Student Name: ___________________________ Lab. Section (time): __________

Make sure you read the lab document before you start the pre-lab, especially the Background section.

1. Write the code for a delay subroutine using the free running counter system of the 68HCS12. Include all timer configuration settings. Use a counter clock prescale (PR) value of %101. Have the subroutine end after the timer overflows a second time (from reset).

2. Assuming the timer counter starts at ‘0’, calculate the number of timer overflows that will be closest to a 1 second delay if the prescale factor (PR) is set to %101 (scale of 32) and the master clock is 8MHz. Remember the timer counter is 16 bits.

3. Plan both programs needed for this lab by creating either a flowchart or pseudocode for the assigned tasks. Show these to the TA before beginning your lab and then attach them to your lab report.

LAB 9 CHECK-OFF SHEET

Student Name: ___________________________ Lab. Section (time): __________

Complete this sheet as you complete the lab. Remember to have the TA check off each section of the assignment. This sheet must be included in your lab report.

Part 1: On-board time generation.

Step 3. What is the observed binary sequence on the LEDs?

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Is the observed delay sufficient to display the count sequence? ________

Part 1: TA sign off

Part 1: On-board time generation . Initial________

Part 2 Interrupts

Step 4. Is the count sequence the same as in part 1? ________

Step 5. Comment on generation of the buzzer sound ___________________

____

Part 2: TA sign off

Part 2: Interrupts Initial________