EE475 Lab #5   Fall 2003

I/O Ports and Real Time Interrupts

In this lab you will use a real time interrupt and several of the hardware I/O ports on the HC12 EVB. You will write a program to use the LED indicators and the toggle switches on the I/O board while generating a stable digital waveform on an output pin for observation with the oscilloscope.

Preliminaries

1. Be sure to have a copy of the Cady M68HC12 book on hand as a reference for any questions on the port bit assignments, etc.
2. Make a temporary local folder for your work: `c:\EEClasses\EE475\tempxxx`.
3. Launch the Cosmic CPU12 program and make a new `lab5.prj` project file for this week based on the lab4 (or lab3) previous projects.
4. Include the file IOBC32.H in your C source file: `#include <IOBC32.H>` This header file includes the names for the HC12 ports and registers (`PORTA`, `RTICTL`, etc.)
5. Verify that the program start address in the `.lkf` file is 0x1000.

Exercise #1: Blink the LEDs

Create a C program that sequentially blinks each of the 8 LEDs on the I/O board, one at a time, in a continuous loop (you might want to include a delay to make each blink last longer). Pressing the IRQ button should exit the blink loop, print "done", and then trigger a software interrupt (SWI, as in the previous labs) to regain control at the monitor prompt.

To implement this program you will need to:

(a) Create and install the pointer to your UserIRQ interrupt handler.

(b) Port P (PORTP, 0x0056) is connected to the LEDs through a 74LS373 tri-state latch. Set the Port P data direction register (DDRP, 0x0057) so that all eight bits are outputs. To enable the '373 latch, set bit 5 of the CAN port data direction register (DDRCAN, 0x013F) to '1', and set bit 5 of the CAN port (PORTCAN, 0x013E) to '1'.

(c) Note that the Port P bits are connected to the cathodes of the diodes (active low).

(d) Arrange your sequential blink loop to exit properly when the IRQ button is pressed.

→ Demonstrate your blinking LEDs (and IRQ to exit) for the instructor.

Exercise #2: Read the toggle switches and set the LEDs

Now modify your program so that instead of blinking the LEDs, you read the position of the toggle switches and only illuminate the LEDs corresponding to 'on' switches. Put your switch reading statements into a loop so that flipping a switch will quickly change the corresponding LED on/off while your program is running. As before, exit the loop and your program by pressing the IRQ button

To implement this program you will need to:

(a) Note that Port AD (PORTAD, 0x006F) is connected to the switches through a tri-state buffer. The active low enable signal for the buffer is bit 6 of the PCAN register (set bit 6 to '0' in PORTCAN, 0x013E), and the PCAN data direction register must be set so that bit 6 is an output (set bit 6 of DDRCAN to '1').

(b) Determine the proper correspondence between the switches and the LED polarity: a switch in the 'on' position means your program should turn the corresponding LED 'on'.

(c) As before, arrange to exit properly when the IRQ button is pressed.

→ Show the instructor your switch-controlled LEDs.

Exercise #3: Create a square wave using the Real Time Interrupt and Port T

Now modify your program to install an interrupt service routine for the `UserRTI` vector, and activate the HC12's `UserRTI` (Real Time Interrupt) timer hardware.

To enable the real time hardware timer your program needs to:

(a) Set up the Real Time Interrupt Control Register (RTICTL, 0x0014) to enable RTIs (bit 7) and select a 1.024ms interrupt period (bits 2:0).

(b) Set up Port T data direction register (DDRT, 0x00AF) so that bit 0 is an output.

(c) In your UserRTI interrupt service routine, arrange to toggle the state of bit 0 of Port T (PORTT, 0x00AE). In other words, the output should change 0 to 1 during one interrupt, then 1 to 0 during the next interrupt, and so forth.

(d) Arrange to set the RTI flag bit *inside* the interrupt service routine: you must set bit 7 (RTIF) of the RTI register (RTIFLG, 0x0015) to 1 for your program to work properly because the flag is managed by the timer interrupt circuitry.

(e) As before, arrange to exit your loop when the IRQ button is pressed.

Hook up one channel of the oscilloscope between pin 0 of the Timer port and the ground pin. Start your program and observe the output waveform. Verify that the LEDs can still be switched on and off while the waveform is generated.

Exercise #4: Generate a pulse pattern waveform

Modify your program once again: your installed UserRTI routine must generate a serial digital output signal consisting of the sequence of bits set by the toggle switches: bit7, bit6, bit5, …bit0, then again bit7, bit6, … In other words, the UserRTI routine should cycle from one bit to the next on each interrupt, while the background loop continues to scan the toggle switches and set the LEDs.

You might consider using pin 1 of PORTT as a "sync" signal: send a pulse on pin 1 at the start of each pulse pattern going out pin 0. Trigger the scope with the sync signal and observe the waveform with the other scope channel.

→ Demonstrate your groovy eight bit serial pattern generator for the instructor.

RTICTL (Real-Time Interrupt Control Register located at 0x0014)

(You will need to set bit7 and bits2:0)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTIE	RSWAI	RSBCK	0	RTBYP	RTR2	RTR1	RTR0

RTIE – Real-Time Interrupt Enable
0 = Interrupt requests from RTI are disabled (default)
1 = Enable RTI interrupts

RSWAI = 0; RSBCK = 0; RTBYP = 0;

RTR2:RTR0 – Real-time Interrupt Rate Select

RTR2	RTR1	RTR0	Divide M by:	Interrupt Time: (M=8.0 MHz)
0	0	0	Off	Off
0	0	1	2^{13}	1.024 ms
0	1	0	2^{14}	2.048 ms
0	1	1	2^{15}	4.096 ms
1	0	0	2^{16}	8.192 ms
1	0	1	2^{17}	16.384 ms
1	1	0	2^{18}	32.768 ms
1	1	1	2^{19}	65.536 ms

RTIFLG, Real-Time Interrupt Register (located at 0x0015)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTIF	0	0	0	0	0	0	0

NOTE: you must set RTIF=1 inside the interrupt service routine for your program to work properly because the flag is changed by the interrupt circuitry.

PORTP (Port P located at 0x0056)

You will write data to this port to turn the LEDs on and off. Note that LEDs are connected active low.

PORTT (Port T located at 0x00AE)

You will write data to bit 0 of this port in order to change the output from '0' to '1' in a periodic fashion (when each interrupt occurs).

PORTAD (Port AD located at 0x006F)

This port is connected to the toggle switches via a tri-state buffer. If the buffer is enabled, the switch positions can be determined by reading this port. The switches read active low.

PORTCAN (Port located at 0x013E)

Set bit 6 to '0' and bit 5 to '1' in this register to enable the tri-state buffers for the LEDs and switches.

DDRP (Port P data direction register located at 0x0057)

You need to set each bit of this register to determine the direction of the data on the corresponding pin. To make a bit in port P an output bit, you set the bit in DDRP to '1'. In this case turn them ALL into outputs.

DDRT (Port T data direction register located at 0x00AF)

To make a bit in port T an output bit, you set the corresponding bit in DDRT to '1'.

DDRCAN (data direction register located at 0x013F)

Set bits 5 and 6 of this register to '1' so that the corresponding bits in PORTCAN are outputs.