ASEN 5067: Lab 5

William Watkins

Assembly program requirements

Once on startup:

- RD4, RD5, RD6, RD7 off
- "ASEN5067" displayed on LCD first line
- "PW1.00ms" displayed on LCD second line
- In sequence,
  - RD5 on for 0.5 s +/- 10 ms then RD5 off
  - RD6 on for 0.5 s +/- 10 ms then RD6 off
  - RD7 on for 0.5 s +/- 10 ms then RD7 off

Loop forever

note smaller tolerance!

- "Alive LED" RD4 blinks: 200 ms +/- 100 µs on, 800 ms +/- 10 µs off
- PWM on RC2 begins w/ period of 20 ms & 5% duty cycle
  - 5% duty cycle - 1 ms on, 19 ms off
  - Tolerances: On - +/- 100 µs, off - +/- 10 µs
- RPG will change pulse width
  - 1/64th of a turn = +/- 0.01 ms (clockwise = +)
  - Hard limits - 1 ms, 2 ms
  - LCD should update w/ pulse width

Constraints

- Timers used for base timing
  - Should not be preloaded each time

- Use CCP to generate output pulse
- Use another CCP for "Alive LED" RD4
- Must handle timing & CCP in ISRs
- Updating LCD, RPG, etc. must be done in main loop

C Program Req's

- Blink RD4 @ 1Hz, 50% duty cycle
  - Must use Timer0 w/ISR

ASSEMBLY

Low Pri - Turn on

High Pri - Turn off

Low

Pseudocode

Timer 1 for high priority - turning ON

Timer 3 for low priority - turning OFF

ECCP1 for turning Alive OFF - low pri.

- Init w/ time on
- Set ECCP2 = ECCP1
- Add time off to ECCP2

ECCP2 for turning Alive ON - High pri.

- Don't init
- Set ECCP1 = ECCP2
- Add time on to ECCP1

ECCP3 for turning PWM OFF - low pri.

- Don't init
- Set CCP4 = ECCP3
- Add time to CCP4

CCP4 for turning PWM ON - High pri.

- Init w/ time on
- Set ECCP3 = CCP4
- Add time to ECCP3

No!!! - Timer1 = Alive, Timer3 = PWM, change low/high pri when handling!

- High pri for turning on

Outline

- Main
- Loop
- Initial
- Lo Pri.
- Hi Pri.
- ECCP1 Handler
- ECCP2 Handler

- Timer1 Handler
- Timer3 Handler
- RPG checker
- LCD updater V
- LCD updater C
- Wait 05s

Initial

- Setup I/O
- Turn off outputs
- Output LCD stuff
- Use Timer 0 & count to 0.5s for ON & OFF
- Set Timer 1 for High Pri. interrupts
- Set Timer 3 for Low Pri. interrupts
- Configure ECCP1-3, CCP4

Low Pri.

- Check Alive CCP bit first
  - If set, turn off alive LED, reset IF
  - Set other Alive CCP bit
- Check PWM CCP bit
  - If set, turn PWM OFF
- Handle Timer 3

High Pri.

- Check Alive CCP bit
  - If set, tu

Lo Pr: - triggered

- Check if

Scenario: LoPr: for CCP1 triggered, then
HiPr: for CCP2 triggered
Both IF bits set, but only CCP2
is HiPr - need to make sure
HiPr: CCPs are handled, not LoPr:

Hi Pr:

BTFSC

Questions

1. I kept having problems getting my PWM output to work. One upload it would work fine, the next it would be stuck in a reset loop. I believe the problem is due to using Timer1 for both ECCPs. This will be rectified by switching ECCP3 to Timer3 instead of Timer1.

Extra Credit

10.2:

An Interrupt Service Routine (ISR) is a special subroutine designed and written to accomplish a specific task requested by the interrupt request. The only difference between an ISR and a regular subroutine is an ISR must use a special RETURN statement - RETFIE (1). This reenables interrupts!

10.8

The only pins that can accept external interrupt sources are:

- RB0/INT0
- RB1/INT1
- RB2/INT2
- RB3/INT3
- PORTB<7:4>

10.14

The WREG, STATUS, and BSR registers are saved on the stack when a high-priority interrupt is triggered. This is in contrast to a low priority interrupt, which requires those registers to be manually saved.