ECE3411 – Fall 2016

Lab 6a.

EEPROM & Watchdog Timer
Analyzing Assembly Code

Marten van Dijk, Chenglu Jin
Department of Electrical & Computer Engineering
University of Connecticut
Email: {marten.van_dijk, chenglu.jin}@uconn.edu

Copied from Lab 6a, ECE3411 – Fall 2015, by
Marten van Dijk and Syed Kamran Haider

UCONN

Task1: Saving the Program State

Your task is to extend the simple ADC voltage measurement code from Lab5b:Task1 with a watchdog timer:

- The watch dog timer should be set up such that if ADC reads $\geq 4V$ continuously for a period of 4 seconds, only then a system reset must occur.
- Note that if ADC reads $< 4V$ at anytime after reading $\geq 4V$ but before the 4 seconds window has elapsed, then the system reset should not occur.
- Before entering the main loop print "Starting ...." to the terminal (this allows you to see when a system reset actually occurs).
- Print a counter value on LCD where the counter is incremented every second.
- Before the system resets, the watch dog timer ISR should store in EEPROM the current value of the counter. This value should be loaded from EEPROM before entering the main loop and the counter should continue from this value onwards.

Task2: Analyzing Assembly Code

Write a C code to program Timer0 in CTC mode to trigger Compare Match A ISR after every 1ms.

- Increment a global counter inside the Compare Match A ISR
- Use Atmel Studio debugger to see the Assembly code of your program.
- By stepping through the Assembly instructions one by one in the debugger, explain the sequence of branch and jump instructions executed to call the Initialization function and TIMER0_COMPA_vect ISR.
- In particular, how does Interrupt Vector Table help in this regard?