

# Exercise 2: ADC with Keyboard Interrupt

EG-252 Group Design Exercise – Microcontroller Laboratory

Dr K. S. (Joseph) Kim      Dr Chris P. Jobling

September 2014

## I. Overview

For this lab exercise you are provided a sample ADC assembly program given in the appendix. An electronic version of the program is available on the Blackboard site. The program uses interrupt generated by push buttons to trigger an ADC process on the MC9S08AW60 evaluation board. You are to carry out the following two tasks with this exercise:

- Use the sample program to practice on pushbutton with interrupt mechanism and ADC process with the evaluation board.
- Design an equivalent program in C language which can perform the same keyboard interrupt and ADC processing functions as provided by the example assembly program.

This exercise is worth 6 marks. For this exercise you need only demonstrate your program with the evaluation board to *Dr Chris Jobling by Monday, 8 December 2013*. No report is needed.

You can view this document as a web page [HTML](#), [PDF](#) or as a Word Document [.docx](#)

## Appendix

### Sample Program in Assembly

```
1 ;*****
2 ;*          kbi_adc.asm                      *
3 ;*                                              *
4 ;*          MC9S08AW60 Evaluation board keyboard interrupt example      *
```

```

5  ;*      - Switch SW3 onboard connected to Port D pin 3, KBI pin6;      *
6  ;*      - Switch SW4 onboard connected to Port D pin 2, KBI pin5      *
7  ;*                                          *
8  ;*      Function:                                          *
9  ;*      On reset all LEDs will light on. If SW3 or SW4 pressed,      *
10 ;* an interrupt is generated, which set LEDs 0:3 to light on.      *
11 ;*      More interrupts are generated if SW3 or SW4 are pressed.      *
12 ;*****
13
14          INCLUDE          'derivative.inc' ; Include derivative-specific definitions
15
16  FLASH      EQU          $2000
17  RAM         EQU          $0070
18  WATCH      EQU          $1802
19
20          ORG              RAM
21  LED_on      DS.B         1          ; Define a variable VAR_D with a size of 1 byte
22
23  ;Start program after reset
24          ORG              FLASH
25  START_UP
26          LDA              #$00
27          STA              WATCH      ; Turn off the watchdog timer
28
29  ;Init_GPIO init code
30          LDA              #$FF
31          STA              PTFDD
32          MOV              #$0F, LED_on      ; Initialize VAR_D, used to control the LEDs
33          LDA              #$FF
34          STA              PTDPE          ; Port D is enabled with pull-up
35          RSP                      ; Reset stack pointer
36
37  ;Enable interrupt for Keyboard input
38          LDA              #$60
39          STA              KBI1PE          ; KBI1PE: enable KBI function for pins 5 and 6 only
40          BSET              $02, KBI1SC      ; KBI1SC: KBACK=1, to clear KBI flag
41          BSET              $01, KBI1SC      ; KBI1SC: KBIE=1, enable KBI
42
43          CLI                      ; Enable interrupt
44
45  MAINLOOP
46          LDA              LED_on          ; Simple loop
47          BRA              MAINLOOP
48
49  ;Interrupt service routine for a keyboard interrupt generated upon the press of a pushbutton
50  ;with a falling edge (transition from high logic level "1" to low logic level "0")

```

```

51 LED_SWITCH
52         BSET    $02, KBI1SC    ; Clear KBI flag
53         LDA     #8
54         STA     ADC1SC1        ; ADC conversion will start after a number is written
55 ADCLOOP
56         TST     ADC1SC1        ; Check the COC0 bit (conversion complete flag).
57         BPL     ADCLOOP        ; if not complete, wait in the ADC loop.
58         LDA     ADC1RL        ; if complete, read the ADC outcome (digital value)
59         STA     PTFD          ; display over LED bar
60         RTI
61
62 ;INT_VECTOR
63         ORG     $FFD2
64         DC.W    LED_SWITCH
65
66         ORG     $FFFE
67         DC.W    START_UP

```

View on [GitHub](#)