

Exercise 2: ADC with Keyboard Interrupt

EG-252 Group Design Exercise – Microcontroller Laboratory

Dr K. S. (Joseph) Kim

Dr Chris P. Jobling

September 2020

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 8 marks. For this exercise you need only convert the provided assembly language programme to C and submit it for assessment. The assessment asks some additional questions related to the set up of the ADC and its use in the micromouse project.

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 bit 3, KBI pin6;    *
6 ;*      - Switch SW4 onboard connected to Port D bit 2, KBI pin5    *
7 ;*                                           *
8 ;*      Function:                                *
```

```

9  ;*          On reset, all LEDs are off. When either SW3 or SW4 are pressed,  *
10 ;*          then the ADC channel 8 is read and sent to the LEDs.          *
11 ;*****
12
13          INCLUDE          'derivative.inc' ; Include derivative-specific definitions
14
15  FLASH          EQU          $2000
16  RAM              EQU          $0070
17  WATCH          EQU          $1802
18
19  ConvComp        EQU          %10000000      ;Mask for Conversion Complete flag
20
21          ORG              RAM
22  LED_on          DS.B          1              ; Define a variable VAR_D with a
23
24  ;Start program after reset
25
26          ORG              FLASH
27  START_UP
28          LDA              #$00
29          STA              WATCH              ; Turn off the watchdog timer
30
31  ;Init_GPIO init code
32          LDA              #$FF
33          STA              PTFDD
34          MOV              #$0F, LED_on      ; Initialize VAR_D, used to control
35          LDA              #$FF
36          STA              PTDPE              ; Port D is enabled with pull-up
37          RSP              ; Reset stack pointer to $0080
38
39  ;Enable interrupt for Keyboard input
40          LDA              #$60
41          STA              KBI1PE            ; KBI1PE: enable KBI function for pins 5 and 6
42          BSET              $02, KBI1SC      ; KBI1SC: KBACK=1, to clear KBI flag
43          BSET              $01, KBI1SC      ; KBI1SC: KBIE=1, enable KBI
44
45          CLI              ; Enable interrupt
46
47  MAINLOOP
48          LDA              LED_on            ; Simple loop with "dummy" operation
49          BRA              MAINLOOP
50
51  ;Interrupt service routine for a keyboard interrupt generated upon the press of a pushbutton
52  ;with a falling edge (transition from high logic level "1" to low logic level "0")
53  LED_SWITCH
54          BSET              $02, KBI1SC      ; Clear KBI flag

```

```

55             LDA             #8                ; Select analogue input 8 (the blue poten
56             STA             ADC1SC1           ; ADC conversion will start after a number
57 ADCLOOP
58             LDA             ADC1SC1           ;
59             AND             #ConvComp         ; Check the COCO bit (conversion com
60             BEQ             ADCLOOP           ; if not complete, wait in the ADC loop.
61             LDA             ADC1RL            ; if complete, read the ADC outcome (digita
62             STA             PTFD              ; display over LED bar
63             RTI
64
65 ;INT_VECTOR
66             ORG             $FFD2
67             DC.W            LED_SWITCH
68
69             ORG             $FFFE
70             DC.W            START_UP
71
72
73
74
75

```

[View on GitHub](#)