**MA4832 Microprocessor Systems**

**Lab Exercise – Push Button, Buzzer, LED, and Dip Switch**

**You will learn in this session how to**

* How to do the hardware connection according to the diagram given
* How to design the software which interfaces the input/output through GPIO
* How to read in and store a 4-bit word as specified by a dip switch
* How to turn on LEDs
* How use a push button to activate a buzzer

1. Dip Switch

* Schematic of the sample circuit is shown in Fig. 1.
* Program: rd\_portA.s

; rd\_portA.s

; read PortA Bit 4-7 (Pins PA4 - PA7 are connected to dip switch)

; GPIO\_PORTA address

GPIO\_PORTA\_DATA\_R EQU 0x400043FC

GPIO\_PORTA\_DIR\_R EQU 0x40004400

GPIO\_PORTA\_AFSEL\_R EQU 0x40004420

GPIO\_PORTA\_PUR\_R EQU 0x40004510

GPIO\_PORTA\_DEN\_R EQU 0x4000451C

GPIO\_PORTA\_AMSEL\_R EQU 0x40004528

GPIO\_PORTA\_PCTL\_R EQU 0x4000452C

PA\_4567 EQU 0x400043C0 ; PortA bit 4-7

SYSCTL\_RCGCGPIO\_R EQU 0x400FE608 ; GPIO run mode clock gating control

THUMB

AREA DATA, ALIGN=4

EXPORT Result [DATA,SIZE=4]

Result SPACE 4

AREA |.text|, CODE, READONLY, ALIGN=2

THUMB

EXPORT Start

Start

; initialize Port A

; enable digital I/O, ensure alt. functions off

; activate clock for PortA

LDR R1, =SYSCTL\_RCGCGPIO\_R ; R1 = address of SYSCTL\_RCGCGPIO\_R

LDR R0, [R1] ;

ORR R0, R0, #0x01 ; turn on GPIOA clock

STR R0, [R1]

NOP ; allow time for clock to finish

NOP

NOP

; no need to unlock Port A bits

; disable analog mode

LDR R1, =GPIO\_PORTA\_AMSEL\_R

LDR R0, [R1]

BIC R0, R0, #0xF0 ; disable analog mode on PortA bit 4-7

STR R0, [R1]

; configure as GPIO

LDR R1, =GPIO\_PORTA\_PCTL\_R

LDR R0, [R1]

BIC R0, R0,#0x00FF0000 ; clear PortA bit 4 & 5

BIC R0, R0,#0XFF000000 ; clear PortA bit 6 & 7

STR R0, [R1]

; set direction register

LDR R1, =GPIO\_PORTA\_DIR\_R

LDR R0, [R1]

BIC R0, R0, #0xF0 ; set PortA bit 4-7 input (0: input, 1: output)

STR R0, [R1]

; disable alternate function

LDR R1, =GPIO\_PORTA\_AFSEL\_R

LDR R0, [R1]

BIC R0, R0, #0xF0 ; disable alternate function on PortA bit 4-7

STR R0, [R1]

; pull-up resistors on switch pins

LDR R1, =GPIO\_PORTA\_PUR\_R ;

LDR R0, [R1] ;

ORR R0, R0, #0xF0 ; enable pull-up on PortA bit 4-7

STR R0, [R1]

; enable digital port

LDR R1, =GPIO\_PORTA\_DEN\_R

LDR R0, [R1]

ORR R0, R0, #0xF0 ; enable digital I/O on PortA bit 4-7

STR R0, [R1]

LDR R1, =PA\_4567

Loop

LDR R0, [R1] ; R0 = dip switch status

LDR R2, =Result

STR R0,[R2] ; store data

B Loop

ALIGN ; make sure the end of this section is aligned

END ; end of file

* Load and run the program.
* Observe the following points:
  + - * 1. How to read in a 4 bit word via GPIO Port A
        2. Check the value at address 0x20000000, it should give you the value indicated by the dip switch
  1. LEDs
* Schematic of the sample circuit is shown in Fig. 2
* Program: wr\_portB.s

; wr\_portB.s

; output the value to PortB bit 0 - 3 (Pins PB0 - PB3 are connected to LEDs)

; GPIO\_PORTB address

GPIO\_PORTB\_DATA\_R EQU 0x400053FC

GPIO\_PORTB\_DIR\_R EQU 0x40005400

GPIO\_PORTB\_AFSEL\_R EQU 0x40005420

GPIO\_PORTB\_PUR\_R EQU 0x40005510

GPIO\_PORTB\_DEN\_R EQU 0x4000551C

GPIO\_PORTB\_AMSEL\_R EQU 0x40005528

GPIO\_PORTB\_PCTL\_R EQU 0x4000552C

PB\_0123 EQU 0x4000503C ; Port B bit 0-3

SYSCTL\_RCGCGPIO\_R EQU 0x400FE608 ; GPIO run mode clock gating control

AREA |.text|, CODE, READONLY, ALIGN=2

THUMB

EXPORT Start

Start

; initialize Port B, all bits

; enable digital I/O, ensure alt. functions off

; activate clock for Port B

LDR R1, =SYSCTL\_RCGCGPIO\_R ; R1 = address of SYSCTL\_RCGCGPIO\_R

LDR R0, [R1]

ORR R0, R0, #0x02 ; set bit 1 to turn on clock for GPIOB

STR R0, [R1]

NOP ; allow time for clock to finish

NOP

NOP

; no need to unlock Port B bits

; disable analog mode

LDR R1, =GPIO\_PORTB\_AMSEL\_R

LDR R0, [R1]

BIC R0, R0, #0x0F ; Clear bit 0-3, disable analog function

STR R0, [R1]

; configure as GPIO

LDR R1, =GPIO\_PORTB\_PCTL\_R

LDR R0, [R1]

BIC R0, R0,#0x000000FF ; bit clear PortA bit 0 & 1

BIC R0, R0,#0X0000FF00 ; bit clear PortA bit 2 & 3

STR R0, [R1]

; set direction register

LDR R1, =GPIO\_PORTB\_DIR\_R

LDR R0, [R1]

ORR R0, R0, #0x0F ; set PortB bit 0-3 as output (0: input, 1: output)

STR R0, [R1]

; disable alternate function

LDR R1, =GPIO\_PORTB\_AFSEL\_R

LDR R0, [R1]

BIC R0, R0, #0x0F ; disable alternate function on PortB bit 0-3

STR R0, [R1]

; enable digital port

LDR R1, =GPIO\_PORTB\_DEN\_R

LDR R0, [R1]

ORR R0, #0x0F ; enable PortB digital I/O

STR R0, [R1]

LDR R1, =PB\_0123

LDR R0, =0x0F ; set PortB bit 0-3 -> turn on 4 Leds

STR R0, [R1]

BL Delay

LDR R2,=0x0F

Loop

EOR R0, R2 ; R0 = Exclusive OR of R0 with R2

STR R0, [R1] ; clear PortB bit 0-3 -> turn off 4 Leds

BL Delay

B Loop

Delay

MOV R7,#0xFFFFFF

Countdown

SUBS R7, #1 ; subtract and sets the flags based on the result

BNE Countdown

BX LR ; return

ALIGN ; make sure the end of this section is aligned

END ; end of file

* Load and run the program
* Observe the following points:
  + - * 1. How to output a 4 bit word via GPIO Port B
        2. Branching and looping

1. Push Button and Buzzer

* Schematic of the sample circuit is shown in Fig. 3
* Program: prog\_portD.s

; prog\_portD.s

; read portD bit 0 (push button) and output the value to portD bit 3 (buzzer)

; GPIO\_PORTD address

GPIO\_PORTD\_DATA\_R EQU 0x400073FC

GPIO\_PORTD\_DIR\_R EQU 0x40007400

GPIO\_PORTD\_AFSEL\_R EQU 0x40007420

GPIO\_PORTD\_PUR\_R EQU 0x40007510

GPIO\_PORTD\_DEN\_R EQU 0x4000751C

GPIO\_PORTD\_AMSEL\_R EQU 0x40007528

GPIO\_PORTD\_PCTL\_R EQU 0x4000752C

PD EQU 0x40007024 ; Enable Port D bit 0 and 3

SYSCTL\_RCGCGPIO\_R EQU 0x400FE608 ; GPIO run mode clock gating control

AREA |.text|, CODE, READONLY, ALIGN=2

THUMB

EXPORT Start

Start

; initialize Port D

; enable digital I/O, ensure alt. functions off

; activate clock for Port D

LDR R1, =SYSCTL\_RCGCGPIO\_R ; R1 = address of SYSCTL\_RCGCGPIO\_R

LDR R0, [R1] ;

ORR R0, R0, #0x08 ; set bit 3 to turn on clock for GPIOD

STR R0, [R1]

NOP ; allow time for clock to finish

NOP

NOP

; no need to unlock Port D bits

; disable analog mode

LDR R1, =GPIO\_PORTD\_AMSEL\_R

LDR R0, [R1]

BIC R0, R0, #0x09 ; Clear bit 0 and 3 to disable analog function

STR R0, [R1]

; configure as GPIO

LDR R1, =GPIO\_PORTD\_PCTL\_R

LDR R0, [R1]

BIC R0, R0,#0x0000000F ; clear PortA bit 0

BIC R0, R0,#0X0000F000 ; clear PortA bit 3

STR R0, [R1]

; set direction register

LDR R1, =GPIO\_PORTD\_DIR\_R

LDR R0, [R1]

BIC R0, R0, #0x01 ; set PortD bit 0 input

ORR R0, R0, #0x08 ; set PortD bit 3 output (0: input, 1: output)

STR R0, [R1]

; disable alternate function

LDR R1, =GPIO\_PORTD\_AFSEL\_R

LDR R0, [R1]

BIC R0, R0, #0x09 ; disable alternate function on bit 0 and 3

STR R0, [R1]

; pull-up resistors on switch pins

LDR R1, =GPIO\_PORTD\_PUR\_R ; R1 = address of GPIO\_PORTD\_PUR\_R

LDR R0, [R1] ;

ORR R0, R0, #0x01 ; enable pull-up on PortD bit 0

STR R0, [R1]

; enable digital port

LDR R1, =GPIO\_PORTD\_DEN\_R

LDR R0, [R1]

ORR R0, R0, #0x09 ; enable digital I/O on bit 0 and 3

STR R0, [R1]

LDR R1, =PD

Again1

MOV R0, #0

STR R0, [R1] ; "off" buzzer

LDR R2, [R1] ; check switch, PortD bit 0 status

TST R2, #1 ;

BNE Again1 ; perform a bitwise AND operation and test again if ; switch is not pressed

MOV R0, #0x08 ; when switch is pressed, set PortD bit 3 "high" to ; turn on buzzer

STR R0, [R1] ;

Again2

LDR R2, [R1] ; check switch

TST R2, #1 ; perform a bitwise AND operation and test again if ; switch is not released

BEQ Again2

B Again1

ALIGN ; make sure the end of this section is aligned

END ; end of file

* Load and run the program.
* Make sure you understand the following points:
  + 1. How to read the status of a push button switch and turn on a buzzer via

GPIO Port D

* + 1. Branching and looping

**Exercise:**

Combine all three programs into one that uses

* Port A to read in the status of a dip switch
* Port B to output a word specified by the dip switch
* Port D to sound a buzzer when a push button is activated

