**EE447 EXPERIMENT #4**

**PRELIMINARY REPORT**

**Question 1-)**

In pulse.s, timer0a is used periodic, count down mode and 16 bit.

My\_Timer0A\_Handler PROC

LDR R1,=TIMER0\_ICR; Clear the Flag

MOV R0,#0x01;

STR R0,[R1]; Time Out Interrupt clearead

LDR R1,=GPIO\_PORTF\_DATA;

LDR R2,[R1];

CMP R2,#0;

BNE MakeItLow;

MOV R2,#4;

STR R2,[R1];

LDR R1,=TIMER0\_TAILR;

LDR R2,=HIGH;

STR R2,[R1];

BX LR;

MakeItLow MOV R2,#0;

STR R2,[R1];

LDR R1,=TIMER0\_TAILR;

LDR R2,=LOW;

STR R2,[R1];

BX LR

ENDP

**Question 2-)**

**READ.S**

; 16/32 Timer Registers

TIMER1\_CFG EQU 0x40031000 ; Configuration Register

TIMER1\_TAMR EQU 0x40031004 ; Mode Register

TIMER1\_CTL EQU 0x4003100C ; Control Register

TIMER1\_RIS EQU 0x4003101C ; Raw interrupt Status

TIMER1\_ICR EQU 0x40031024 ; Interrupt Clear Register

TIMER1\_TAILR EQU 0x40031028 ; Interval Load Register

TIMER1\_TAMATCHR EQU 0x40031030 ; Match Register

TIMER1\_TAPR EQU 0x40031038 ; Prescaling Divider

TIMER1\_TAR EQU 0x40031048 ; Counter Register

TIMER1\_IMR EQU 0x40031018 ; Defining Interrupt

TIMER1\_TAV EQU 0x40031050 ; To set the timer initial value

;GPIO Registers

;Port B base 0x40005000

GPIO\_PORTB\_IM EQU 0x40005010 ; Interrupt Mask

GPIO\_PORTB\_DIR EQU 0x40005400 ; Port Direction

GPIO\_PORTB\_AFSEL EQU 0x40005420 ; Alt Function enable

GPIO\_PORTB\_DEN EQU 0x4000551C ; Digital Enable

GPIO\_PORTB\_AMSEL EQU 0x40005528 ; Analog enable

GPIO\_PORTB\_PCTL EQU 0x4000552C ; Alternate Functions

GPIO\_PORTB\_PDR EQU 0x40005514 ; Pull down

;System Registers

SYSCTL\_RCGCGPIO EQU 0x400FE608 ; GPIO Gate Control

SYSCTL\_RCGCTIMER EQU 0x400FE604 ; GPTM Gate Control

AREA routines, CODE, READONLY

THUMB

EXPORT READ\_INIT

READ\_INIT PROC

LDR R1, =SYSCTL\_RCGCGPIO ; start GPIO clock

LDR R0, [R1]

ORR R0, R0, #0x02 ; set bit 2 for port B

STR R0, [R1]

NOP ; allow clock to settle

NOP

NOP

LDR R1, =GPIO\_PORTB\_DIR

LDR R0, [R1]

BIC R0, R0, #0x10 ; clear bit 4 for input

STR R0, [R1]

; enable alternate function

LDR R1, =GPIO\_PORTB\_AFSEL

LDR R0, [R1]

ORR R0, R0, #0x10 ; set bit4 for alternate fuction on PB4

STR R0, [R1]

; set alternate function to T1CCP0 (7)

LDR R1, =GPIO\_PORTB\_PCTL

LDR R0, [R1]

ORR R0, R0, #0x00070000 ; set bits 27:24 of PCTL to 7

STR R0, [R1] ; to enable T1CCP0 on PB4

; disable analog

LDR R1, =GPIO\_PORTB\_AMSEL

MOV R0, #0 ; clear AMSEL to diable analog

STR R0, [R1]

LDR R1, =GPIO\_PORTB\_DEN ; enable port digital

LDR R0, [R1]

ORR R0, R0, #0x10

STR R0, [R1]

; Set pull down

; LDR R1, =GPIO\_PORTB\_PDR

; MOV R0, #0x10; set PB4 as pull down

; STR R0, [R1];

;Timer1,A initialization

LDR R1, =SYSCTL\_RCGCTIMER

LDR R2, [R1] ; Start timer 1

ORR R2, R2, #0x02 ; Timer module = bit position (1)

STR R2, [R1]

NOP

NOP

NOP ; allow clock to settle

; disable timer during setup

LDR R1, =TIMER1\_CTL

LDR R2, [R1]

BIC R2, R2, #0x01 ; clear bit 0 to disable Timer 0

STR R2, [R1]

; set to 16bit Timer Mode

LDR R1, =TIMER1\_CFG

MOV R2, #0x04 ; set bits 2:0 to 0x04 for 16bit timer

STR R2, [R1]

; set for edge time and capture mode

LDR R1, =TIMER1\_TAMR; set bit 4 to 0x01 for up counting

MOV R2, #0x07 ; set bit2 to 0x01 for Edge Time Mode,

STR R2, [R1] ; set bits 1:0 to 0x03 for Capture Mode

; set edge detection to both

LDR R1, =TIMER1\_CTL

LDR R2, [R1]

ORR R2, R2, #0x0C ; set bits 3:2 to 0x03

STR R2, [R1]

; set start value

LDR R1, =TIMER1\_TAILR ; counter counts down,

MOV R0, #0xFFFFFFFF ; so start counter at max value

;MOV R0, #0x00000000 ; so start counter at min value

STR R0, [R1]

; Enable timer

LDR R1, =TIMER1\_CTL ;

LDR R2, [R1] ;

ORR R2, R2, #0x01 ; set bit 0 to enable

STR R2, [R1]

BX LR

ENDP

END

**\_\_main.s**

GPIO\_PORTB\_DATA EQU 0x400053FC

; 16/32 Timer Registers

TIMER1\_CFG EQU 0x40031000 ; Configuration Register

TIMER1\_TAMR EQU 0x40031004 ; Mode Register

TIMER1\_CTL EQU 0x4003100C ; Control Register

TIMER1\_RIS EQU 0x4003101C ; Raw interrupt Status

TIMER1\_ICR EQU 0x40031024 ; Interrupt Clear Register

TIMER1\_TAILR EQU 0x40031028 ; Interval Load Register

TIMER1\_TAMATCHR EQU 0x40031030 ; Match Register

TIMER1\_TAPR EQU 0x40031038 ; Prescaling Divider

TIMER1\_TAR EQU 0x40031048 ; Counter Register

TIMER1\_IMR EQU 0x40031018 ; Defining Interrupt

TIMER1\_TAV EQU 0x40031050 ; To set the timer initial value

FIRST EQU 0x20000480

FREQ EQU 0x00F42400 ; Freq 16M

AREA sdata , DATA, READONLY

THUMB

MSG DCB "PULSE WIDTH "

DCB 0x0D

DCB 0x04

MSG1 DCB "PERIOD "

DCB 0x0D

DCB 0x04

MSG2 DCB "DUTY CYCLE % "

DCB 0x0D

DCB 0x04

;LABEL DIRECTIVE VALUE COMMENT

AREA main, READONLY, CODE

THUMB

EXTERN PULSE\_INIT ; Pulse initialization

EXTERN READ\_INIT

EXTERN OutStr

EXTERN CONVRT

EXPORT \_\_main ; Make available

\_\_main

BL READ\_INIT; initialize read

BL PULSE\_INIT; initialize pulse

START

MOV R0,#0; R0 is turn counter

MOV R10, #0

MOV R8, #0

MOV R6, #0

PUSH {R0}

loop LDR R1, =TIMER1\_RIS

LDR R2, [R1]

ANDS R2, #04 ; isolate CAERIS bit

BEQ loop ; if no capture, then loop

LDR R1, =TIMER1\_ICR;

ORR R2, #0x04; by setting CAECINT bit to 1, CAERIS bir is cleared

STR R2, [R1]

LDR R1, =GPIO\_PORTB\_DATA

LDR R2, [R1]

LDR R1, =TIMER1\_TAR ; address of timer register

LDR R0, [R1] ; Get timer register value

CMP R6, #0

BEQ FIRST\_NUMBER

CMP R8, #0

BEQ SECOND\_NUMBER

CMP R10, #0

BEQ THIRD\_NUMBER

FIRST\_NUMBER

CMP R2, #0x10 ;IF sees positive edge, contunie

BNE loop ;if not, go begin

MOV R6, R0

B loop

SECOND\_NUMBER

MOV R8, R0

B loop

THIRD\_NUMBER

MOV R10, R0

B CONTINUE

CONTINUE PUSH {R5, R6, R7, R8}

LDR R5,=MSG

BL OutStr;to write string above definition

POP {R5, R6, R7, R8}

MOV R2, #16

UDIV R6, R6, R2 ; r6 to microsec(us)

UDIV R8, R8, R2 ;r8 to us

UDIV R10, R10, R2 ;r10 to us

SUB R4, R6, R8 ;to find pulse width

PUSH {R5, R6, R7, R8}

LDR R5,=FIRST;

BL CONVRT

LDR R5,=FIRST;

BL OutStr ; write pulse width

POP {R5, R6, R7, R8}

PUSH {R5, R6, R7, R8}

LDR R5,=MSG1

BL OutStr ;write string

POP {R5, R6, R7, R8}

MOV R4, #0

SUB R4, R6, R10 ;to find period

PUSH {R5, R6, R7, R8}

LDR R5,=FIRST;

BL CONVRT

LDR R5,=FIRST;

BL OutStr ;write period

POP {R5, R6, R7, R8}

PUSH {R5, R6, R7, R8}

LDR R5,=MSG2

BL OutStr ;write string

POP {R5, R6, R7, R8}

SUB R4, R6, R10 ;period

MOV R11, #100

SUB R6, R6, R8 ;pulse width

MUL R6, R6, R11

UDIV R4, R6, R4 ;duty cycle

PUSH {R5, R6, R7, R8}

LDR R5,=FIRST;

BL CONVRT

LDR R5,=FIRST;

BL OutStr ;write duty cycle

POP {R5, R6, R7, R8}

B START

ALIGN

END