**Lab 4 Deliverables**

Akaash Chikarmane and Milan Feliciello

Section: 16085

Spring 2016

3/1/16

**Simulation**



**Final Program**

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main.s \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Program written by: Akaash Chikarmane/Milan Feliciello

; Date Created: 2/22/2016

; Last Modified: 2/29/2016

; Section Tuesday 2-3

; Instructor: Ramesh Yerraballi

; Lab number: 4

; Brief description of the program

; If the switch is presses, the LED toggles at 8 Hz

; Hardware connections

; PE1 is switch input (1 means pressed, 0 means not pressed)

; PE0 is LED output (1 activates external LED on protoboard)

;Overall functionality of this system is the similar to Lab 3, with three changes:

;1- initialize SysTick with RELOAD 0x00FFFFFF

;2- add a heartbeat to PF2 that toggles every time through loop

;3- add debugging dump of input, output, and time

; Operation

; 1) Make PE0 an output and make PE1 an input.

; 2) The system starts with the LED on (make PE0 =1).

; 3) Wait about 62 ms

; 4) If the switch is pressed (PE1 is 1), then toggle the LED once, else turn the LED on.

; 5) Steps 3 and 4 are repeated over and over

SWITCH EQU 0x40024004 ;PE0

LED EQU 0x40024008 ;PE1

SYSCTL\_RCGCGPIO\_R EQU 0x400FE608

SYSCTL\_RCGC2\_GPIOE EQU 0x00000010 ; port E Clock Gating Control

SYSCTL\_RCGC2\_GPIOF EQU 0x00000020 ; port F Clock Gating Control

GPIO\_PORTE\_DATA\_R EQU 0x400243FC

GPIO\_PORTE\_DIR\_R EQU 0x40024400

GPIO\_PORTE\_AFSEL\_R EQU 0x40024420

GPIO\_PORTE\_PUR\_R EQU 0x40024510

GPIO\_PORTE\_DEN\_R EQU 0x4002451C

GPIO\_PORTF\_DATA\_R EQU 0x400253FC

GPIO\_PORTF\_DIR\_R EQU 0x40025400

GPIO\_PORTF\_AFSEL\_R EQU 0x40025420

GPIO\_PORTF\_DEN\_R EQU 0x4002551C

NVIC\_ST\_CTRL\_R EQU 0xE000E010

NVIC\_ST\_RELOAD\_R EQU 0xE000E014

NVIC\_ST\_CURRENT\_R EQU 0xE000E018

count EQU 0x131000

countone EQU 50

counttwo EQU 50

FILLone EQU 0xFFFFFFFF

FILLtwo EQU 0xFFFFFFFF

THUMB

AREA DATA, ALIGN=4

SIZE EQU 50

;You MUST use these two buffers and two variables

;You MUST not change their names

;These names MUST be exported

EXPORT DataBuffer

EXPORT TimeBuffer

EXPORT DataPt [DATA,SIZE=4]

EXPORT TimePt [DATA,SIZE=4]

DataBuffer SPACE SIZE\*4

TimeBuffer SPACE SIZE\*4

DataPt SPACE 4

TimePt SPACE 4

ALIGN

PRESERVE8

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

THUMB

EXPORT Start

IMPORT TExaS\_Init

Start BL TExaS\_Init

BL Debug\_Init; running at 80 MHz, scope voltmeter on PD3

; initialize Port E

; initialize Port F

; initialize debugging dump, including SysTick

LDR R1, =SYSCTL\_RCGCGPIO\_R ;initialize port E

LDR R0, [R1]

ORR R0, R0, #0x10

STR R0, [R1]

NOP

NOP

NOP

NOP

LDR R1, =GPIO\_PORTE\_DEN\_R

ORR R0, #0x03

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_AFSEL\_R

BIC R0, #0x03

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_DIR\_R ;pin 0 output, pin 1 input

ORR R0, #0x01

BIC R0, #0x02

STR R0, [R1]

LDR R1, =SYSCTL\_RCGCGPIO\_R

LDR R0, [R1]

ORR R0, #0x20

STR R0, [R1]

NOP

NOP

LDR R1, =GPIO\_PORTF\_DIR\_R

LDR R0, [R0]

ORR R0, #0x04

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_AFSEL\_R

LDR R0, [R1]

AND R0, #0xFB

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_DEN\_R

LDR R0, [R1]

ORR R0, #0x04

STR R0, [R1]

CPSIE I ; TExaS voltmeter, scope runs on interrupts

;loop BL Debug\_Capture

loop

BL Debug\_Capture

BL delay

LDR R1, =GPIO\_PORTF\_DATA\_R

LDR R0,[R1] ;heartbeat off

BIC R0, #0x04

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_DATA\_R

LDR R0, [R1] ;checks if switch is pressed

AND R2,R0, #0x02 ;selects on input bits we need

LSR R2,R2, #1

EOR R2, #0x01

BIC R0, R0, #0x01 ;gets the output into the appropriate place for PF4

ORR R0,R0,R2 ;makes code friednly

LDR R1, =GPIO\_PORTE\_DATA\_R

STR R0, [R1]

BL delay

ORR R0, R0, #0x01 ;consitently turns LED on to toggle the LED

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_DATA\_R

LDR R0, [R1]

ORR R0, #0x04 ;heartbeat on

STR R0,[R1]

;heartbeat

; Delay

;input PE1 test output PE0

B loop

delay

LDR R2, =count

again

SUBS R2,R2, #0x01

BNE again

BX LR

;------------Debug\_Init------------

; Initializes the debugging instrument

; Input: none

; Output: none

; Modifies: none

; Note: push/pop an even number of registers so C compiler is happy

Debug\_Init

PUSH{R14}

PUSH{R0,R12}

LDR R1, =DataBuffer ;initialize pointers by filling them with starting locations

LDR R0, =DataPt

STR R1, [R0]

LDR R1, =TimeBuffer

LDR R0, =TimePt

STR R1, [R0]

LDR R0, =countone ;R0 is the count

LDR R2, =FILLone

LDR R1, =DataPt

LDR R1, [R1]

againinit

;pointer

STR R2, [R1] ;fill DataBuffer array with 0xFFFFFFFF

ADD R1, #0x04

SUB R0, #0x01

CMP R0, #0

BNE againinit

LDR R0, =counttwo

LDR R2, =FILLtwo

LDR R1, =TimePt

LDR R1, [R1]

againinittwo

STR R2, [R1] ;fill TimeBuffer array with 0xFFFFFFFF

ADD R1, #0x04

SUB R0, #0x01

CMP R0, #0

BNE againinittwo

LDR R1, =NVIC\_ST\_CTRL\_R

MOV R0, #0

STR R0, [R1] ;iniitalize systick

LDR R1, =NVIC\_ST\_RELOAD\_R

LDR R0, =0x00FFFFFF

STR R0, [R1]

LDR R1, =NVIC\_ST\_CURRENT\_R

MOV R0, #0

STR R0, [R1]

LDR R1, =NVIC\_ST\_CTRL\_R

MOV R0, #0x05

STR R0, [R1]

POP{R0,R12}

POP{R14}

BX LR

; init SysTick

;------------Debug\_Capture------------

; Dump Port E and time into buffers

; Input: none

; Output: none

; Modifies: none

; Note: push/pop an even number of registers so C compiler is happy

Debug\_Capture

PUSH{R14}

PUSH{R0,R12}

LDR R0, =countone

LDR R1, [R0] ;If we have already filled the array we return to main

SUB R1, #0x01

CMP R1, #0

BLS stop

STR R1, [R0]

LDR R1, =GPIO\_PORTE\_DATA\_R

LDR R0, [R1]

AND R2, R0, #0x02

AND R0, #0x01

LSL R2, #3

ORR R0, R0, R2

LDR R1, =DataPt

LDR R2, [R1]

STR R0, [R2]

ADD R2, #0x04

STR R2, [R1]

LDR R1, =NVIC\_ST\_CURRENT\_R

LDR R0, [R1]

LDR R2, =TimePt

LDR R3, [R2]

STR R0, [R3]

ADD R3, #0x04

STR R3, [R2]

stop

POP{R0,R12}

POP{R14}

BX LR

ALIGN ; make sure the end of this section is aligned

END ; end of file

**Estimation of Execution Time**

Execution: ; 29 instructions in Debug\_capture subroutine @ 2 cycles per instruction

Time between cycles: ;there are 21 instructions (42 cycles), not including the delay, between captures. Including the delay, there are 5,000,042 cycles @ 2 cycles per instruction

Intrusiveness:

**Results of Debugging Instrument**

Actual: 62.489 ms

Calculated:

0x00D9FAD7 = 14,285,527

0x00417A27 = 4,291,111

14285527 – 4291111 = 9994416

((9994416 cycles)/2) = 4997208 cycles ;Divide by two because it stores data/time every 2 periods

4997208 \* 12.5e-9 = 62.4651