

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

; Program written by: Ali Ziyaan Momin and Zain Modi

; Date Created: 10/6/2015

; Last Modified: 10/7/2015

; Section 4-5pm TA: Jenny Lee

; Lab number: 4

; Brief description of the program

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

; Hardware connections

; PE0 is switch input (1 means pressed, 0 means not pressed)

; PE1 is LED output (1 activates external LED on protoboard)

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

;1- activate the PLL to run at 80 MHz (12.5ns bus cycle time)

;2- initialize SysTick with RELOAD 0x00FFFFFF

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

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

; Operation

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

; 2) The system starts with the LED on (make PE1 =1).

; 3) Wait about 62 ms

; 4) If the switch is pressed (PE0 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

TIME\_DELAY EQU 0x000F2300

large\_num EQU 0xFFFFFFFF

MATRIX\_RELOADED EQU 0x00FFFFFF

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

Buffer\_Counter SPACE 4

ALIGN

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

THUMB

EXPORT Start

IMPORT TExaS\_Init

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

; initialize Port E

; initialize Port F

; initialize debugging dump, including SysTick

BL Port\_Init ;subroutine that initializes port E

LDR R1, =GPIO\_PORTE\_DATA\_R ;turn on led via PE1

LDR R0, [R1]

ORR R0, R0, #0x02

STR R0, [R1]

MOV R0, #0x32

LDR R1, =Buffer\_Counter

STR R0, [R1]

BL Debug\_Init

CPSIE I ; TExaS voltmeter, scope runs on interrupts

loop BL Debug\_Capture

;heartbeat

; Delay

;input PE0 test output PE1

BL Delay ;go to delay subroutine

LDR R1, =GPIO\_PORTF\_DATA\_R ;the following steps will turn off the LED via PF1

LDR R0, [R1] ;THIS IS MY HEARTBEAT

EOR R0, R0, #0x04 ;the exclusive or will make the LED turn off/on since 1EOR1 = 0

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_DATA\_R ;the following steps compare PE0 to zero

LDR R0, [R1]

AND R2, R0, #0x01 ;mask data register to obtain value of PE0

CMP R2, #0x01 ;compare PE0 to zero

BNE TON ;if not equal to zero then go to TON (turn on subroutine), this means switch is not pressed

LDR R1, =GPIO\_PORTE\_DATA\_R ;the following steps will turn off the LED via PE1

LDR R0, [R1]

EOR R0, R0, #0x02 ;the exclusive or will make the LED turn off since 1EOR1 = 0

STR R0, [R1] ;store the PE1 modification back to the data register

B DONE

TON ;the section toggles led via PE1 bit being flipped

LDR R1, =GPIO\_PORTE\_DATA\_R

LDR R0, [R1]

ORR R0, #0x02

STR R0, [R1]

DONE

B loop

;------------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 {R0-R11} ;save registers

LDR R2, =DataBuffer

LDR R3, =TimeBuffer ;load buffer address

LDR R5, =DataPt ;load DataPt and TimePt

STR R2, [R5]

LDR R5, =TimePt ;

STR R3, [R5] ;

MOV R4, #0

LDR R5, =large\_num ;put xFFFFFFF in buffers

LOOPER

STR R5, [R2]

STR R5, [R3]

ADD R2, #0x04

ADD R3, #0x04

ADD R4, #0x01

CMP R4, #0x32

BNE LOOPER ;initializing buffers

; init SysTick

;Disable Enable bit

LDR R0, =NVIC\_ST\_CTRL\_R

LDR R1, [R0]

BIC R1, #0x01

STR R1, [R0]

;x00FFFFF into reloaded

LDR R0, =NVIC\_ST\_RELOAD\_R

LDR R1, =MATRIX\_RELOADED

STR R1, [R0]

;Clear current register

LDR R0, =NVIC\_ST\_CURRENT\_R

MOV R1, #0x00

STR R1, [R0]

;Put in 101

LDR R0, =NVIC\_ST\_CTRL\_R

MOV R1, #0x05

STR R1, [R0]

POP {R0-R11}

BX LR

;------------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 ;Around 25 instructions

;Get pointers

PUSH {R0-R11}

LDR R7, =DataPt

LDR R3, [R7]

LDR R8, =TimePt

LDR R4, [R8]

;Get count and compare

LDR R11, =Buffer\_Counter

LDR R0, [R11]

CMP R0, #0

BEQ Big\_Bang

;Load data in data\_buffer and time info in time\_buffer

LDR R1, =GPIO\_PORTE\_DATA\_R

LDR R2, [R1]

AND R9, R2, #0x01

AND R10, R2, #0x02

LSL R9, #4

LSR R10, #1

ADD R2, R9, R10

STR R2, [R3]

LDR R5, =NVIC\_ST\_CURRENT\_R

LDR R5, [R5]

STR R5, [R4]

ADDS R0, #-1

STR R0, [R11]

ADD R3, #4

STR R3, [R7]

ADD R4, #4

STR R4, [R8]

;Restore registers

Big\_Bang

POP {R0-R11}

BX LR

Port\_Init

LDR R1, =SYSCTL\_RCGCGPIO\_R ;activate clock

LDR R0, [R1]

ORR R0, R0, #0x0030

STR R0, [R1]

NOP

NOP

NOP

NOP ;delay to allow time for clock to activate

LDR R1, =GPIO\_PORTE\_DIR\_R ;the subroutine that turns on the led via PE0

LDR R0, [R1]

ORR R0, R0, #0x02

BIC R0, #0x01

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_AFSEL\_R ;turn off alternate function for bits PE0 and PE1

LDR R0, [R1]

BIC R0, R0, #0x03

STR R0, [R1]

LDR R1, =GPIO\_PORTE\_DEN\_R ;digital enable on PE0 and PE1 bits

LDR R0, [R1]

ORR R0, R0, #0x03

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_DIR\_R ;the subroutine that turns on the led via PF2

LDR R0, [R1]

ORR R0, R0, #0x04

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_AFSEL\_R ;turn off alternate function for bits PF2

LDR R0, [R1]

BIC R0, R0, #0x04

STR R0, [R1]

LDR R1, =GPIO\_PORTF\_DEN\_R ;digital enable on PF2

LDR R0, [R1]

ORR R0, R0, #0x04

STR R0, [R1]

BX LR

Delay

LDR R0, =TIME\_DELAY ;move the TIME\_DELAY into R0

Decrement

ADD R0, R0, #-1 ;decrement by 1

CMP R0, #0x00 ;compare to 0

BNE Decrement ;if not equal to zero keep decrementing

BX LR

ALIGN ; make sure the end of this section is aligned

END ; end of file

Intrusiveness

Total Time running = approximately 62ms

Time for Debug\_Capture = 25 instructions \* 2 \* 12.5ns = 625 ns

(625 ns / 62ms) \* 100 = .0010 %

Capture Timing

0x00BF7648 – 0x000B25B0 = 0x00B45098

0x00FFFFFF – 0x00B45098 = 0x004BAF67

0x004BAF67 to decimal = 4,960,103

4,960,103 \* 12.5x(10^(-9)) = 0.06200128 s = 62ms

2(0.06200128) = period = 0.12400256 s = 124ms