EE447 EXPERIMENT #5

PRELIMINARY REPORT

Question 1-) Init_ADC.s

```
; ADC Registers
RCGCADC                 EQU 0x400FE638 ; ADC clock register
;ADC0 base address EQU 0x40038000
ADC0_ACTSS              EQU 0x40038000 ; Sample sequencer (ADC0 base address)
ADC0_RIS                EQU 0x40038004 ; Interrupt status
ADC0_IM                 EQU 0x40038008 ; Interrupt select
ADC0_EMUX               EQU 0x40038014 ; Trigger select
ADC0_PSSI               EQU 0x40038028 ; Initiate sample
ADC0_SSMUX3   EQU 0x400380A0 ; Input channel select
ADC0_SSCTL3   EQU 0x400380A4 ; Sample sequence control
ADC0_SSFIFO3  EQU 0x400380A8 ; Channel 3 results
ADC0_PP                 EQU 0x40038FC4 ; Sample rate
; GPIO Registers
RCGCGPIO                EQU 0x400FE608 ; GPIO clock register
;PORT E base address EQU 0x40024000
PORTE_DEN               EQU 0x4002451C ; Digital Enable
PORTE_PCTL              EQU 0x4002452C ; Alternate function select
PORTE_AFSEL   EQU 0x40024420 ; Enable Alt functions
PORTE_AMSEL   EQU 0x40024528 ; Enable analog
PORTE_DIR               EQU     0x40024400      ;Set direction
                        AREA    routines, CODE, READONLY
                        THUMB
                        EXPORT          Init_ADC
Init_ADC        PROC
                ; Start clocks for features to be used
                        LDR R1, =RCGCADC ; Turn on ADC clock
                        LDR R0, [R1]
                        ORR R0, R0, #0x01 ; set bit 0 to enable ADC0 clock
                        STR R0, [R1]
                        NOP
                        NOP
                        NOP ; Let clock stabilize
                        LDR R1, =RCGCGPIO ; Turn on GPIO clock
                        LDR R0, [R1]
                        ORR R0, R0, #0x10 ; set bit 4 to enable port E clock
                        STR R0, [R1]
                        NOP
                        NOP
                        NOP ; Let clock stabilize
                        LDR R1, =PORTE_AFSEL; Setup GPIO to make PE3 input for ADC0
                        LDR R0, [R1] ; Enable alternate functions
                        ORR R0, R0, #0x08 ; set bit 3 to enable alt functions on PE3
                        STR R0, [R1]
                        LDR R1, =PORTE_DIR
                        LDR R0, [R1]
                        BIC R0, R0, #0x08 ; set bit 3 to input for PE3
                        STR R0, [R1]
                        ; PCTL does not have to be configured
                        ; since ADC0 is automatically selected when
                        ; port pin is set to analog.
                        LDR R1, =PORTE_DEN
                        LDR R0, [R1]    ; Disable digital on PE3
                        BIC R0, R0, #0x08 ; clear bit 3 to disable analog on PE3
                        STR R0, [R1]
```

```
                    LDR R1, =PORTE_AMSEL; Enable analog on PE3
                    LDR R0, [R1]
                    ORR R0, R0, #0x08 ; set bit 3 to enable analog on PE3
                    STR R0, [R1]
                    LDR R1, =ADC0_ACTSS     ; Disable sequencer while ADC setup
                    LDR R0, [R1]
                    BIC R0, R0, #0x08 ; clear bit 3 to disable seq 3
                    STR R0, [R1]
                    ; Select trigger source
                    LDR R1, =ADC0_EMUX
                    LDR R0, [R1]
                    BIC R0, R0, #0xF000 ; clear bits 15:12 to select SOFTWARE
                    STR R0, [R1] ; trigger
                    ; Select input channel
                    LDR R1, =ADC0_SSMUX3
                    LDR R0, [R1]
                    BIC R0, R0, #0x000F ; clear bits 3:0 to select AIN0
                    STR R0, [R1]
                    ; Config sample sequence
                    LDR R1, =ADC0_SSCTL3
                    LDR R0, [R1]
                    ORR R0, R0, #0x06 ; set bits 2:1 (IE0, END0)
                    STR R0, [R1]
                    ; Set sample rate
                    LDR R1, =ADC0_PP
                    LDR R0, [R1]
                    ORR R0, R0, #0x01 ; set bits 3:0 to 1 for 125k sps
                    STR R0, [R1]
                    ; Done with setup, enable sequencer
                    LDR R1, =ADC0_ACTSS
                    LDR R0, [R1]
                    ORR R0, R0, #0x08 ; set bit 3 to enable seq 3
                    STR R0, [R1] ; sampling enabled but not initiated yet
                    BX      LR;
                    ENDP
                    ALIGN
                    END
```

__main.s

```
ADC0_RIS            EQU 0x40038004 ; Interrupt status
ADC0_SSFIFO3    EQU 0x400380A8 ; Channel 3 results
ADC0_PSSI           EQU 0x40038028 ; Initiate sample
ADC0_ISC            EQU     0x4003800C ; ISC
;LABEL          DIRECTIVE       VALUE           COMMENT
                                AREA    main, READONLY, CODE
                                THUMB
                                IMPORT          Init_ADC; Initialize subroutine
                                EXPORT          __main    ; Make available
__main
                                BL      Init_ADC; GPIO & ADC initialized
                                MOV             R6,#0;
getsample           LDR             R1,=ADC0_PSSI; request a sample
                                LDR             R2,[R1];
                                ORR             R2,R2,#0x08; get a sample
                                STR             R2,[R1];
loop                LDR             R1,=ADC0_RIS; check for interrup flag
                                LDR             R2,[R1];
                                ANDS    R2,#0x08;
                                BEQ             loop
                                LDR             R1,=ADC0_ISC; clear the interrupt flag
                                LDR             R2,[R1];
```

```
                ORR     R2,#0x08;
                STR     R2,[R1]; Interrupt flag is cleared
                LDR     R1,=ADC0_SSFIFO3;
                LDR     R2,[R1]; R2 is the data
                B       getsample
                ALIGN
                END
```

Question 2-) __main.s

```
ADC0_RIS        EQU 0x40038004 ; Interrupt status
ADC0_SSFIFO3    EQU 0x400380A8 ; Channel 3 results
ADC0_PSSI       EQU 0x40038028 ; Initiate sample
ADC0_ISC        EQU 0x4003800C ; ISC
;LABEL          DIRECTIVE       VALUE           COMMENT
                AREA    main, READONLY, CODE
                THUMB
                IMPORT          Init_ADC; Initialize subroutine
                EXPORT          __main    ; Make available
__main
                BL      Init_ADC; GPIO & ADC initialized
                MOV             R6,#0;
getsample       LDR             R1,=ADC0_PSSI; request a sample
                LDR             R2,[R1];
                ORR             R2,R2,#0x08; get a sample
                STR             R2,[R1];
loop            LDR             R1,=ADC0_RIS; check for interrup flag
                LDR             R2,[R1];
                ANDS    R2,#0x08;
                BEQ             loop
                LDR             R1,=ADC0_ISC; clear the interrupt flag
                LDR             R2,[R1];
                ORR             R2,#0x08;
                STR             R2,[R1]; Interrupt flag is cleared
                LDR             R1,=ADC0_SSFIFO3;
                LDR             R2,[R1]; R2 is the data
move            MOV             R0,#1241; get the first digit
                UDIV    R7,R2,R0;
                MOV             R1, R7
                MUL             R1,R1,R0;
                SUB             R2,R2,R1; R2 is newed
                MOV             R0,#124; get the second digit
                UDIV    R8,R2,R0;
                MOV             R1, R8
                MUL             R1,R1,R0;
                SUB             R2,R2,R1; R2 is newed
                MOV             R0,#12; get the last digit
                UDIV    R9,R2,R0;
                B               getsample;
                ALIGN
                END
```

Question 3-) __main.s

```
ADC0_RIS        EQU 0x40038004 ; Interrupt status
ADC0_SSFIFO3    EQU 0x400380A8 ; Channel 3 results
ADC0_PSSI       EQU 0x40038028 ; Initiate sample
ADC0_ISC        EQU 0x4003800C ; ISC
;LABEL          DIRECTIVE       VALUE           COMMENT
                AREA    main, READONLY, CODE
                THUMB
                IMPORT          Init_ADC; Initialize subroutine
                IMPORT          OutChar;
```

```
                EXPORT      __main    ; Make available
__main
                BL      Init_ADC; GPIO & ADC initialized
                MOV         R6,#0;
getsample       LDR         R1,=ADC0_PSSI; request a sample
                LDR         R2,[R1];
                ORR         R2,R2,#0x08; get a sample
                STR         R2,[R1];
loop            LDR         R1,=ADC0_RIS; check for interrup flag
                LDR         R2,[R1];
                ANDS    R2,#0x08;
                BEQ         loop
                LDR         R1,=ADC0_ISC; clear the interrupt flag
                LDR         R2,[R1];
                ORR         R2,#0x08;
                STR         R2,[R1]; Interrupt flag is cleared
                LDR         R1,=ADC0_SSFIFO3;
                LDR         R2,[R1]; R2 is the data
                SUB         R0,R2,R6; check sampled data - previous > 0.02
                CMP         R0,#24;
                BGT         move;
                SUB         R0,R6,R2;
                CMP         R0,#24; check previous - sampled data > 0.02
                BLT         getsample;
move            MOV         R6,R2;
                MOV         R0,#1241; get the first digit
                UDIV    R1,R2,R0;
                MOV         R5,R1;
                ADD         R5,R5,#0x30; ascii conversion
                PUSH{R0,R1,R2}
                BL          OutChar; print
                POP{R0,R1,R2}
                MOV         R5,#0x2E; for '.'
                PUSH{R0,R1,R2}
                BL          OutChar; print
                POP{R0,R1,R2}
                MUL         R1,R1,R0;
                SUB         R2,R2,R1; R2 is newed
                MOV         R0,#124; get the second digit
                UDIV    R1,R2,R0;
                MOV         R5,R1;
                ADD         R5,R5,#0x30; ascii conversion
                PUSH{R0,R1,R2}
                BL          OutChar; print
                POP{R0,R1,R2}
                MUL         R1,R1,R0;
                SUB         R2,R2,R1; R2 is newed
                MOV         R0,#12; get the last digit
                UDIV    R1,R2,R0;
                MOV         R5,R1;
                ADD         R5,R5,#0x30; ascii conversion
                PUSH{R0,R1,R2}
                BL          OutChar; print
                POP{R0,R1,R2}
                MOV         R5,#0x0D; for new line
                PUSH{R0,R1,R2}
                BL          OutChar; print
                POP{R0,R1,R2}
                B           getsample;
                ALIGN
                END
```