

Riley Ruckman  
TCES 460, Wi20  
Lab 9

**Note:** The temperature displayed on the 7-seg display is in Fahrenheit, instead of Celsius. As seen in the following code, this is done by the simple conversion equation:  $T_F = (T_C \times (9/5)) + 32$ .

Video Link:

<https://drive.google.com/file/d/1R5vHnXaYbAi5PQAxVoYYLr0ADf-GAOvY/view?usp=sharing>

```
// Riley Ruckman
// TCES460, Wi21
// Lab 9
```

```
#include <stdint.h>
#include <stdio.h>
```

```
/*
```

```
*/
```

```
/*
```

```
7-Segment Display
```

```
*/
```

```
/*
```

```
*/
```

```
/// Create definition for PORT B registers
```

```
#define GPIO_PORTB_DATA_R    (*((volatile unsigned long *)0x400053FC))
#define GPIO_PORTB_DIR_R     (*((volatile unsigned long *)0x40005400))
#define GPIO_PORTB_PUR_R     (*((volatile unsigned long *)0x40005510))
#define GPIO_PORTB_DEN_R     (*((volatile unsigned long *)0x4000551C))
#define GPIO_PORTB_CR_R      (*((volatile unsigned long *)0x40005524))
#define GPIO_PORTB_AMSEL_R   (*((volatile unsigned long *)0x40005528))
#define GPIO_PORTB_PCTL_R    (*((volatile unsigned long *)0x4000552C))
#define SYSCTL_RCGC2_R       (*((volatile unsigned long *)0x400FE108))
```

```
//Create definitions for Port E registers
```

```
#define GPIO_PORTE_DATA_R    (*((volatile unsigned long *)0x400243FC))
#define GPIO_PORTE_DIR_R     (*((volatile unsigned long *)0x40024400))
#define GPIO_PORTE_PUR_R     (*((volatile unsigned long *)0x40024510))
#define GPIO_PORTE_DEN_R     (*((volatile unsigned long *)0x4002451C))
#define GPIO_PORTE_CR_R      (*((volatile unsigned long *)0x40024524))
#define GPIO_PORTE_AMSEL_R   (*((volatile unsigned long *)0x40024528))
```

```

#define GPIO_PORTE_AFSEL_R  (*((volatile unsigned long *)0x40024420))
#define GPIO_PORTE_PCTL_R   (*((volatile unsigned long *)0x4002452C))

////////////////////////////////////

/*

*/

/*
    ADC
*/

/*

*/

//Create definitions for ADC0 registers
#define SYSCTL_RCGCADC_R      (*((volatile unsigned long *)0x400FE638))
#define ADC0_ACTSS_R          (*((volatile unsigned long *)0x40038000))
#define ADC0_EMUX_R           (*((volatile unsigned long *)0x40038014))
#define ADC0_SSMUX3_R         (*((volatile unsigned long *)0x400380A0))
#define ADC0_SSCTL3_R         (*((volatile unsigned long *)0x400380A4))
#define ADC0_PSSI_R           (*((volatile unsigned long *)0x40038028))
#define ADC0_RIS_R            (*((volatile unsigned long *)0x40038004))
#define ADC0_SSFIFO3_R        (*((volatile unsigned long *)0x400380A8))
#define ADC0_ISC_R            (*((volatile unsigned long *)0x4003800C))
#define ADC0_PC_R             (*((volatile unsigned long *)0x40038FC4))

#define ADC0_IM_R             (*((volatile unsigned long *)0x40038008))
#define NVIC_EN0_R            (*((volatile unsigned long *)0xE000E100))

#define NVIC_PRI4_R           (*((volatile unsigned long *)0xE000E410))

//Create definitions for PORT D registers
#define SYSCTL_RCGC2_R        (*((volatile unsigned long *)0x400FE108))
#define GPIO_PORTD_DATA_R     (*((volatile unsigned long *)0x400073FC))
#define GPIO_PORTD_DIR_R      (*((volatile unsigned long *)0x40007400))
#define GPIO_PORTD_AFSEL_R    (*((volatile unsigned long *)0x40007420))
#define GPIO_PORTD_PUR_R      (*((volatile unsigned long *)0x40007510))
#define GPIO_PORTD_DEN_R      (*((volatile unsigned long *)0x4000751C))
#define GPIO_PORTD_LOCK_R     (*((volatile unsigned long *)0x40007520))
#define GPIO_PORTD_CR_R       (*((volatile unsigned long *)0x40007524))
#define GPIO_PORTD_AMSEL_R    (*((volatile unsigned long *)0x40007528))

```



```

void timer0A_delayMs(int ttime);
void timer0_Init(void);
void PortBE_Init(void);
void Display(int digit, int number);
void NumSplit(int counted);
void ADC0SS3_Handler(void);

static int numbers[10] = {0x40,0x79,0x24,0x30,0x19, // Each value turns on bits needed
0x12,0x02,0x78,0x00,0x10}; // to show numbers in display
static int digit1, digit2, digit3, digit4; // Number to be displayed in each digit
static int temperature = 0;

////////////////////////////////////

void timer0_Init(void){

    SYSCTL_RCGCTIMER_R |= 0x01; /* enable clock to Timer0 */
    TIMER0_CTL_R = 0x00; /* disable Timer before initialization */
    TIMER0_CFG_R = 0x04; /* 16-bit option */
    TIMER0_TAMR_R = 0x02; /* periodic mode and down-counter */
    TIMER0_TAILR_R = 50000 - 1; /* Timer A interval load value register */
    TIMER0_ICR_R = 0x1; /* clear the TimerA timeout flag*/
    TIMER0_CTL_R |= 0x01; /* enable Timer A after initialization */
    TIMER0_TAPR_R = 0; // Prescaler value.. Can extend the cycle time max 256 times
}

void ADC_Init(void){

    SYSCTL_RCGC2_R |= 0x08;
    SYSCTL_RCGCADC_R |= 0x01; // Enable clock ADC0

    /* initialize PD1 for AIN6 input */
    GPIO_PORTD_AFSEL_R |= 0x2; /* enable alternate function */
    GPIO_PORTD_DEN_R &= ~0x2; /* disable digital function */
    GPIO_PORTD_AMSEL_R |= 0x2; /* enable analog function */

    /* initialize ADC0 */
    ADC0_ACTSS_R &= ~0x8; /* disable SS3 during configuration */
    ADC0_EMUX_R = 0x5000; /* timer trigger conversion */
    ADC0_SSMUX3_R = 0x6; /* get input from channel 6 */
    ADC0_SSCTL3_R |= 0x6; /* take one sample at a time, set flag at 1st sample */
    ADC0_PC_R = 0x00; /* sets sampling rate to 125 kHz */

    ADC0_IM_R |= 0x08; /* enables interrupt mask for SS3 in ADC0 */

```

```

NVIC_EN0_R = 0x00020000; /* Enable interrupt 17 in NVIC */
NVIC_PRI4_R = (NVIC_PRI4_R & 0xFFFF0FFF) | 0x00004000; /* priority 2 */
ADC0_ACTSS_R |= 0x8; /* enable ADC0 sequencer 3 */

// Initialize Timer1
SYSCTL_RCGCTIMER_R |= 0x02; /* enable clock to Timer1 */

TIMER1_CTL_R = 0x00; /* disable Timer before initialization */
TIMER1_CFG_R = 0x00; /* 32-bit option */
TIMER1_TAMR_R = 0x02; /* periodic mode and down-counter */
TIMER1_TAILR_R = 500000000 - 1; /* Timer A interval load value register */
TIMER1_ICR_R = 0x1; /* clear the Timer A timeout flag */
TIMER1_TAPR_R = 0; /* Prescaler
value.. Can extend the cycle time max 255 times

TIMER1_CTL_R |= 0x21; /* enable Timer A after initialization */
}

// Subroutine to initialize ports B, E
void PortBE_Init(void){

    SYSCTL_RCGC2_R |= 0x00000002; // Port B clock initialized
    GPIO_PORTB_CR_R = 0x7F; // Allow changes to PB6-PB0
    GPIO_PORTB_AMSEL_R = 0x00; // Disable analog function
    GPIO_PORTB_PCTL_R = 0x00000000; // GPIO clear bit PCTL
    GPIO_PORTB_DIR_R = 0x7F; // Set PB6-PB0 outputs
    GPIO_PORTB_PUR_R = 0x00; // Enable pullup resistors on PB4,PF0
    GPIO_PORTB_DEN_R = 0x7F; // 7) Enable digital pins PB6-PB0

    SYSCTL_RCGC2_R |= 0x00000010; // Port E clock initialized
    GPIO_PORTE_CR_R = 0x0F; // Allow changes to PE4-0
    GPIO_PORTE_AMSEL_R = 0x00; // Disable analog function
    GPIO_PORTE_PCTL_R = 0x00000000; // GPIO clear bit PCTL
    GPIO_PORTE_DIR_R = 0x0F; // PE3-PE0 output
    GPIO_PORTE_PUR_R = 0x00; // Disable pullup resistors
    GPIO_PORTE_DEN_R = 0x0F; // Enable digital pins PE3-PE0
}

int main(){

    // initialize necessary ports/modules
    PortBE_Init();
    ADC_Init();
    while(1){

```

```

        // Splits current temperature value into 4 digits
        NumSplit(temperature);
        timer0A_delayMs(2);

        // Displays temperature on 7-seg display
        Display(1,digit4);
        timer0A_delayMs(2);
        Display(2,digit3);
        timer0A_delayMs(2);
        Display(4,digit2);
        timer0A_delayMs(2);
        Display(8,digit1);
        timer0A_delayMs(2);
    }
}

void ADC0SS3_Handler(void) {

    //temperature = ((ADC0_SS_FIFO3_R & 0xFFF) - 500)/10;
    // Converts 12-bit conversion result in SS3's FIFO to millivolts, then Celsius, and lastly
    Fahrenheit
    temperature = (int)((((ADC0_SS_FIFO3_R & 0xFFF) * (3.3/4096.0) * 1000.0) - 500) / 10)
    * (9.0/5.0)) + 32);
    ADC0_ISC_R |= 0x8; /* clear completion flag */
    TIMER1_ICR_R = 0x01; /* clear the TimerA timeout flag */
}

void Display(int digit, int number){

    GPIO_PORTB_DATA_R = 0x00; // Turns off LEDs
    GPIO_PORTC_DATA_R = digit; // Selects digit
    GPIO_PORTB_DATA_R = numbers[number]; // Turns on number in selected digit
    timer0A_delayMs(2);
}

// Splits number in counter into 4 separate numbers for each digit
void NumSplit(int counted){

    digit1 = counted%10; //Copies value in counter, divides it by 10 and then keeps
    remainder
    counted /= 10; //Dividing value in counter by 10 shifts it by one decimal
    digit2 = counted%10;
    counted /= 10;
    digit3 = counted%10;

```

```

        counted /= 10;
        digit4 = counted%10; // digit1: holds 1000's digit, digit2: 100's digit,
        counted /=10; // digit3: 10th digit, digit4: unit digit
    }

void timer0A_delayMs(int ttime) {

    timer0_Init();
    int i;
    for(i = 0; i < ttime; i++) {
        while ((TIMER0_RIS_R & 0x01) == 0); /* wait for TimerA timeout flag */
        TIMER0_ICR_R = 0x01; /* clear the TimerA timeout flag */
    }
}

```

```

1 // Riley Ruckman
2 // TCES460, Wi21
3 // Lab 9
4
5 #include <stdint.h>
6 #include <stdio.h>
7
8 /*
9 /*
10 /*
11
12 /// Create definition for PORT B registers
13 #define GPIO_PORTB_DATA_R ((volatile unsigned long *)0x400053FC))
14 #define GPIO_PORTB_DIR_R ((volatile unsigned long *)0x40005400))
15 #define GPIO_PORTB_PUR_R ((volatile unsigned long *)0x40005510))
16 #define GPIO_PORTB_DEN_R ((volatile unsigned long *)0x4000551C))
17 #define GPIO_PORTB_CR_R ((volatile unsigned long *)0x40005524))
18 #define GPIO_PORTB_AMSEL_R ((volatile unsigned long *)0x40005528))
19 #define GPIO_PORTB_PCTL_R ((volatile unsigned long *)0x4000552C))
20 #define SYSCCTL_RCGC2_R ((volatile unsigned long *)0x400FE108))
21
22 //Create definitions for Port E registers
23 #define GPIO_PORTE_DATA_R ((volatile unsigned long *)0x400243FC))
24 #define GPIO_PORTE_DIR_R ((volatile unsigned long *)0x40024400))
25 #define GPIO_PORTE_PUR_R ((volatile unsigned long *)0x40024510))
26 #define GPIO_PORTE_DEN_R ((volatile unsigned long *)0x4002451C))
27 #define GPIO_PORTE_CR_R ((volatile unsigned long *)0x40024524))
28 #define GPIO_PORTE_AMSEL_R ((volatile unsigned long *)0x40024528))
29 #define GPIO_PORTE_AFSEL_R ((volatile unsigned long *)0x40024420))
30 #define GPIO_PORTE_PCTL_R ((volatile unsigned long *)0x4002452C))
31
32 ///////////////////////////////////////////////////
33
34 /*
35 /*
36 /*
37
38 //Create definitions for ADC0 registers
39 #define SYSCCTL_RCGCADC_R ((volatile unsigned long *)0x400FE638))
40 #define ADC0_ACTSS_R ((volatile unsigned long *)0x40038000))
41 #define ADC0_EMUX_R ((volatile unsigned long *)0x40038014))
42 #define ADC0_SSMUX3_R ((volatile unsigned long *)0x400380A0))
43 #define ADC0_SSCTL3_R ((volatile unsigned long *)0x400380A4))
44 #define ADC0_PSSI_R ((volatile unsigned long *)0x40038028))
45 #define ADC0_RIS_R ((volatile unsigned long *)0x40038004))
46 #define ADC0_SSIF03_R ((volatile unsigned long *)0x400380A8))
47 #define ADC0_ISC_R ((volatile unsigned long *)0x4003800C))
48 #define ADC0_PC_R ((volatile unsigned long *)0x40038FC4))
49
50 #define ADC0_IM_R ((volatile unsigned long *)0x40038008))
51 #define NVIC_EN0_R ((volatile unsigned long *)0xE000E100))
52 #define NVIC_PRI4_R ((volatile unsigned long *)0xE000E410))
53
54 //Create definitions for PORT D registers
55 #define SYSCCTL_RCGC2_R ((volatile unsigned long *)0x400FE108))
56 #define GPIO_PORTD_DATA_R ((volatile unsigned long *)0x400073FC))
57 #define GPIO_PORTD_DIR_R ((volatile unsigned long *)0x40007400))
58 #define GPIO_PORTD_AFSEL_R ((volatile unsigned long *)0x40007420))
59 #define GPIO_PORTD_PUR_R ((volatile unsigned long *)0x40007510))

```



```

60 #define GPIO_PORTD_DEN_R      (*((volatile unsigned long *)0x4000751C))
61 #define GPIO_PORTD_LOCK_R    (*((volatile unsigned long *)0x40007520))
62 #define GPIO_PORTD_CR_R      (*((volatile unsigned long *)0x40007524))
63 #define GPIO_PORTD_AMSEL_R   (*((volatile unsigned long *)0x40007528))
64 #define GPIO_PORTD_PCTL_R    (*((volatile unsigned long *)0x4000752C))
65
66 //Create definitions for Timer1 registers
67 #define TIMER1_CFG_R          (*((volatile unsigned long *)0x40031000))
68 #define TIMER1_TAMR_R         (*((volatile unsigned long *)0x40031004))
69 #define TIMER1_CTL_R          (*((volatile unsigned long *)0x4003100C))
70 #define TIMER1_RIS_R          (*((volatile unsigned long *)0x4003101C))
71 #define TIMER1_ICR_R          (*((volatile unsigned long *)0x40031024))
72 #define TIMER1_TAILR_R        (*((volatile unsigned long *)0x40031028))
73 #define TIMER1_TAPR_R         (*((volatile unsigned long *)0x40031038))
74
75 ///////////////////////////////////////////////////
76
77 /*                                                    */
78 /*                      Normal-Use Timer              */
79 /*                                                    */
80
81 //Create definitions for Timer0 registers
82 #define TIMER0_CFG_R          (*((volatile unsigned long *)0x40030000))
83 #define TIMER0_TAMR_R         (*((volatile unsigned long *)0x40030004))
84 #define TIMER0_CTL_R          (*((volatile unsigned long *)0x4003000C))
85 #define TIMER0_RIS_R          (*((volatile unsigned long *)0x4003001C))
86 #define TIMER0_ICR_R          (*((volatile unsigned long *)0x40030024))
87 #define TIMER0_TAILR_R        (*((volatile unsigned long *)0x40030028))
88 #define TIMER0_TAPR_R         (*((volatile unsigned long *)0x40030038))
89 #define SYSCTL_RCGCTIMER_R    (*((volatile unsigned long *)0x400FE604))
90
91 ///////////////////////////////////////////////////
92
93 /*                                                    */
94 /*      Function Prototypes & Global Variable Initialization      */
95 /*                                                    */
96
97 void ADC_Init(void);
98 void timer0A_delayMs(int ttime);
99 void timer0_Init(void);
100 void PortBE_Init(void);
101 void Display(int digit, int number);
102 void NumSplit(int counted);
103 void ADCOSS3_Handler(void);
104
105 static int numbers[10] = {0x40,0x79,0x24,0x30,0x19, // Each value turns on bits needed
106 0x12,0x02,0x78,0x00,0x10}; // to show numbers in display
107 static int digit1, digit2, digit3, digit4; // Number to be displayed in each digit
108 static int temperature = 0;
109
110 ///////////////////////////////////////////////////
111
112 void timer0_Init(void){
113
114     SYSCTL_RCGCTIMER_R |= 0x01; /* enable clock to Timer0 */
115     TIMER0_CTL_R = 0x00; /* disable Timer before initialization */
116     TIMER0_CFG_R = 0x04; /* 16-bit option */

```

```

117     TIMER0_TAMR_R = 0x02; /* periodic mode and down-counter */
118     TIMER0_TAILR_R = 50000 - 1; /* Timer A interval load value register */
119     TIMER0_ICR_R = 0x1; /* clear the TimerA timeout flag*/
120     TIMER0_CTL_R |= 0x01; /* enable Timer A after initialization */
121     TIMER0_TAPR_R = 0; // Prescaler value.. Can extend the cycle time max 256 times
122 }
123
124 void ADC_Init(void){
125
126     SYSCTL_RCGC2_R |= 0x08;
127     SYSCTL_RCGCADC_R |= 0x01; // Enable clock ADC0
128
129     /* initialize PD1 for AIN6 input */
130     GPIO_PORTD_AFSEL_R |= 0x2; /* enable alternate function */
131     GPIO_PORTD_DEN_R &= ~0x2; /* disable digital function */
132     GPIO_PORTD_AMSEL_R |= 0x2; /* enable analog function */
133
134     /* initialize ADC0 */
135     ADC0_ACTSS_R &= ~0x8; /* disable SS3 during configuration */
136     ADC0_EMUX_R = 0x5000; /* timer trigger conversion */
137     ADC0_SSMUX3_R = 0x6; /* get input from channel 6 */
138     ADC0_SSCTL3_R |= 0x6; /* take one sample at a time, set flag at 1st sample */
139     ADC0_PC_R = 0x00; /* sets sampling rate to 125 kHz */
140
141     ADC0_IM_R |= 0x08; /* enables interrupt mask for SS3 in ADC0 */
142     NVIC_ENO_R = 0x00020000; /* Enable interrupt 17 in NVIC */
143     NVIC_PRI4_R = (NVIC_PRI4_R & 0xFFFF0FFF) | 0x00004000; /* priority 2 */
144     ADC0_ACTSS_R |= 0x8; /* enable ADC0 sequencer 3 */
145
146     // Initialize Timer1
147     SYSCTL_RCGCTIMER_R |= 0x02; /* enable clock to Timer1 */
148
149     TIMER1_CTL_R = 0x00; /* disable Timer before initialization */
150     TIMER1_CFG_R = 0x00; /* 32-bit option */
151     TIMER1_TAMR_R = 0x02; /* periodic mode and down-counter */
152     TIMER1_TAILR_R = 500000000 - 1; /* Timer A interval load value register */
153     TIMER1_ICR_R = 0x1; /* clear the Timer A timeout flag*/
154     TIMER1_TAPR_R = 0; // Prescaler value.. Can extend the cycle time max 255 times
155
156     TIMER1_CTL_R |= 0x21; /* enable Timer A after initialization */
157 }
158
159 // Subroutine to initialize ports B, E
160 void PortBE_Init(void){
161
162     SYSCTL_RCGC2_R |= 0x00000002; // Port B clock initialized
163     GPIO_PORTB_CR_R = 0x7F; // Allow changes to PB6-PB0
164     GPIO_PORTB_AMSEL_R = 0x00; // Disable analog function
165     GPIO_PORTB_PCTL_R = 0x00000000; // GPIO clear bit PCTL
166     GPIO_PORTB_DIR_R = 0x7F; // Set PB6-PB0 outputs
167     GPIO_PORTB_PUR_R = 0x00; // Enable pullup resistors on PB4,PF0
168     GPIO_PORTB_DEN_R = 0x7F; // 7) Enable digital pins PB6-PB0
169
170     SYSCTL_RCGC2_R |= 0x00000010; // Port E clock initialized
171     GPIO_PORTE_CR_R = 0x0F; // Allow changes to PE4-0
172     GPIO_PORTE_AMSEL_R = 0x00; // Disable analog function
173     GPIO_PORTE_PCTL_R = 0x00000000; // GPIO clear bit PCTL

```



```

174 GPIO_PORTB_DIR_R = 0x0F; // PE3-PE0 output
175 GPIO_PORTB_PUR_R = 0x00; // Disable pullup resistors
176 GPIO_PORTB_DEN_R = 0x0F; // Enable digital pins PE3-PE0
177 }
178
179 int main() {
180     // initialize necessary ports/modules
181     PortBE_Init();
182     ADC_Init();
183     while(1) {
184         // Splits current temperature value into 4 digits
185         NumSplit(temperature);
186         timer0A_delayMs(2);
187
188         // Displays temperature on 7-seg display
189         Display(1,digit4);
190         timer0A_delayMs(2);
191         Display(2,digit3);
192         timer0A_delayMs(2);
193         Display(4,digit2);
194         timer0A_delayMs(2);
195         Display(8,digit1);
196         timer0A_delayMs(2);
197     }
198 }
199
200 void ADCOSS3_Handler(void) {
201     //temperature = ((ADC0_SSIFIFO3_R & 0xFFF) - 500)/10;
202     // Converts 12-bit conversion result in SS3's FIFO to millivolts, then Celsius, and lastly Fahrenheit
203     temperature = (int) (((((ADC0_SSIFIFO3_R & 0xFFF) * (3.3/4096.0) * 1000.0) - 500) / 10) * (9.0/5.0)) + 32);
204     ADC0_ISC_R |= 0x8; /* clear completion flag */
205     TIMER1_ICR_R = 0x01; /* clear the TimerA timeout flag */
206 }
207
208 void Display(int digit, int number) {
209     GPIO_PORTB_DATA_R = 0x00; // Turns off LEDs
210     GPIO_PORTB_DATA_R = digit; // Selects digit
211     GPIO_PORTB_DATA_R = numbers[number]; // Turns on number in selected digit
212     timer0A_delayMs(2);
213 }
214
215 // Splits number in counter into 4 separate numbers for each digit
216 void NumSplit(int counted) {
217     digit1 = counted%10; //Copies value in counter, divides it by 10 and then keeps remainder
218     counted /= 10; //Dividing value in counter by 10 shifts it by one decimal
219     digit2 = counted%10;
220     counted /= 10;
221     digit3 = counted%10;
222     counted /= 10;
223     digit4 = counted%10; // digit1: holds 1000's digit, digit2: 100's digit,
224     counted /= 10; // digit3: 10th digit, digit4: unit digit
225 }
226
227 void timer0A_delayMs(int ttime) {
228     timer0_Init();
229     int i;
230     for(i = 0; i < ttime; i++) {
231         while ((TIMER0_RIS_R & 0x01) == 0); /* wait for TimerA timeout flag */
232         TIMER0_ICR_R = 0x01; /* clear the TimerA timeout flag */
233     }
234 }
235
236
237
238
239
240

```