ECE 2049 – Laboratory Report

Embedded Computing in Engineering Design

Worcester Polytechnic Institute

C-Term 2021

Section: C03

Laboratory 3

Making a Time and Temperature Display

Submitted by:

Jonathan R. Lopez

ECE Mailbox #178

3/3/2021

Professor: Yarkin Doroz

Co-Instructor: Fatemeh (Saba) Ganji

<u>Introduction:</u>

The purpose of lab 3 is to make a time and temperature display using our MSP430F5529, our keypad, our buzzer and our SHRP128 LCD screen. We will be using the ADC converter on the MSP430 to display the temperature from the onboard temperature sensor. We will also display the time with some decimal to ASCII conversion to display the time.

<u>Materials:</u>

TI MSP430F5529 Launchpad based lab board.
Jumper wires
Buzzer
Breadboard
Sharp128 LCD Screen
0-9, * and # Keypad
Micro-USB to USB-A cable
Computer with Code Composer Studio ver. 10.2

<u>Discussion and Results:</u>

Getting started:

With the MSP430F5529 already wired up from Lab 0, Lab 1 and Lab2, we got right into launching Code Composer Studio (CCS).

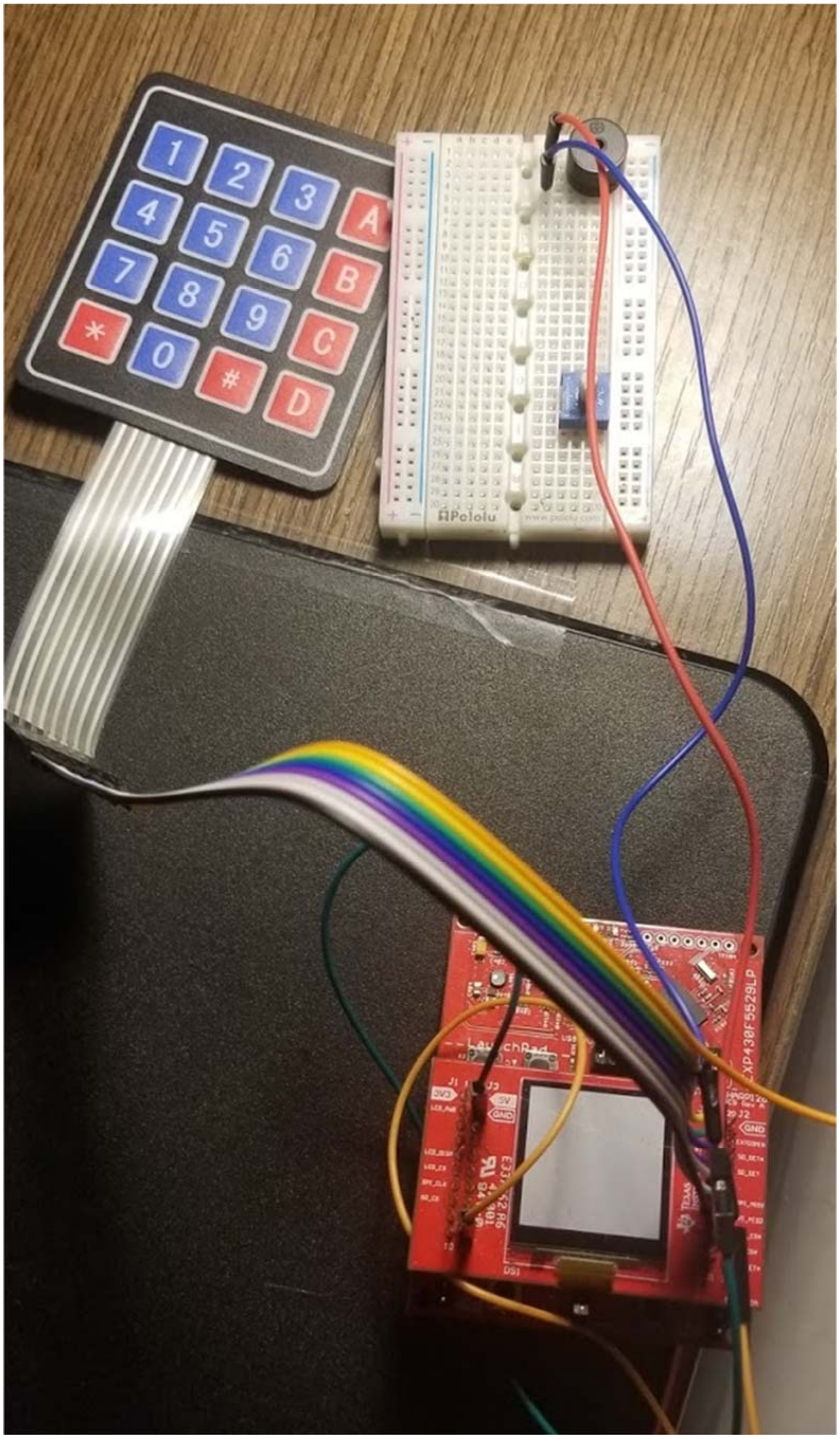


Figure 1: MSP430F5529 already wired up and ready for Lab 3

<u>Approach:</u>

Pre-Lab:

Explain why it is important to pass a copy of the time into the function rather than just using the global variable.

It's a good idea to pass a copy of the timer into the functions just in case the time variable gets screwed up in those functions and then your whole system time is off. This could lead to interrupt errors and others.

Within the *displayTime(long unsigned int inTime)* function there is two helper functions to determine the month and day of the month.

Through dividing and multiplying the remainder by the day, min and second conversions we are able to get the time into an array of chars (dateFormat). Then this is displayed with the display function and flashed to the screen.

The helpers can be found in Figures 3 and 4.

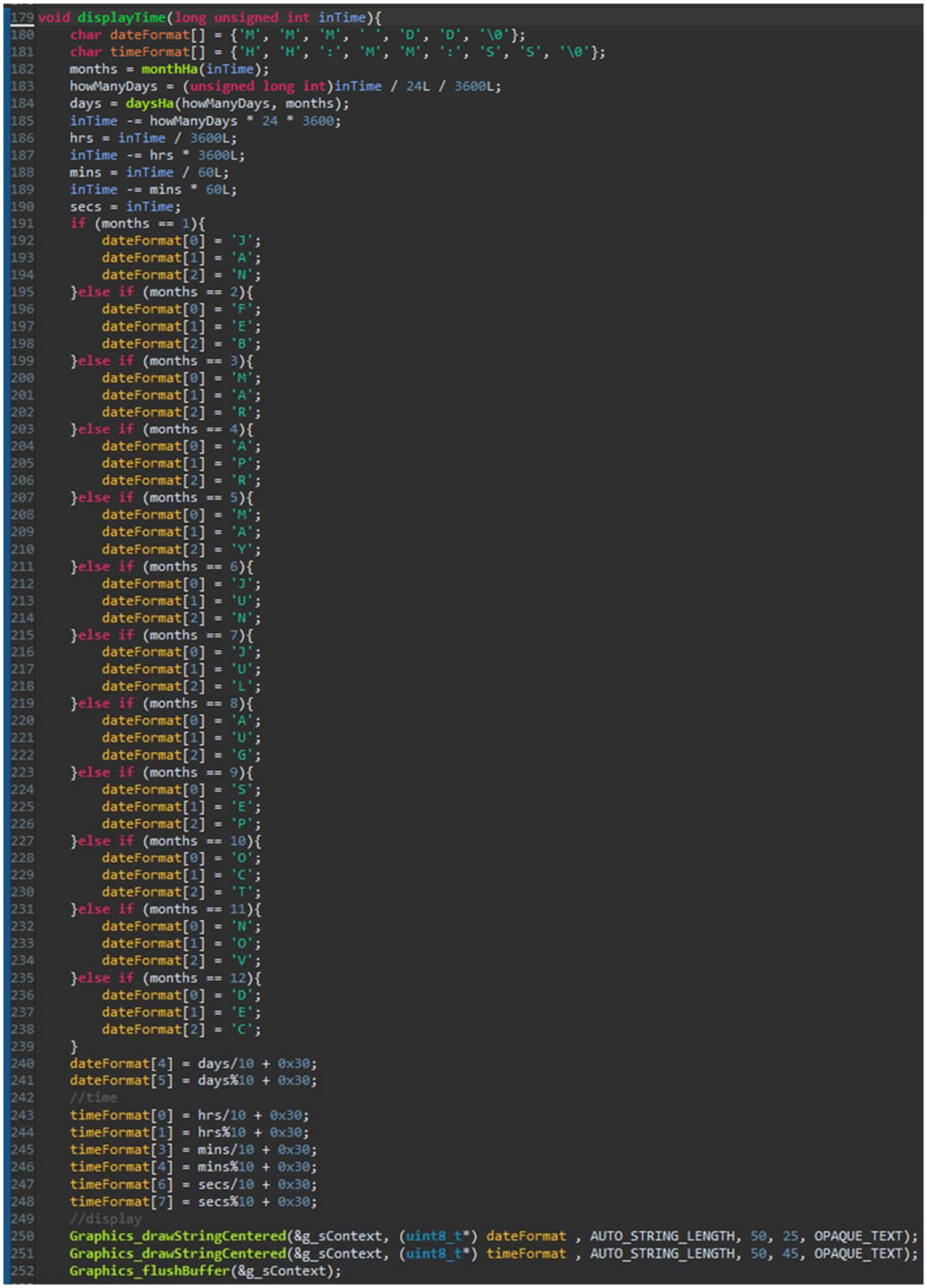
1. monthHa finds the month based on the inTime. It sets a integer (mths) to the number of the corresponding month that the daysGoneBy finds how many days into the year.

2. daysHa calculates the day in a month from the inTime and how many days passed in the year. This starts from midnight Jan 1

Figure 2 shows the displayTime function.

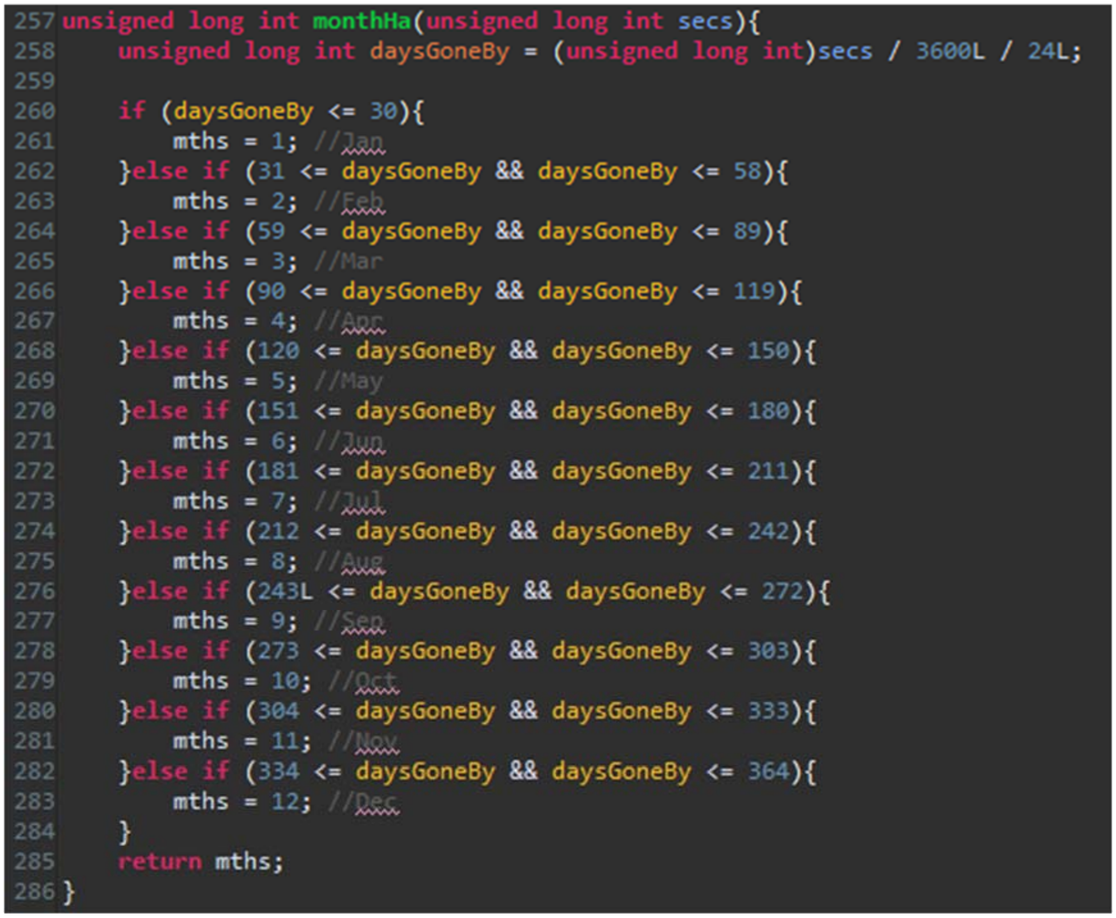
For the *displayTemp(float inAvgTempC)* it takes in a code float and turns that into the temperature in degrees Celsius and Fahrenheit.

Figure 5 shows the displayTemp function.



```c
void displayTime(long unsigned int inTime){
    char dateFormat[] = {'M', 'M', 'M', ' ', 'D', 'D', '\0'};
    char timeFormat[] = {'H', 'H', ':', 'M', 'M', ':', 'S', 'S', '\0'};
    months = monthHa(inTime);
    howManyDays = (unsigned long int)inTime / 24L / 3600L;
    days = daysHa(howManyDays, months);
    inTime -= howManyDays * 24 * 3600;
    hrs = inTime / 3600L;
    inTime -= hrs * 3600L;
    mins = inTime / 60L;
    inTime -= mins * 60L;
    secs = inTime;
    if (months == 1){
        dateFormat[0] = 'J';
        dateFormat[1] = 'A';
        dateFormat[2] = 'N';
    }else if (months == 2){
        dateFormat[0] = 'F';
        dateFormat[1] = 'E';
        dateFormat[2] = 'B';
    }else if (months == 3){
        dateFormat[0] = 'M';
        dateFormat[1] = 'A';
        dateFormat[2] = 'R';
    }else if (months == 4){
        dateFormat[0] = 'A';
        dateFormat[1] = 'P';
        dateFormat[2] = 'R';
    }else if (months == 5){
        dateFormat[0] = 'M';
        dateFormat[1] = 'A';
        dateFormat[2] = 'Y';
    }else if (months == 6){
        dateFormat[0] = 'J';
        dateFormat[1] = 'U';
        dateFormat[2] = 'N';
    }else if (months == 7){
        dateFormat[0] = 'J';
        dateFormat[1] = 'U';
        dateFormat[2] = 'L';
    }else if (months == 8){
        dateFormat[0] = 'A';
        dateFormat[1] = 'U';
        dateFormat[2] = 'G';
    }else if (months == 9){
        dateFormat[0] = 'S';
        dateFormat[1] = 'E';
        dateFormat[2] = 'P';
    }else if (months == 10){
        dateFormat[0] = 'O';
        dateFormat[1] = 'C';
        dateFormat[2] = 'T';
    }else if (months == 11){
        dateFormat[0] = 'N';
        dateFormat[1] = 'O';
        dateFormat[2] = 'V';
    }else if (months == 12){
        dateFormat[0] = 'D';
        dateFormat[1] = 'E';
        dateFormat[2] = 'C';
    }
    dateFormat[4] = days/10 + 0x30;
    dateFormat[5] = days%10 + 0x30;
    //time
    timeFormat[0] = hrs/10 + 0x30;
    timeFormat[1] = hrs%10 + 0x30;
    timeFormat[3] = mins/10 + 0x30;
    timeFormat[4] = mins%10 + 0x30;
    timeFormat[6] = secs/10 + 0x30;
    timeFormat[7] = secs%10 + 0x30;
    //display
    Graphics_drawStringCentered(&g_sContext, (uint8_t*) dateFormat , AUTO_STRING_LENGTH, 50, 25, OPAQUE_TEXT);
    Graphics_drawStringCentered(&g_sContext, (uint8_t*) timeFormat , AUTO_STRING_LENGTH, 50, 45, OPAQUE_TEXT);
    Graphics_flushBuffer(&g_sContext);
```

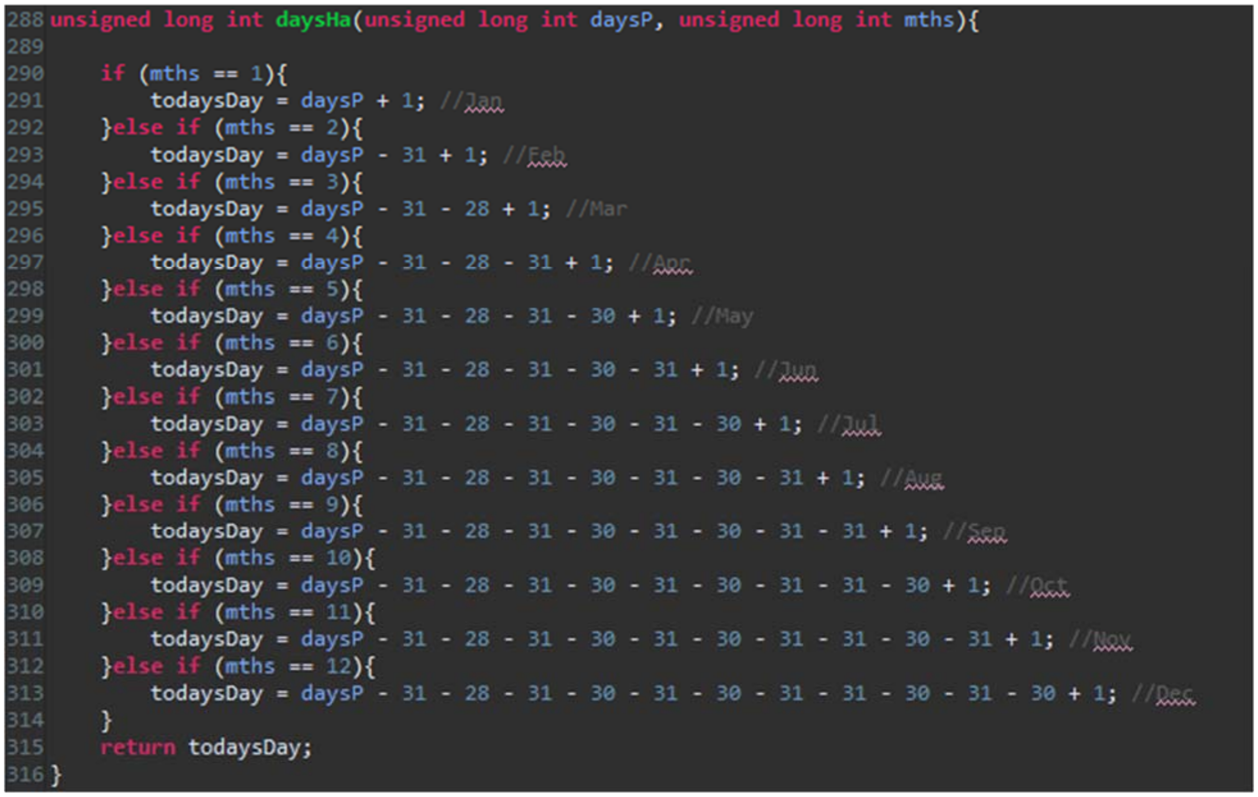
Figure 2: displayTime function



```
unsigned long int monthHa(unsigned long int secs){
    unsigned long int daysGoneBy = (unsigned long int)secs / 3600L / 24L;

    if (daysGoneBy <= 30){
        mths = 1; //Jan
    }else if (31 <= daysGoneBy && daysGoneBy <= 58){
        mths = 2; //Feb
    }else if (59 <= daysGoneBy && daysGoneBy <= 89){
        mths = 3; //Mar
    }else if (90 <= daysGoneBy && daysGoneBy <= 119){
        mths = 4; //Apr
    }else if (120 <= daysGoneBy && daysGoneBy <= 150){
        mths = 5; //May
    }else if (151 <= daysGoneBy && daysGoneBy <= 180){
        mths = 6; //Jun
    }else if (181 <= daysGoneBy && daysGoneBy <= 211){
        mths = 7; //Jul
    }else if (212 <= daysGoneBy && daysGoneBy <= 242){
        mths = 8; //Aug
    }else if (243L <= daysGoneBy && daysGoneBy <= 272){
        mths = 9; //Sep
    }else if (273 <= daysGoneBy && daysGoneBy <= 303){
        mths = 10; //Oct
    }else if (304 <= daysGoneBy && daysGoneBy <= 333){
        mths = 11; //Nov
    }else if (334 <= daysGoneBy && daysGoneBy <= 364){
        mths = 12; //Dec
    }
    return mths;
}
```

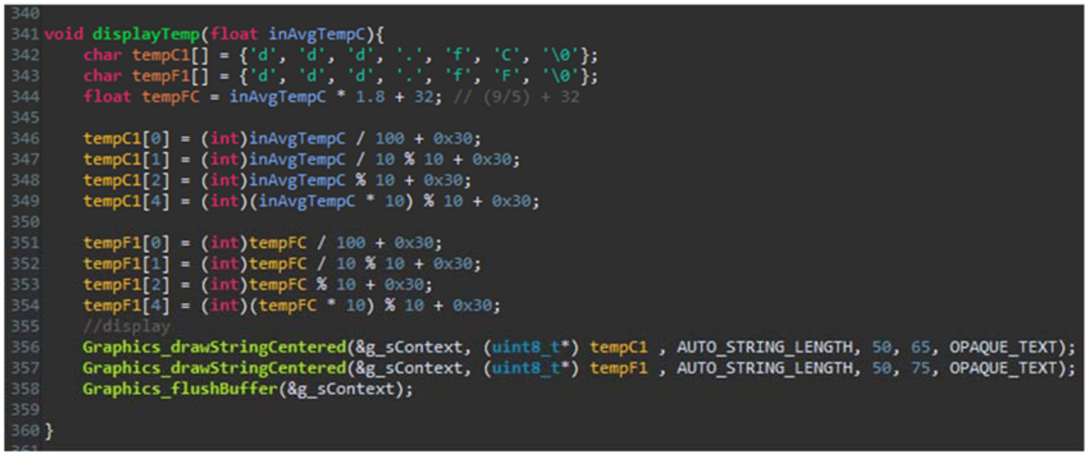
Figure 3: monthHa function



```c
unsigned long int daysHa(unsigned long int daysP, unsigned long int mths){

    if (mths == 1){
        todaysDay = daysP + 1; //Jan
    }else if (mths == 2){
        todaysDay = daysP - 31 + 1; //Feb
    }else if (mths == 3){
        todaysDay = daysP - 31 - 28 + 1; //Mar
    }else if (mths == 4){
        todaysDay = daysP - 31 - 28 - 31 + 1; //Apr
    }else if (mths == 5){
        todaysDay = daysP - 31 - 28 - 31 - 30 + 1; //May
    }else if (mths == 6){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 + 1; //Jun
    }else if (mths == 7){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 + 1; //Jul
    }else if (mths == 8){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 - 31 + 1; //Aug
    }else if (mths == 9){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 + 1; //Sep
    }else if (mths == 10){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30 + 1; //Oct
    }else if (mths == 11){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30 - 31 + 1; //Nov
    }else if (mths == 12){
        todaysDay = daysP - 31 - 28 - 31 - 30 - 31 - 30 - 31 - 31 - 30 - 31 - 30 + 1; //Dec
    }
    return todaysDay;
}
```

Figure 3: daysHa function



```c

void displayTemp(float inAvgTempC){
    char tempC1[] = {'d', 'd', 'd', '.', 'f', 'C', '\0'};
    char tempF1[] = {'d', 'd', 'd', '.', 'f', 'F', '\0'};
    float tempFC = inAvgTempC * 1.8 + 32; // (9/5) + 32

    tempC1[0] = (int)inAvgTempC / 100 + 0x30;
    tempC1[1] = (int)inAvgTempC / 10 % 10 + 0x30;
    tempC1[2] = (int)inAvgTempC % 10 + 0x30;
    tempC1[4] = (int)(inAvgTempC * 10) % 10 + 0x30;

    tempF1[0] = (int)tempFC / 100 + 0x30;
    tempF1[1] = (int)tempFC / 10 % 10 + 0x30;
    tempF1[2] = (int)tempFC % 10 + 0x30;
    tempF1[4] = (int)(tempFC * 10) % 10 + 0x30;
    //display
    Graphics_drawStringCentered(&g_sContext, (uint8_t*) tempC1 , AUTO_STRING_LENGTH, 50, 65, OPAQUE_TEXT);
    Graphics_drawStringCentered(&g_sContext, (uint8_t*) tempF1 , AUTO_STRING_LENGTH, 50, 75, OPAQUE_TEXT);
    Graphics_flushBuffer(&g_sContext);

}
```

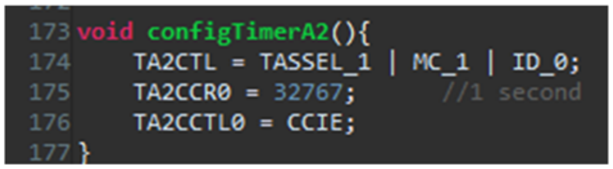
Figure 5: displayTemp function

Setup Timer A2:

Timmer A2 is configured to measure 1 second intervals. Using the ACLK course I set Max count to 32767 and then added 1 for it to become 32768 which happens to match the frequency of the ACLK. (32768).

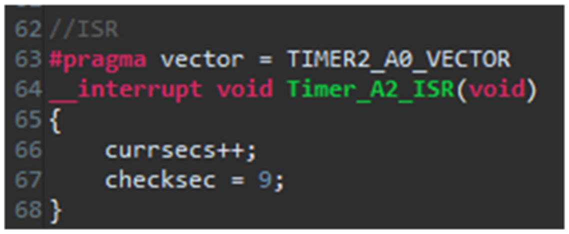
32768/32768 = 1

Figure 6 shows the setup and Figure 7 shows the #pragma statement of Timer A2



```
void configTimerA2(){
    TA2CTL = TASSEL_1 | MC_1 | ID_0;
    TA2CCR0 = 32767;        //1 second
    TA2CCTL0 = CCIE;
}
```

Figure 6: TimerA2 config

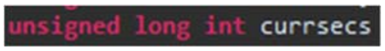


```
//ISR
#pragma vector = TIMER2_A0_VECTOR
__interrupt void Timer_A2_ISR(void)
{
    currsecs++;
    checksec = 9;
}
```

Figure 7: #pragma

What data type did you use to store your time count?

I used a unsigned long int to hold the time.



```
unsigned long int currsecs
```

Figure 8: data type for the time

ADC Setup:

Justify your choices in your report. What will the resolution of your readings be in volts and in C?

In the lab I decided to use one of the source voltages from the MSP as the reference. Since the temp sensor was internal it would work better on the 1.5 scale of the MSP. The reference I choose where at 30 and 85 degrees Celsius. They were best based on the resolution of the MSP's temp sensor.

The resolutions will be:

```
res = ((float)(85.0 - 30.0))/((float)(bits85-bits30));
degC / bit
```

1.5V / (2^12) bits = 0.000366 V/bit

You should use "circular indexing" (i.e. something like index = time count modulo 30) to index your array rather than shifting the arrays around. It's much more efficient. Explain why.

It's more efficient since it doesn't take up as much memory if you were to do it the more conventional way of shifting the array with the new one taking the last spot and shifting the past temps down and taking a new average.

Other:

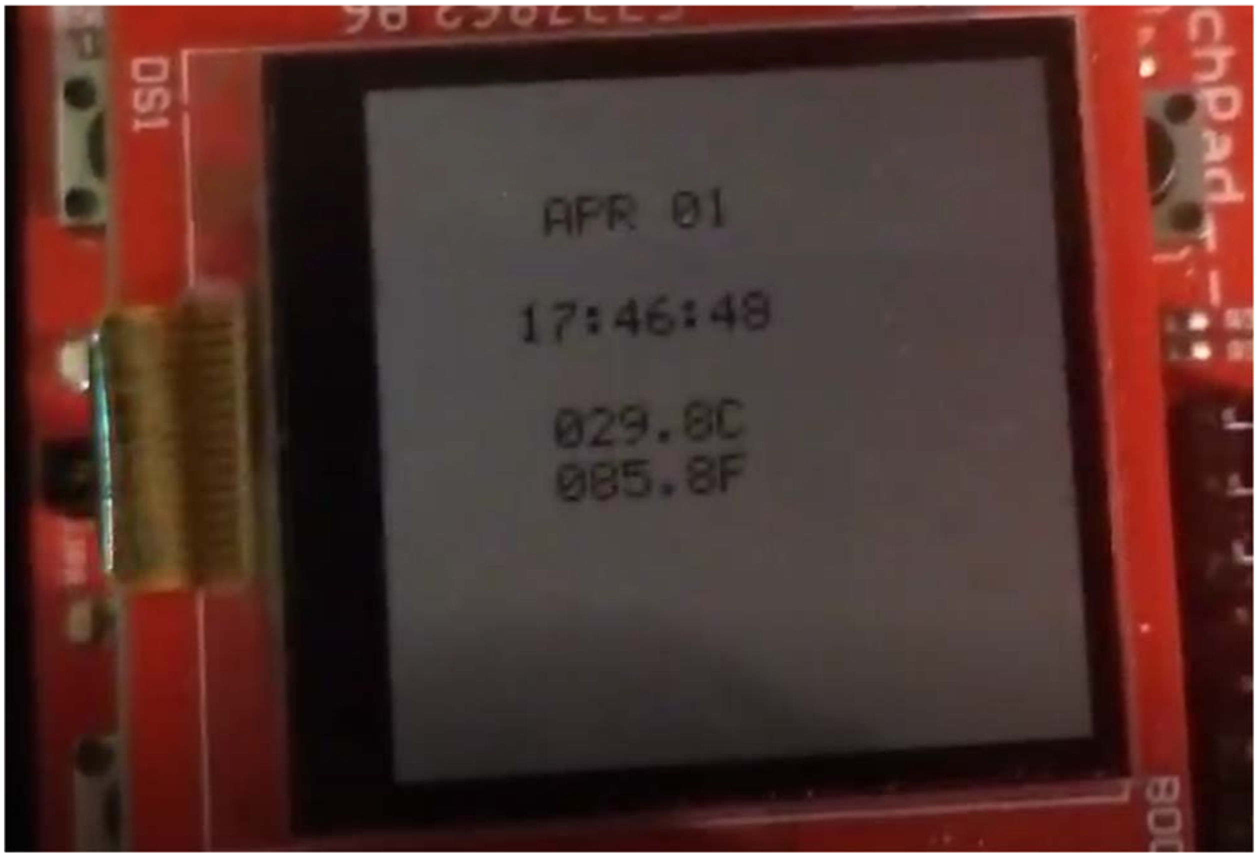


Figure 9: Display

The edit time is shown in the video on canvas. I went to a TA for a signoff and my webcam was cooperating. Hence the video.

I did not attempt either bonus.

Summary and Conclusion:

In conclusion, this lab had the purpose of getting students more familiar with the interrupts, timers, the C programming language and the environment of Code Composer Studio as well as get the hang of using the builder, debugger and the variable window within Eclipse/CCS. This concludes this laboratory experiment with the Date and Time Display on the MSP430F5529 using the buzzer, keypad, and Code Composer Studio.

Appendices

ASCII table:
https://canvas.wpi.edu/courses/23392/files/3432901?wrap=1

Lab 3 Handout:
https://canvas.wpi.edu/courses/23392/files/3416792?wrap=1

Demo Project:
https://canvas.wpi.edu/courses/23392/files/3416786?wrap=1

CCS download:
https://software-dl.ti.com/ccs/esd/documents/ccs_downloads.html

CCS tutorial:
https://www.youtube.com/watch?v=2W0ZHO0vzJE

MSP430 user guide:
https://canvas.wpi.edu/courses/23392/files/3432903?wrap=1

C Review pdf
https://canvas.wpi.edu/courses/23392/files/3439859?module_item_id=542026

ECE 2049 Lab BOM:
https://canvas.wpi.edu/courses/23392/files/3411701?module_item_id=527923

ECE 2049 Course Information:
https://www.wpi.edu/academics/calendar-courses/course-descriptions/17856/electrical-computer-engineering#ECE-2049

ECE2049 C-2020 Lab 3
Sign-off Sheet
***Report due*: 03/10/2021**

Student 1: ____________________________ **ECE mailbox:** _________

Student 2: ____________________________ **ECE mailbox:** _________

Board #: _______

YOU ARE RESPONSIBLE FOR *ALL* THE REQUIREMENTS LISTED IN THE REQUIREMENTS SECTION OF THIS ASSIGNMENT!

Timer A2 measuring seconds	10	
ADC12 making single channel, single measurements for ADC12_A temp sensor once per second	20	
Proper conversion and display of date & time (month and day – FEB 2 and hr min sec = HH.MM.SS)	15	
Proper conversion and display temperature in degrees C and F	10	
Edit mode using the push buttons	15	
Use ADC12_A interrupts rather than busy bit polling to get results	***5***	
BONUS: Indicating field being edited by underline, blink or color inversion or such	***5***	
Report (answering ***all*** questions from the requirements section)	30	
Total points	100	

***** Both Students MUST be present at Sign-off for any and all parts!!***