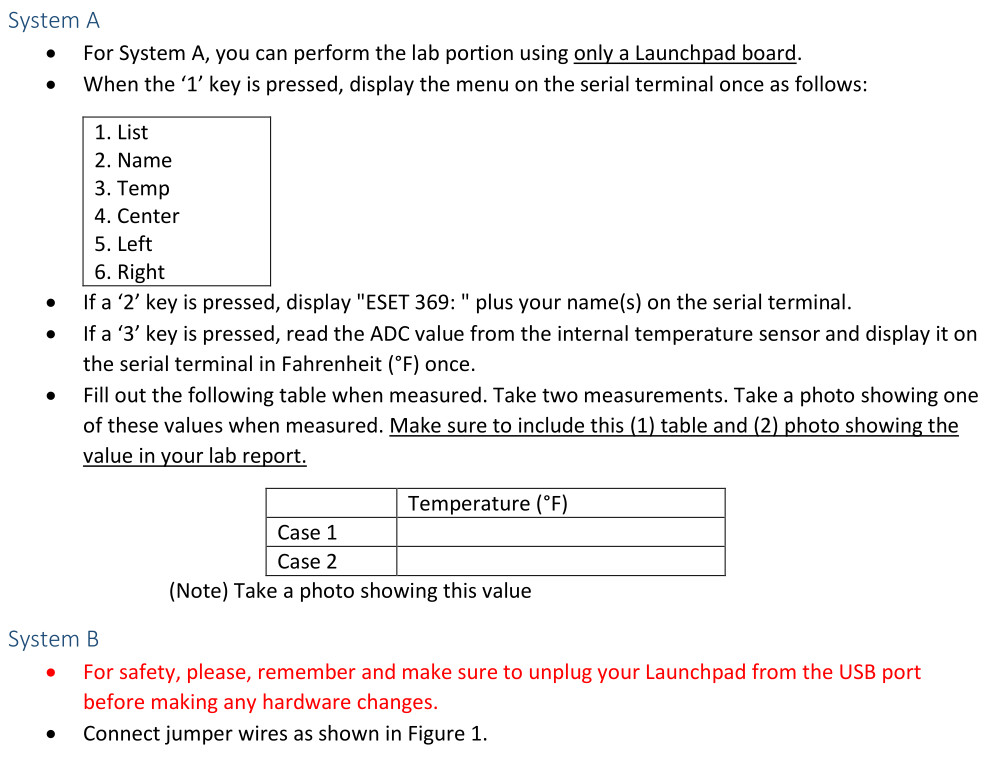
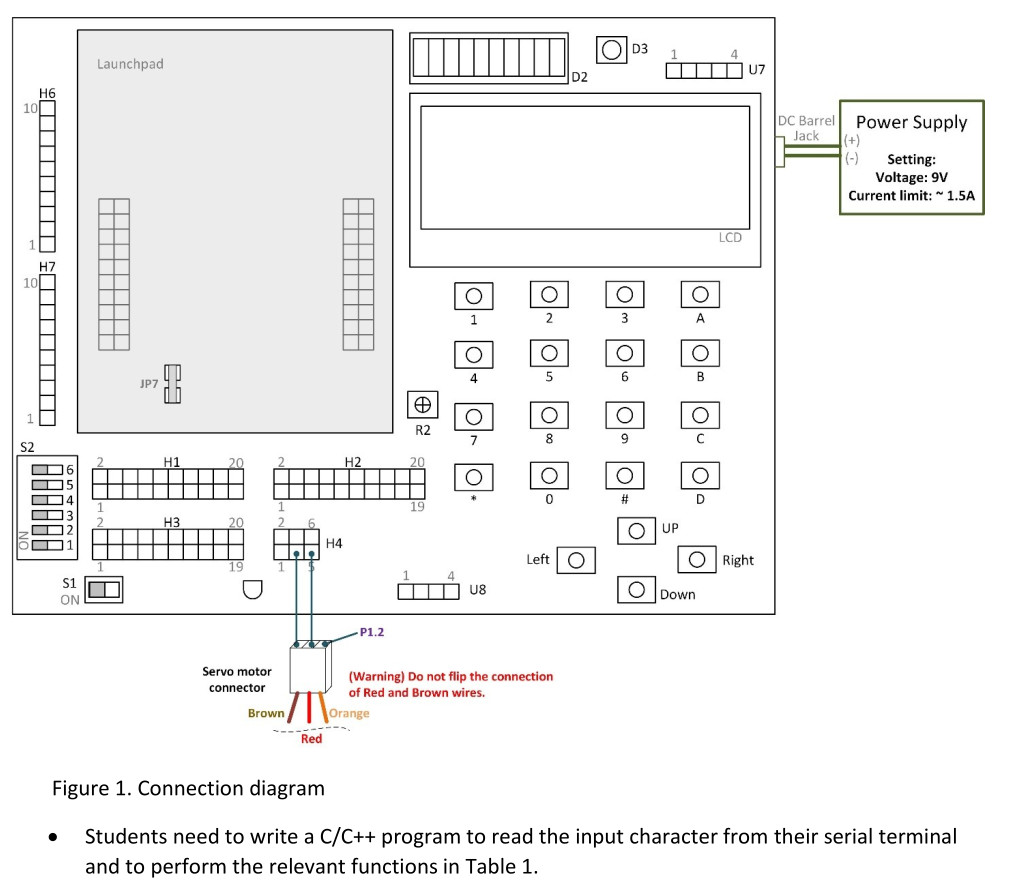
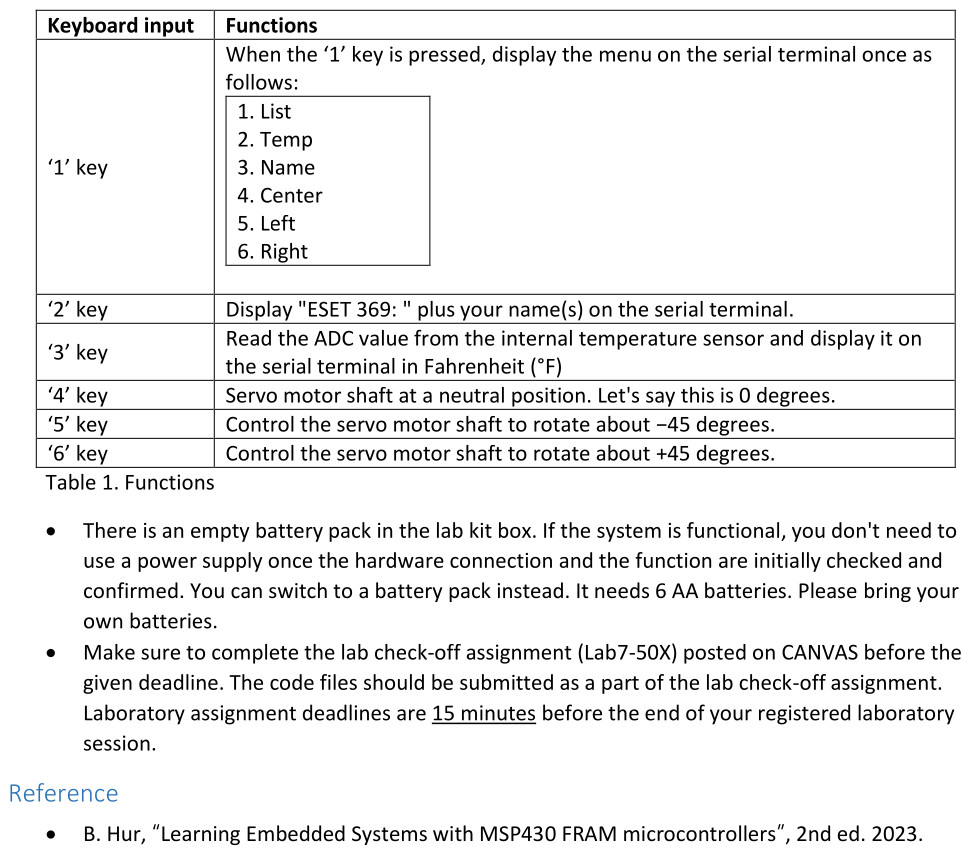
**Prompt from Canvas:**







**Simplified prompt:**

Students need to write a C/C++ program to read the input character from their serial terminal and to perform the relevant functions in Table 1.

**Table 1. Functions**

| **Keyboard Input** | **Functions** |
| --- | --- |
| **‘1’ key** | When the ‘1’ key is pressed, display the menu on the serial terminal once as follows:  1. List  2. Name  3. Temp  4. Center  5. Left  6. Right |
| **‘2’ key** | Display "ESET 369: " plus your name(s) on the serial terminal. |
| **‘3’ key** | Read the ADC value from the internal temperature sensor and display it on the serial terminal in Fahrenheit (°F). |
| **‘4’ key** | Servo motor shaft at a neutral position (0 degrees). |
| **‘5’ key** | Control the servo motor shaft to rotate about -45 degrees. |
| **‘6’ key** | Control the servo motor shaft to rotate about +45 degrees. |

Table 1. Functions

There is an empty battery pack in the lab kit box. If the system is functional, you don’t need to use a power supply once the hardware connection and the function are initially checked and confirmed. You can switch to a battery pack instead. It needs 6 AA batteries. Please bring your own batteries.

**Example of complete program with servo additions that don’t work (That might help):**

**#include** <msp430.h>

**#include** <stdio.h>

**unsigned** **char** **UART\_RX**(**void**);

**void** **UART\_TX**(**unsigned** **char** ch);

**void** **UART\_TX\_string**(**char** \*str);

**unsigned** **int** **read\_ADC\_temp**(**void**);

**float** **convert\_to\_Fahrenheit**(**unsigned** **int** adc\_value);

**void** **UART\_TX\_float**(**float** value);

**void** **set\_servo\_angle**(**int** angle);

**int** **main**(**void**) {

WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer

PM5CTL0 &= ~LOCKLPM5; // Unlock GPIO

// UART configuration

UCA0CTLW0 = UCSWRST;

UCA0CTLW0 |= UCSSEL\_2; // SMCLK

UCA0BRW = 6;

UCA0MCTLW = UCOS16 | (8 << 4) | (32 << 8);

P2SEL1 |= BIT0 | BIT1; // P2.0 = UCA0RXD, P2.1 = UCA0TXD

P2SEL0 &= ~(BIT0 | BIT1);

UCA0CTLW0 &= ~UCSWRST;

// ADC configuration

ADC12CTL0 = ADC12SHT0\_2 | ADC12ON;

ADC12CTL1 = ADC12SHP;

ADC12CTL2 |= ADC12RES\_2;

ADC12MCTL0 |= ADC12INCH\_30 | ADC12VRSEL\_1;

REFCTL0 |= REFON | REFVSEL\_0;

// Servo motor PWM configuration

P1DIR |= BIT2;

P1SEL0 |= BIT2;

P1SEL1 &= ~BIT2;

TA0CTL = TASSEL\_2 | MC\_1 | TACLR;

TA0CCR0 = 20000 - 1;

TA0CCTL1 = OUTMOD\_7;

set\_servo\_angle(0);

**while** (1) {

**unsigned** **char** ch = UART\_RX();

**if** (ch == '1') {

UART\_TX\_string("\r\n1. List\r\n2. Name\r\n3. Temp\r\n4. Center\r\n5. Left\r\n6. Right\r\n");

} **else** **if** (ch == '2') {

UART\_TX\_string("ESET 369: Kyle Rex\r\n");

} **else** **if** (ch == '3') {

**unsigned** **int** adc\_value = read\_ADC\_temp();

**float** temp\_F = convert\_to\_Fahrenheit(adc\_value);

UART\_TX\_string("Temp: ");

UART\_TX\_float(temp\_F);

UART\_TX\_string(" F\r\n");

} **else** **if** (ch == '4') {

set\_servo\_angle(0);

UART\_TX\_string("Servo set to 0 degrees (center)\r\n");

} **else** **if** (ch == '5') {

set\_servo\_angle(-45);

UART\_TX\_string("Servo set to -45 degrees (left)\r\n");

} **else** **if** (ch == '6') {

set\_servo\_angle(45);

UART\_TX\_string("Servo set to 45 degrees (right)\r\n");

}

}

**return** 0;

}

**unsigned** **char** **UART\_RX**(**void**) {

**while** (!(UCA0IFG & UCRXIFG));

**return** UCA0RXBUF;

}

**void** **UART\_TX**(**unsigned** **char** ch) {

**while** (!(UCA0IFG & UCTXIFG));

UCA0TXBUF = ch;

}

**void** **UART\_TX\_string**(**char** \*str) {

**while** (\*str) {

UART\_TX(\*str++);

}

}

**unsigned** **int** **read\_ADC\_temp**(**void**) {

ADC12CTL0 |= ADC12ENC | ADC12SC;

**while** (ADC12CTL1 & ADC12BUSY);

**return** ADC12MEM0;

}

**float** **convert\_to\_Fahrenheit**(**unsigned** **int** adc\_value) {

**float** temp\_C = ((adc\_value - 630.0f) / 10.0f) + 30.0f;

**return** (temp\_C \* 9.0f / 5.0f) + 32.0f;

}

**void** **UART\_TX\_float**(**float** value) {

**int** int\_part = (**int**)value;

**int** frac\_part = (**int**)((value - int\_part) \* 100);

**char** buffer[10];

**int** i = 0;

**if** (int\_part == 0) {

UART\_TX('0');

} **else** {

**if** (int\_part < 0) {

UART\_TX('-');

int\_part = -int\_part;

}

**while** (int\_part) {

buffer[i++] = (int\_part % 10) + '0';

int\_part /= 10;

}

**while** (i) {

UART\_TX(buffer[--i]);

}

}

UART\_TX('.');

UART\_TX((frac\_part / 10) + '0');

UART\_TX((frac\_part % 10) + '0');

}

**void** **set\_servo\_angle**(**int** angle) {

**int** pulse\_width;

pulse\_width = 1500 + (angle \* 5.56);

**if** (pulse\_width < 1000) pulse\_width = 1000;

**if** (pulse\_width > 2000) pulse\_width = 2000;

TA0CCR1 = pulse\_width;

}

**Example of a servo motor program from a previous lab that works (That might help):**

**#include** <msp430.h>

**int** **main**(**void**) {

WDTCTL = WDTPW | WDTHOLD; // hold the watchdog timer

PM5CTL0 &= ~LOCKLPM5; // clear LOCKLPM5 bit

P1DIR |= BIT0; // output direction (P1.0)

P1DIR |= BIT2; // output direction (P1.2)

TA1CCR0 = 20000; // TA1CCR0 value

TA1CCR1 = 1500; // TA1CCR1 value

TA1CTL = TASSEL\_2 | MC\_1 | TACLR; // TA1CTL setup

TA1CCTL0 = CCIE; // enable CCIE

TA1CCTL1 = CCIE; // enable CCIE

\_enable\_interrupt(); // enable general interrupt

**while**(1) {

P1OUT ^= BIT0; // toggle (P1.0)

**\_\_delay\_cycles**(250000); // delay

}

**return** 0;

}

**#pragma** vector = TIMER1\_A0\_VECTOR

\_\_interrupt **void** **Timer1\_A0\_ISR**(**void**) {

P1OUT |= BIT2; // set (P1.2)

}

**#pragma** vector = TIMER1\_A1\_VECTOR

\_\_interrupt **void** **Timer1\_A1\_ISR**(**void**) {

**if**((TA1CCTL1 & CCIFG) != 0) { // check CCIFG flag

P1OUT &= ~BIT2; // clear (P1.2)

TA1CCTL1 &= ~CCIFG; // clear CCIFG flag

}

}

**Connections:**

Orange (Control) Servo Wire connected to P1.2 of MSP430FR5994 Launchpad

Red (VCC) Servo Wire connected to 9V Power Source

Brown (GND) Servo wire connected to GND