CPE 325: Intro to Embedded Computer System

Lab08

UART

Submitted by: Caleb Keller

Date of Experiment:_____10/17/2022_____

Report Deadline:_____10/20/2022_____

Demonstration Deadline: _____10/25/2022_____

Introduction

Write a brief discussion on what the lab is about. (Use the tutorial and write in your own words. DO NOT copy text).

In this lab, I am writing C code to implement a UART program called "CalcBot" that will perform mathematical operations on inputted integers. I will also be implementing code to display a triangular wave using the UAH serial app.

Theory

Write short notes on each topic discussed in lab.

Topic 1: Serial Communication and UART

a.) Serial Communication: this is how a MSP430 system can communicate with another system. When the two devices have a common clock source, communication can begin. There are two modes in which the devices can communicate. Synchronous and asynchronous mode.

Topic 2: UAH Serial App

a.) The UAH Serial app is used to view the RAMP signal implemented in a program. It is also used to send and view other types of data including ints, floats, etc.. It translates serial packets that it receives, and then it can graphically represent the data versus time.

Results & Observation

Program 1:

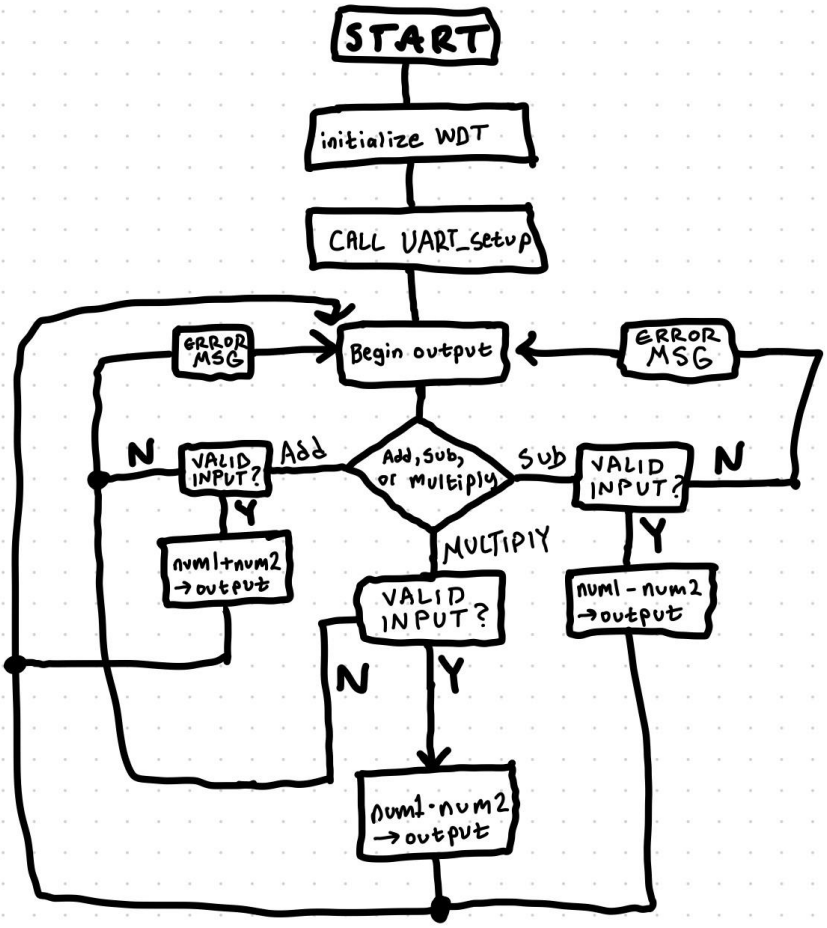
Program Description:

Explain your approach in solving the problem.

To perform the given tasks for this program, I began by writing the function definitions for sending and receiving characters using UART. Once I had the functions defined, I began writing the main code that will implement the CalcBot program. My program unfortunately did not display, therefore I do not have any screenshots to represent the output.

Program 1 Flowchart

LAB 8 PART 1


START
initialize WDT
CALL UART_setup
Begin output
ERROR MSG
ERROR MSG
Add, Sub, or multiply
Add
Sub
MULTIPLY
VALID INPUT?
N
Y
num1+num2 → output
VALID INPUT?
N
Y
num1 - num2 → output
VALID INPUT?
N
Y
num1 · num2 → output

Program 2:

Program Description:

Explain your approach in solving the problem.

N/A

Program 2 Flowchart

N/A

Conclusion

While working on this program, I ran into many challenges and issues. At the beginning of my coding process, I was able to display data, but somewhere during the implementation of the rest of my code, that was no longer the case. Although I wish my program would have worked, I am thankful for the things I have learned within this lab.

Appendix

Your first code goes here, if any. Make sure you use a 1X1 table for this.

(Note: Make sure the code is readable, have comments. Also reduce spacing between lines to avoid lengthy reports.

Program 1 source code

```
/*
 * Lab 08
 * Author: Caleb Keller
 * Description: this UART program implements the "CalcBot" program to add,
subtract, or multiply inputed data
 */



#include <msp430.h>
#include <stdio.h> // I/O
#include <string.h> // to use string functions

static int limit = 50; // somme max number input
char a[] = "add";
char s[] = "subtract";
char m[] = "multiply";
char greet[] = "Hi, I am CalcBot! Do you need help?";
char choice[] = "Would you like to 'add', 'subtract', or 'multiply'?";
char first[] = "What is your first operand?";
char second[] = "What is your second operand";
char error[] = " is not a valid operation. Try again!";
char answer1[] = "The answer is ";
char answer2[] = ". I hope you get a good grade!";
```

```
int inputsize = 18; // size of required input
char input[] = "";
int num1, num2, result; // for input

void UART_setup()
{
    // configures UCI to work in the UART mode
    // at the baud rate of 115,200
    P2SEL |= BIT4 + BIT5; // set UC0TXD and UC0RXD to transmit and receive data
    UCA0CTL1 |= BIT0; // software reset
    UCA0CTL0 = 0; // USCI_A0 control register (default no changes)
    UCA0CTL1 |= UCSSEL_2; // SMCLK clock source (0 = UCLK, 1 = ACLK, 2 and 3 =
SMCLK)
    UCA0BR0 = 9; // baud rate: 1048576 Hz/ clock source (2^20 for SMCLK) = , 115200
lower byte
    UCA0BR1 = 0; // upper byte
    UCA0MCTL = 0x02; // modulation
    UCA0CTL1 &= ~BIT0; // UCSWRST reset

}
void UART_sendCharacter(char myChar)
{
    // sends character via UART

        while(!(IFG2 & UCA0TXIFG)){
        UCA0TXBUF = myChar; // transmit char *TX*
        }

    while(!(IFG2 & UCA0TXIFG))
    {
        UCA0TXBUF = 0x0D; // carriage return
    }

}
char UART_getCharacter()
{
    // waits for the character from UART and returns it
    while(!(IFG2&UCA0RXIFG));
    UCA0TXBUF = UCA0RXBUF; // TXBUF <= RXBUF *receive char via RX to TX*
}
void UART_sendString(char* str)
{
   // sends a string via UART using sendCharacter(char myChar)
    //sprintf("Hi, I am Calcbot! Do you need help? \n");

    int i;

    while(!(IFG2 & UCA0TXIFG))
    {
        for(i = 0; i < strlen(str); i++)
        {
            UART_sendCharacter(str[i]);
        }
    }
}
```

```
void UART_getLine(char* buf, int limit)
{
    // receives characters via UART using UART_getCharacter() until it finds
    // the new line character '\n' or until the limit of characters is exceeded
    // Writes that string (excluding the new line character) to the buffer
    // allocated outside of the function. Terminates the string with the null
character

    int i;

    for(i = 0; i < limit; i++)
    {
    while(!(IFG2 & UCA0RXIFG)) // wait for new char
    {
        // new char i here in UCA0RXBUF
        while(!(IFG2 & UCA0TXIFG)) // wait until TXBUF is free
        {
            UCA0TXBUF = UCA0RXBUF; // TXBUF <= RXBUF (echo)

            if(UCA0TXBUF == '\n') // if new line character end of input
            {
                buf[i] = NULL;
                UART_sendString('\n');
                break;
            }
        }
    }
    }
    return;

}

int main(void)

{


       WDTCTL = WDT_ADLY_1000;
       //_EINT(); // enable interrupts
       UART_setup(); // call setup function

       for(;;) // infinite loop
       {
           UART_sendString(greet); // start printing out opening message
           UART_getLine(input, inputsize);

               while(!(inputsize)) // while not the size of "Help me, CalcBot! ->
18"
               {
                   UART_sendString(greet);
               }
           UART_sendString(choice);
               if(UCA0RXBUF != 'add' || 'subtract' || 'multiply')
               {
                   char temp[] = "";
```

```
                    UCA0TXBUF = UCA0RXBUF; // echo
                    UART_sendString(error);
                }
                if(UCA0RXBUF == 'add')
                {
                    UART_sendString(first);
                    num1 = UCA0RXBUF;
                    UART_sendString(second);
                    num2 = UCA0RXBUF;

                    result = num1 + num2;
                    UART_sendString(answer1);
                    UART_sendString(result);
                }
                if(UCA0RXBUF == 'subtract')
                {
                    UART_sendString(first);
                    num1 = UCA0RXBUF;
                    UART_sendtring(second);
                    num2 = UCA0RXBUF;

                    result = num1 - num2;
                    UART_sendString(answer1);
                    UART_sendString(num2);
                    UART_sendString(answer2);

                }
                if(UCA0RXBUF == 'multiply')
                {
                    UART_sendString(first);
                    num1 = UCA0RXBUF;
                    UART_sendStrinng(second);
                    num2 = UCA0RXBUF;

                    result = num1 * num2;
                    UART_sendString(answer1);
                    UART_sendString(result);
                    UART_sendString(answer2);
                }
        }

}

#pragma vector = USCIAB0RX_VECTOR
__interrupt void USCIA0RX_ISR (void){

    UCA0TXBUF = UCA0RXBUF;
}

/*
#pragma vector = WDT_VECTOR
__interrupt void watchdog_timer(void)
{
    char *myPointer = (char* )&myData;
    char index = 0;
```

```
    UART_sendCharacter(0x55);
    for(index = 0; index < 4; index++)
    {
        UART_sendCharacter(myPointer[index]);
        myData = (myData + 0.1);
        if(myData >= 17.0)
        {
            myData = 0.0;
        }
    }
}
*/
```

Program 2 Source Code

Make sure to submit a single pdf file only