|  |
| --- |
| EEL-4742-12 |
| EEL 4742 Laboratory |
| Laboratory 2 Report |
|  |
| **Andrew Mendez, Daniel Franco** |
|  |

**Objective:**

In this laboratory experiment, we learned and developed programs that used serial communication using MSP430 Universal Asynchronous Receiver and Transmitter (UART). We developed several C and Assembly language programs utilizing MSP430 UART for serial communication.

**Apparatus List:**

* Dell Computer, with monitor, keyboard, and mouse.
* CCS software.
* MSP430FG4618
* HyperTerminal

**Procedure:**

The procedure is composed of two sections. The first section is implementing MSP430 UART Communication programs using C language. The second section is implementing MSP430 UART Communication programs using MSP430 Assembly language.

Section 1:

Using the C language, our procedure consists of implementing the following four programs that used the MSP430 UART communication to communicate to the HyperTerminal program:

1. A program to enter character using a computer keyboard and the character displays on the HyperTerminal. Once the character is displays on the HyperTerminal, the UART communication is stopped and a yellow LED blinks indicating the communication is terminated.
2. A program that enables the user to print one character at a time, forming a string of characters on the HyperTerminal characters until null is printed-0x00. Once the null character is entered, the UART communication stops and a yellow LED blinks indicating the communication is terminated.
3. A program to display the string ,“Laboratory #2 for EEL 4742 Embedded Systems” onto the HyperTerminal using UART Serial Communication. Once the string is printed, the UART communication stops and a yellow LED blinks indicating the communication is terminated.
4. A program that enables the user to select either the Green LED button ASCII characters “G”, or the yellow LED button using ASCII characters “Y”. The user turns on the led by selecting the character “G” or “Y”. The selected LED will stay on until user selects the same led again. Both LED’s can be turned on.
5. A program that printed the status of the SW1 and SW2 switches onto the HyperTerminal. The program displays a 1 to the SWX if a button is pressed, and if the button is not pressed the program will print a zero as the SWX value. Three strings are printed to handle three conditions: SW1 is pressed (SW1=1) and SW2 is not pressed(SW2=0), SW1 is not pressed(SW1=0) and SW2 is pressed (SW2=1), and finally both SW1 and SW2 are pressed (SW1=1 SW2=1). If SW1 and SW2 are not pressed, then nothing is printed on the HyperTerminal.

Section 2:

Using the the MSP430 assembly language, our procedure consists of implementing the previous four programs that used the MSP430 UART communication to communicate to the HyperTerminal.

**Design Methodology:**

Section 1:

1. For the first program, the code was provided in the lab report. We used the Code Composer Studio v5.3 to run the provided code. We made a blank css project that generated a main.c and in this file we pasted the given code into this file. We utilized this code to learn the MSP430 UART communication and used it as a template for our other programs.
2. For the second program, we used the first program as a template to communicate from the MSP430 to the HyperTerminal. In order to develop a program that enables the user to print on character at a time, we needed a continuous way to print characters typed on the keyboard. We decided to use a while loop. The loop would make the program continuously wait for the user to enter a character and print the character onto the HyperTerminal. We added an if statement to check if the character was 0x00. If the user printed the null character, we used a command to break the while loop. Then after the while loop broke, the yellow LED would continuously blink indicating the communication is terminated.
3. For our third program, we extended the first program from printing a single character onto the hyper terminal to printing an entire string. The UART\_OUT function is the function used to display a character. In order to display a string on the hyper terminal using this function, we decided to use a while loop that iterated through all of the characters of the string. For each character in the string, the UART\_OUT function would be called to display the characters of the string. Once the entire string has been iterated, the yellow LED would continuously blink indicating the communication is terminated.
4. For our forth program, we needed to use our previous knowledge of how to turn on and off LED’s from the first laboratory experiment to design this program that used the ASCII characters “G” and “Y” and pseudo SW1 and SW2 respectively. We needed to set the directions of the P2DIR register to set the green and yellow led to be outputs. Then similar to how the we handled the buttons in the first experiment, we handle the conditions of whether the “G” character was entered, “Y” character was entered, and both were entered to set the P2OUT register to turn on the green led, the yellow led, or both LED’s respectively. We realized we can toggle the switching of LED’s on and off using if statements to determine which ASCII character was selected and XOR’ing the values of the P2OUT to toggle the green and yellow LED’s on and off.
5. For the 5th program, we needed to use our previous knowledge of how to handle input from the SW1 and SW2 buttons from the first laboratory experiment to design this program that prints the state of the buttons onto the hyper terminal. In order to accomplish this, we need to set the P1DIR register to select the SW1 and SW2 ports to be inputs. Also we need to initialize three strings as the output to the HyperTerminal to display the three possible states of the buttons. We then devised a control scheme to handle what string to display on the HyperTerminal based on the three possible conditions that can occur.

Section 2:

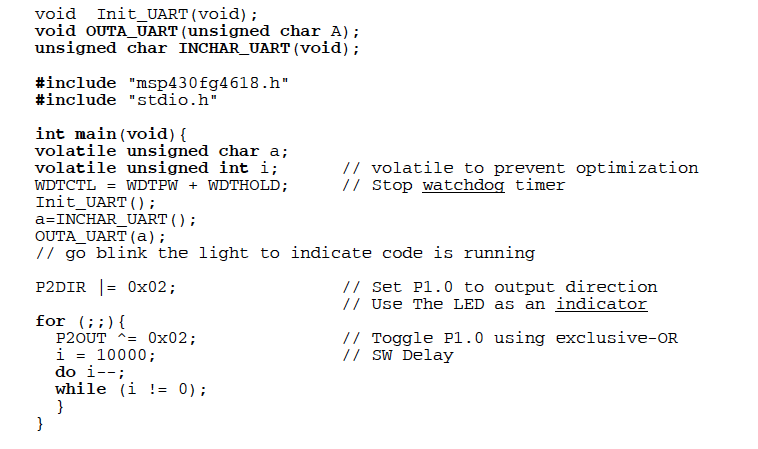
We implement the same design methodology using the MSP430 assembly language. The only difference is the syntax that is used to handle the specifications of the 5 programs.

**Test Plan:**

For each program in each of the two sections, we checked if the programs meet the procedure specifications using our design methodology.

**Source Code:**

1. Print once character in HyperTerminal, then print yellow LED



2. Print one character at a time, forming a string of characters on the HyperTerminal characters until null is printed-0x00.

void Init\_UART(void);

void OUTA\_UART(unsigned char A);

unsigned char INCHAR\_UART(void);

#include "msp430fg4618.h"

#include "stdio.h"

int main(void){

volatile unsigned char a;

volatile unsigned int i=1; // volatile to prevent optimization

WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

//initalized UART communication with MSP430 to hyperterminal

Init\_UART();

while(i!=0){

a=INCHAR\_UART();//function that reads character from keyboard

if(a=='0x00'){//if character enter is null, break loop

break;

}

OUTA\_UART(a);//display character in hyperterminal

}

// go blink the light to indicate code is running

P2DIR |= 0x02; // Set P1.0 to output direction

// Use The LED as an indicator

for (;;){

P2OUT ^= 0x02; // Toggle P1.0 using exclusive-OR

i = 10000; // SW Delay

do i--;

while (i != 0);

}

}

3. A program to display the string ,“Laboratory #2 for EEL 4742 Embedded Systems” ont0 the HyperTerminal using UART Serial Communication.

void Init\_UART(void);

void OUTA\_UART(unsigned char A);

unsigned char INCHAR\_UART(void);

#include "msp430fg4618.h"

#include "stdio.h"

int main(void){

volatile unsigned char a[100]="Laboratory #2 for EEL4742 Embedded Systems0";//string to print

volatile unsigned char b;

volatile unsigned int i=0; // index of string

WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

Init\_UART();

while(i>=0){//loop until entire string is printed

b=a[i];//temp variable to hold character

if(a[i]=='0'){//if null character is found, break loop

break;//break statement

}

OUTA\_UART(b);//display character in hyperterminal

i++;//increment index

}

// go blink the light to indicate code is running

P2DIR |= 0x02; // Set P1.0 to output direction

// Use The LED as an indicator

for (;;){

P2OUT ^= 0x02; // Toggle P1.0 using exclusive-OR

i = 10000; // SW Delay

do i--;

while (i != 0);

}

}

4. A program that enables the user to select either the Green LED button ASCII characters “G”, or the yellow LED button using ASCII characters “Y”. The user turns on the led by selecting the character “G” or “Y”.

void Init\_UART(void);

void OUTA\_UART(unsigned char A);

unsigned char INCHAR\_UART(void);

#include "msp430fg4618.h"

#include "stdio.h"

int main(void){

volatile unsigned char a; // variable to store input

volatile unsigned int i=0; // volatile to prevent optimization

//int on=0;

//int on2=0;

//P2OUT=0x04;

P2DIR |= 0x06; // Set P1.0 to output direction

WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

Init\_UART();//initalize MSP430 UART serial comm

P2OUT=0x00;

for(;;){//loop continuously

a= INCHAR\_UART();//read input character from keyboard

if(a=='G')//if character is G, then select the green LED

P2OUT = P2OUT ^ 0X04;//toggle value of light: if off, then on and vice versa

if(a=='Y')//if character is Y, then select the yellow LED

P2OUT = P2OUT ^ 0X02;//toggle value of light: if off, then on and vice versa

}

}

5. A program that printed the status of the SW1 and SW2 switches onto the HyperTerminal. The program displays a 1 to the SWX if a button is pressed, and if the button is not pressed the program will print a zero as the SWX value.

int main(void){

volatile unsigned char a[40]=" SW1=1,SW2=0 P"; // string that displays first case

volatile unsigned char b[40]=" SW1=0, SW2=1 P"; // string that displays second case

volatile unsigned char c[40]=" SW1=1, SW2=1 P ";// string that displays third case

volatile unsigned int i=0; // volatile to prevent optimization

P2DIR |= 0x06; // Set P2.0 to output direction, which are the LEDs

P1DIR |= 0x00; // Set P1.0 to output direction, which are the buttons

WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

Init\_UART();//initalize MSP430 UART comm

//int i;

int state=0;//initalize variable to be checking different button states

P2OUT=0x00;//initalize LED's to be off

for(;;){

//a= INCHAR\_UART();

if(P1IN==0x01 && state !=1){//check condition when button two is down

i=0;

while(b[i]!='P'){//print string of first case, end when found "P" to terminate loop

OUTA\_UART(b[i]);

i++;//increment string index

}

OUTA\_UART(0x0A);//send newline character to hyperterminal

OUTA\_UART(0x0D);//send carriage return character to hyperterminal

state=1;

}

else if(P1IN==0x02 && state !=2){// check condition when button one is down

i=0;

while(a[i]!='P'){//print string of second case, end when found "P" to terminate loop

OUTA\_UART(a[i]);

i++;//increment index

}

OUTA\_UART(0x0A);//send newline character to hyperterminal

OUTA\_UART(0x0D);//send carriage return character to hyperterminal

state=2;

}

else if(P1IN==0x00 && state!=3){//check condition when both buttons are down

i=0;

while(c[i]!='P'){//print string of third case, end when found "P" to terminate loop

OUTA\_UART(c[i]);

i++;//increment index

}

OUTA\_UART(0x0A);//send newline character to hyperterminal

OUTA\_UART(0x0D);//send carriage return character to hyperterminal

state=3;

}

else//none of the buttons are being pressed

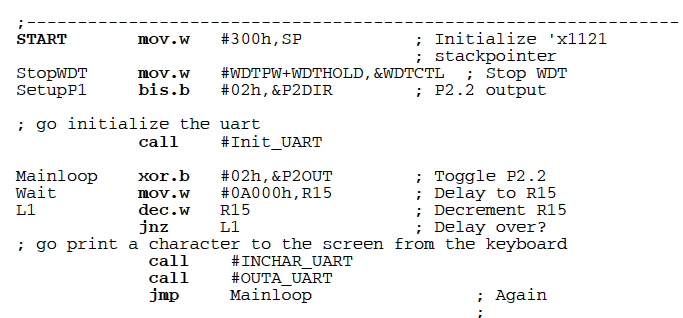
state=5;

}

}

Section 2:

1. Print once character in HyperTerminal,



1. Print one character at a time, forming a string of characters on the HyperTerminal characters until null is printed-0x00.

;----------------------------------------------------------------

START mov.w #300h,SP ; Initialize 'x1121 stackpointer

StopWDT mov.w #WDTPW+WDTHOLD,&WDTCTL ; Stop WDT

SetupP1 bis.b #02h,&P2DIR ;P2.2 output

call #Init\_UART ;initialize the uart

Mainloop

xor.b #02h,&P2OUT ;Toggle P2.2, flashes LED

Wait

mov.w #0A000h,R15 ;Delay to R15

L1

dec.w R15 ; Decrement R15

jnz L1

call #INCHAR\_UART

;check the value of R4 which contains the ASCII value

cmp.b #0, R4 ;check null, if so return

jmp Mainloop ;loop back and run again

1. A program to display the string ,“Laboratory #2 for EEL 4742 Embedded Systems” ont0 the HyperTerminal using UART Serial Communication.

strg1 .string "Laboratory #2 for EEL4742 embedded Systems" ;Define string one to be printed

.byte 0x0d,0x0a ;CR and LF

.byte 0x00 ;null terminate the string with

.text ;program start

.global \_START ;define entry point

;----------------------------------------------------------------

START mov.w #300h,SP ; Initialize 'x1121 stackpointer

StopWDT mov.w #WDTPW+WDTHOLD,&WDTCTL ; Stop WDT

SetupP1 bis.b #02h,&P2DIR ;P2.2 output

call #Init\_UART ;initialize the uart

Mainloop

xor.b #02h,&P2OUT ;Toggle P2.2, flashes LED

Wait

mov.w #0A000h,R15 ;Delay to R15

L1

dec.w R15 ; Decrement R15

jnz L1

mov #strg1,R6 ;set up string pointer, string1 to register 6

call #send ;send string to be outputed

getStuck jmp getStuck ; infinite loop

;loop to send in characters to OUTA

send

mov.b @R6, R4 ;move 1 char of string1 to R4 for OUTA

cmp.b #0, R4 ;check null, if so return

jeq L3

call #OUTA\_UART ;output character

inc R6 ;increment string pointer to next char

mov.b @R6, R7 ;move next character into R7

cmp.b #0x00, R7 ;check for NULL character unless repeat

jnz send

L3 ret

1. A program that enables the user to select either the Green LED button ASCII characters “G”, or the yellow LED button using ASCII characters “Y”. The user turns on the led by selecting the character “G” or “Y”.

;----------------------------------------------------------------

START mov.w #300h,SP ; Initialize 'x1121 stackpointer

StopWDT mov.w #WDTPW+WDTHOLD,&WDTCTL ; Stop WDT

SetupP1 bis.b #02h,&P2DIR ;P2.2 output

call #Init\_UART ;initialize the uart

mov.b #0x00,&P2OUT ;turn off the LEDs

Mainloop

call #INCHAR\_UART ;get input from keyboard

call #OUTA\_UART ;output the input to screen

cmp.b #0x47, R4 ;check if input was "G" if so go to Green

jeq Green

cmp.b #0x59, R4 ;check if input was "G" if so go to Green

jeq Yellow

jmp Mainloop ;if not "G" or "Y" restart loop

Green

xor.b #0x04,&P2OUT ;toggle the green led on or off

jmp Mainloop

Yellow

xor.b #0x02,&P2OUT ;toggle the yellow led on or off

jmp Mainloop

1. A program that printed the status of the SW1 and SW2 switches onto the HyperTerminal. The program displays a 1 to the SWX if a button is pressed, and if the button is not pressed the program will print a zero as the SWX value.

;strings for SW1,SW2, none, or both

strg1 .string "SW1 = 1,SW2 = 0";

.byte 0x00

strg2 .string "SW1 = 0,SW2 = 1";

.byte 0x00

strg3 .string "SW1 = 1,SW2 = 1";

.byte 0x00

strg4 .string "SW1 = 0,SW2 = 0";

.byte 0x00

.byte 0x0d,0x0a

.byte 0x00 ; null terminate the string with

.text ;program start

.global \_START ;define entry point

;----------------------------------------------------------------

START mov.w #300h,SP ; Initialize 'x1121 stackpointer

StopWDT mov.w #WDTPW+WDTHOLD,&WDTCTL ; Stop WDT

SetupP1 bis.b #02h,&P2DIR ;P2.2 output

call #Init\_UART ;initialize the uart

mov.b #0x00,&P2OUT ;turn off the LEDs

mov.b #0x00, R8 ;intiate counter register 8 to 0

Mainloop

cmp.b #0x02, &P1IN ;check if SW1 is pressed, if so jump to S1

jeq S1

cmp.b #0x01, &P1IN ;check if SW2 is pressed, if so jump to S2

jeq S2

cmp.b #0x00, &P1IN ;check if both switches are pressed

jeq Both

jmp Neither ;default is neither are pressed

S1

cmp.b #0x01, R8 ;check if counter is one, if so SW1 is

jeq Mainloop ;already pressed and printed, jump to main

mov #strg1,R6 ;if not set up string pointer

call #send ;send char to send, set counter to 1 meaning SW1

mov.b #0x01, R8 ;was last pressed switch

jmp Mainloop

S2

cmp.b #0x02, R8 ;check if counter is two, if so SW2 is

jeq Mainloop ;already pressed and printed, jump to main

mov #strg2,R6 ;if not set up string pointer

call #send ;send char to send, set counter to 2 meaning SW2

mov.b #0x02, R8 ;was last pressed switch

jmp Mainloop

Both

cmp.b #0x03, R8 ;same thing for both,

jeq Mainloop ;use 3 for counter

mov #strg3,R6 ;if not send in string3 and set counter to 3

call #send

mov.b #0x03, R8

jmp Mainloop

Neither

cmp.b #0x00, R8 ;if nothing is pressed, mainloop comes here

jeq Mainloop ;checks if neither was already printed, if so jmp main

mov #strg4,R6 ;otherwise print and change counter

call #send

mov.b #0x00, R8

jmp Mainloop

;loop to send in characters to OUTA

send

mov.b @R6, R4 ;move 1 char of string1 to R4 for OUTA

cmp.b #0, R4 ;check null, if so return

jeq L3

call #OUTA\_UART ;output character

inc R6 ;increment string pointer

mov.b @R6, R7 ;move next character into R7

cmp.b #0x00, R7 ;check for NULL character unless repeat

jnz send

mov.b #0x0D, R4 ;go to next line and

call #OUTA\_UART ;move cursor to beginnning of line

mov.b #0x0A, R4

call #OUTA\_UART

L3 ret

**Conclusion:**

In this lab, we learned and developed programs that used serial communication using MSP430 Universal Asynchronous Receiver and Transmitter (UART). We developed several C and Assembly language programs utilizing MSP430 UART for serial communication. We developed several programs that utilized the MSP430 UART communication in both assembly language and C language. We learned how to print a character inputted from a computer keyboard, print a string, print a user inputted string inputted from a computer keyboard, control LED’s with a computer keyboard, and display the state of buttons using serial communication and display on a HyperTerminal program.