**C4:**

**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** x;

**volatile** **unsigned** **char** z;

**volatile** **unsigned** **short** **int** b=0;

WDTCTL = WDTPW + WDTHOLD;

Init\_UART();

**for** (;;){

b=0;

**int** y;

**char** a;

**for**(y=0;y<8;y++){

a = INCHAR\_UART();

OUTA\_UART(a);

a=a-0x30;

a=a<<(7-y);

b=b|a;

}

OUTA\_UART(0x0A); //newline

OUTA\_UART(0x0D);

x= b & 0xF0;

z= b & 0x0F;

x=x>>4;

**if**(x>=0xA && x <=0xF) //check if hex value is a letter A-F

{

x+=0x37; //if yes subtract 37 to get decimal value

}

**else** **if** (x>=0x0 && x<=0x9) //check if hex value is number 0-9

{

x+=0x30; //if yes subtract 30 to get decimal value

}

**if**(z>=0xA && z<=0xF)

{

z+=0x37;

}

**else** **if** (z>=0x0 && z<=0x9)

{

z+=0x30;

}

OUTA\_UART(x);

OUTA\_UART(z);

OUTA\_UART(0x0A); //newline

OUTA\_UART(0x0D);

}

}

**void** **OUTA\_UART**(**unsigned** **char** A){

**do**{

}**while** ((IFG2&0x02)==0);

UCA0TXBUF =A;

}

**unsigned** **char** **INCHAR\_UART**(**void**){

**do**{

}**while** ((IFG2&0x01)==0);

**return** (UCA0RXBUF);

}

**void** **Init\_UART**(**void**){

P2SEL=0x30;

UCA0CTL0=0;

UCA0CTL1= 0x41;

UCA0BR1=0;

UCA0BR0=3;

UCA0MCTL=0x06;

UCA0STAT=0;

UCA0CTL1=0x40;

IE2=0;

}

**C5**

**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** b;

WDTCTL = WDTPW + WDTHOLD;

Init\_UART();

**for** (;;) {

**char** str[32];

**int** x=0;

**int** a=0;

**for** (x = 0; x < 32; x++) { // for loop to copy to a string

a = INCHAR\_UART();

str[x] = a;

OUTA\_UART(a); // print string to hyperterminal

**if** (a == 0x0D) { // if enter key is pressed break from loop

OUTA\_UART(0x0A); //newline

OUTA\_UART(0x0D);

**break**;

}

}

OUTA\_UART(0x0A); //newline

OUTA\_UART(0x0D);

**int** i=0; //algorithm for sorting

**int** j=0;

**int** temp;

**for** (i = 0; i < x - 1; i++) {

**for** (j = 0; j < x - 1; j++) {

**if** (str[j] > str[j + 1]) {

temp = str[j];

str[j] = str[j + 1];

str[j + 1] = temp;

}

}

}

**int** y =0;// while x decrements y increments and prints the sorted string

**while**(x>=0){

OUTA\_UART(str[x-(x-y)]);

y++;

x--;

}

OUTA\_UART(0x0A); //newline

OUTA\_UART(0x0D);

}

}

**void** **OUTA\_UART**(**unsigned** **char** A) {

**do** {

} **while** ((IFG2 & 0x02) == 0);

UCA0TXBUF = A;

}

**unsigned** **char** **INCHAR\_UART**(**void**) {

**do** {

} **while** ((IFG2 & 0x01) == 0);

**return** (UCA0RXBUF);

}

**void** **Init\_UART**(**void**) {

P2SEL = 0x30;

UCA0CTL0 = 0;

UCA0CTL1 = 0x41;

UCA0BR1 = 0;

UCA0BR0 = 3;

UCA0MCTL = 0x06;

UCA0STAT = 0;

UCA0CTL1 = 0x40;

IE2 = 0;

}

**ASSEMBLY 1:**

.cdecls C,LIST,"msp430fg4618.h"

**.sect** ".const"

.bss label, 4

**.word** 0x1234

strg2 .string "Laboratory #2 for EEL4742 embedded Systems"

**.byte** 0x0d,0x0a ;

**.byte** 0x00

**.text**

**.global** \_START

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

**START** **mov.w** #300h,SP

StopWDT **mov.w** #WDTPW+WDTHOLD,&WDTCTL

**call** #Init\_UART

Mainloop

**call** #INCHAR\_UART

**call** #OUTA\_UART

**mov** R4, R7

**call** #INCHAR\_UART

**call** #OUTA\_UART

**mov** R4,R8

LETTERCHECK **cmp.b** #0x41, R7

**jge** letter1

**jmp** NUMCHECK

letter1 **cmp.b** #0x47, R7

**jge** EXIT

**jmp** letter2

letter2 **sub.b** #0x37, R7

**jmp** BSHIFT

NUMCHECK **cmp.b** #0x30, R7

**jge** number1

**jmp** EXIT

number1 **cmp.b** #0x39, R7

**jge** Mainloop

**jmp** number2

number2 **sub.b** #0x30,R7

**jmp** BSHIFT

BSHIFT **mov.b** R7, R4

;call #OUTA\_UART

**rla** R4

**rla** R4

**rla** R4

**rla** R4

NEXT **cmp.b** #0x41, R8

**jge** letter3

**jmp** NUMCHECK2

letter3 **cmp.b** #0x47, R8

**jge** EXIT

**jmp** letter4

letter4 **sub.b** #0x37, R8

**jmp** PRINT

NUMCHECK2 **cmp.b** #0x30, R8

**jge** number3

**jmp** EXIT

number3 **cmp.b** #0x39, R8

**jge** EXIT

**jmp** number4

number4 **sub.b** #0x30,R8

**jmp** PRINT

PRINT **add.b** R8,R4

**call** #OUTA\_UART

EXIT **jmp** Mainloop

;

OUTA\_UART

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

; prints to the screen the ASCII value stored in register 4 and

; uses register 5 as a temp value

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

; IFG2 register (1) = 1 transmit buffer is empty,

; UCA0TXBUF 8 bit transmit buffer

; wait for the transmit buffer to be empty before sending the

; data out

**push** R5

lpa **mov.b** &IFG2,R5

**and.b** #0x02,R5

**cmp.b** #0x00,R5

**jz** lpa

; send the data to the transmit buffer UCA0TXBUF = A;

**mov.b** R4,&UCA0TXBUF

**pop** R5

**ret**

INCHAR\_UART

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

; returns the ASCII value in register 4

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

; IFG2 register (0) = 1 receive buffer is full,

; UCA0RXBUF 8 bit receive buffer

; wait for the receive buffer is full before getting the data

**push** R5

lpb **mov.b** &IFG2,R5

**and.b** #0x01,R5

**cmp.b** #0x00,R5

**jz** lpb

**mov.b** &UCA0RXBUF,R4

**pop** R5

; go get the char from the receive buffer

**ret**

Init\_UART

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

; Initialization code to set up the uart on the experimenter board to 8 data,

; 1 stop, no parity, and 9600 baud, polling operation

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

;P2SEL=0x30;

; transmit and receive to port 2 b its 4 and 5

**mov.b** #0x30,&P2SEL

; Bits p2.4 transmit and p2.5 receive UCA0CTL0=0

; 8 data, no parity 1 stop, uart, async

**mov.b** #0x00,&UCA0CTL0

; (7)=1 (parity), (6)=1 Even, (5)= 0 lsb first,

; (4)= 0 8 data / 1 7 data, (3) 0 1 stop 1 / 2 stop, (2-1) --

; UART mode, (0) 0 = async

; UCA0CTL1= 0x41;

**mov.b** #0x41,&UCA0CTL1

; select ALK 32768 and put in software reset the UART

; (7-6) 00 UCLK, 01 ACLK (32768 hz), 10 SMCLK, 11 SMCLK

; (0) = 1 reset

;UCA0BR1=0;

; upper byte of divider clock word

**mov.b** #0x00,&UCA0BR1

;UCA0BR0=3; ;

; clock divide from a clock to bit clock 32768/9600 = 3.413

**mov.b** #0x03,&UCA0BR0

; UCA0BR1:UCA0BR0 two 8 bit reg to from 16 bit clock divider

; for the baud rate

;UCA0MCTL=0x06;

; low frequency mode module 3 modulation pater used for the bit

; clock

**mov.b** #0x06,&UCA0MCTL

;UCA0STAT=0;

; do not loop the transmitter back to the receiver for echoing

**mov.b** #0x00,&UCA0STAT

; (7) = 1 echo back trans to rec

; (6) = 1 framing, (5) = 1 overrun, (4) =1 Parity, (3) = 1 break

; (0) = 2 transmitting or receiving data

;UCA0CTL1=0x40;

; take UART out of reset

**mov.b** #0x40,&UCA0CTL1

;IE2=0;

; turn transmit interrupts off

**mov.b** #0x00,&IE2

; (0) = 1 receiver buffer Interrupts enabled

; (1) = 1 transmit buffer Interrupts enabled

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

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

; IFG2 register (0) = 1 receiver buffer is full, UCA0RXIFG

; IFG2 register (1) = 1 transmit buffer is empty, UCA0RXIFG

; UCA0RXBUF 8 bit receiver buffer, UCA0TXBUF 8 bit transmit

; buffer

**ret**

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

; Interrupt Vectors

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

**.sect** ".reset" ; MSP430 RESET Vector

**.short** **START** ;

.end

**ASSEMBLY 2:**

.cdecls C,LIST,"msp430fg4618.h"

**.sect** ".const"

.bss label, 4

**.word** 0x1234

strg2 .string "Laboratory #2 for EEL4742 embedded Systems"

**.byte** 0x0d,0x0a ;

**.byte** 0x00

**.text**

**.global** \_START

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

**START** **mov.w** #300h,SP

StopWDT **mov.w** #WDTPW+WDTHOLD,&WDTCTL

**call** #Init\_UART

Mainloop

**call** #INCHAR\_UART

**call** #OUTA\_UART

**mov.b** R4, R7

**and.b** #0xF0, R7

**mov.b** R4, R8

**and.b** #0x0F, R8

**rra** R7

**rra** R7

**rra** R7

**rra** R7

LETTERCHECK **cmp.b** #0x0A, R7

**jge** letter1

**jmp** NUMCHECK

letter1 **cmp.b** #0x0F, R7

**jge** EXIT

**jmp** letter2

letter2 **add.b** #0x37, R7

**jmp** BSHIFT

NUMCHECK **cmp.b** #0x0, R7

**jge** number1

**jmp** EXIT

number1 **cmp.b** #0x9, R7

**jge** Mainloop

**jmp** number2

number2 **add.b** #0x30,R7

**jmp** BSHIFT

BSHIFT **mov.b** R7, R4

;call #OUTA\_UART

NEXT **cmp.b** #0x0A, R8

**jge** letter3

**jmp** NUMCHECK2

letter3 **cmp.b** #0x0F, R8

**jge** EXIT

**jmp** letter4

letter4 **add.b** #0x37, R8

**jmp** PRINT

NUMCHECK2 **cmp.b** #0x0, R8

**jge** number3

**jmp** EXIT

number3 **cmp.b** #0x9, R8

**jge** EXIT

**jmp** number4

number4 **add.b** #0x30,R8

**jmp** PRINT

PRINT **mov.b** R7, R4

**call** #OUTA\_UART

**mov.b** R8, R4

**call** #OUTA\_UART

EXIT **jmp** Mainloop

OUTA\_UART

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

; prints to the screen the ASCII value stored in register 4 and

; uses register 5 as a temp value

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

; IFG2 register (1) = 1 transmit buffer is empty,

; UCA0TXBUF 8 bit transmit buffer

; wait for the transmit buffer to be empty before sending the

; data out

**push** R5

lpa **mov.b** &IFG2,R5

**and.b** #0x02,R5

**cmp.b** #0x00,R5

**jz** lpa

; send the data to the transmit buffer UCA0TXBUF = A;

**mov.b** R4,&UCA0TXBUF

**pop** R5

**ret**

INCHAR\_UART

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

; returns the ASCII value in register 4

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

; IFG2 register (0) = 1 receive buffer is full,

; UCA0RXBUF 8 bit receive buffer

; wait for the receive buffer is full before getting the data

**push** R5

lpb **mov.b** &IFG2,R5

**and.b** #0x01,R5

**cmp.b** #0x00,R5

**jz** lpb

**mov.b** &UCA0RXBUF,R4

**pop** R5

; go get the char from the receive buffer

**ret**

Init\_UART

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

; Initialization code to set up the uart on the experimenter board to 8 data,

; 1 stop, no parity, and 9600 baud, polling operation

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

;P2SEL=0x30;

; transmit and receive to port 2 b its 4 and 5

**mov.b** #0x30,&P2SEL

; Bits p2.4 transmit and p2.5 receive UCA0CTL0=0

; 8 data, no parity 1 stop, uart, async

**mov.b** #0x00,&UCA0CTL0

; (7)=1 (parity), (6)=1 Even, (5)= 0 lsb first,

; (4)= 0 8 data / 1 7 data, (3) 0 1 stop 1 / 2 stop, (2-1) --

; UART mode, (0) 0 = async

; UCA0CTL1= 0x41;

**mov.b** #0x41,&UCA0CTL1

; select ALK 32768 and put in software reset the UART

; (7-6) 00 UCLK, 01 ACLK (32768 hz), 10 SMCLK, 11 SMCLK

; (0) = 1 reset

;UCA0BR1=0;

; upper byte of divider clock word

**mov.b** #0x00,&UCA0BR1

;UCA0BR0=3; ;

; clock divide from a clock to bit clock 32768/9600 = 3.413

**mov.b** #0x03,&UCA0BR0

; UCA0BR1:UCA0BR0 two 8 bit reg to from 16 bit clock divider

; for the baud rate

;UCA0MCTL=0x06;

; low frequency mode module 3 modulation pater used for the bit

; clock

**mov.b** #0x06,&UCA0MCTL

;UCA0STAT=0;

; do not loop the transmitter back to the receiver for echoing

**mov.b** #0x00,&UCA0STAT

; (7) = 1 echo back trans to rec

; (6) = 1 framing, (5) = 1 overrun, (4) =1 Parity, (3) = 1 break

; (0) = 2 transmitting or receiving data

;UCA0CTL1=0x40;

; take UART out of reset

**mov.b** #0x40,&UCA0CTL1

;IE2=0;

; turn transmit interrupts off

**mov.b** #0x00,&IE2

; (0) = 1 receiver buffer Interrupts enabled

; (1) = 1 transmit buffer Interrupts enabled

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

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

; IFG2 register (0) = 1 receiver buffer is full, UCA0RXIFG

; IFG2 register (1) = 1 transmit buffer is empty, UCA0RXIFG

; UCA0RXBUF 8 bit receiver buffer, UCA0TXBUF 8 bit transmit

; buffer

**ret**

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

; Interrupt Vectors

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

**.sect** ".reset" ; MSP430 RESET Vector

**.short** **START** ;

.end

**ASSEMBLY 3:**

.cdecls C,LIST,"msp430fg4618.h"

**.sect** ".const"

.bss label, 4

**.word** 0x1234

strg2 .string "Laboratory #2 for EEL4742 embedded Systems"

**.byte** 0x0d,0x0a ;

**.byte** 0x00

**.text**

**.global** \_START

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

**START** **mov.w** #300h,SP

StopWDT **mov.w** #WDTPW+WDTHOLD,&WDTCTL

**call** #Init\_UART

Mainloop

**call** #INCHAR\_UART

**call** #OUTA\_UART

LOWERCHECK **cmp.b** #0x61, R4

**jge** letter1

**jmp** UPPERCHECK

letter1 **cmp.b** #0x7A, R4

**jge** EXIT

**jmp** letter2

letter2 **sub.b** #0x20, R4

**call** #OUTA\_UART

**jmp** Mainloop

UPPERCHECK **cmp.b** #0x41, R4

**jge** number1

**jmp** EXIT

number1 **cmp.b** #0x5C, R4

**jge** Mainloop

**jmp** number2

number2 **add.b** #0x20,R4

**jmp** BSHIFT

BSHIFT **call** #OUTA\_UART

EXIT **jmp** Mainloop

OUTA\_UART

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

; prints to the screen the ASCII value stored in register 4 and

; uses register 5 as a temp value

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

; IFG2 register (1) = 1 transmit buffer is empty,

; UCA0TXBUF 8 bit transmit buffer

; wait for the transmit buffer to be empty before sending the

; data out

**push** R5

lpa **mov.b** &IFG2,R5

**and.b** #0x02,R5

**cmp.b** #0x00,R5

**jz** lpa

; send the data to the transmit buffer UCA0TXBUF = A;

**mov.b** R4,&UCA0TXBUF

**pop** R5

**ret**

INCHAR\_UART

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

; returns the ASCII value in register 4

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

; IFG2 register (0) = 1 receive buffer is full,

; UCA0RXBUF 8 bit receive buffer

; wait for the receive buffer is full before getting the data

**push** R5

lpb **mov.b** &IFG2,R5

**and.b** #0x01,R5

**cmp.b** #0x00,R5

**jz** lpb

**mov.b** &UCA0RXBUF,R4

**pop** R5

; go get the char from the receive buffer

**ret**

Init\_UART

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

; Initialization code to set up the uart on the experimenter board to 8 data,

; 1 stop, no parity, and 9600 baud, polling operation

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

;P2SEL=0x30;

; transmit and receive to port 2 b its 4 and 5

**mov.b** #0x30,&P2SEL

; Bits p2.4 transmit and p2.5 receive UCA0CTL0=0

; 8 data, no parity 1 stop, uart, async

**mov.b** #0x00,&UCA0CTL0

; (7)=1 (parity), (6)=1 Even, (5)= 0 lsb first,

; (4)= 0 8 data / 1 7 data, (3) 0 1 stop 1 / 2 stop, (2-1) --

; UART mode, (0) 0 = async

; UCA0CTL1= 0x41;

**mov.b** #0x41,&UCA0CTL1

; select ALK 32768 and put in software reset the UART

; (7-6) 00 UCLK, 01 ACLK (32768 hz), 10 SMCLK, 11 SMCLK

; (0) = 1 reset

;UCA0BR1=0;

; upper byte of divider clock word

**mov.b** #0x00,&UCA0BR1

;UCA0BR0=3; ;

; clock divide from a clock to bit clock 32768/9600 = 3.413

**mov.b** #0x03,&UCA0BR0

; UCA0BR1:UCA0BR0 two 8 bit reg to from 16 bit clock divider

; for the baud rate

;UCA0MCTL=0x06;

; low frequency mode module 3 modulation pater used for the bit

; clock

**mov.b** #0x06,&UCA0MCTL

;UCA0STAT=0;

; do not loop the transmitter back to the receiver for echoing

**mov.b** #0x00,&UCA0STAT

; (7) = 1 echo back trans to rec

; (6) = 1 framing, (5) = 1 overrun, (4) =1 Parity, (3) = 1 break

; (0) = 2 transmitting or receiving data

;UCA0CTL1=0x40;

; take UART out of reset

**mov.b** #0x40,&UCA0CTL1

;IE2=0;

; turn transmit interrupts off

**mov.b** #0x00,&IE2

; (0) = 1 receiver buffer Interrupts enabled

; (1) = 1 transmit buffer Interrupts enabled

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

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

; IFG2 register (0) = 1 receiver buffer is full, UCA0RXIFG

; IFG2 register (1) = 1 transmit buffer is empty, UCA0RXIFG

; UCA0RXBUF 8 bit receiver buffer, UCA0TXBUF 8 bit transmit

; buffer

**ret**

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

; Interrupt Vectors

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

**.sect** ".reset" ; MSP430 RESET Vector

**.short** **START** ;

.end

**Assembly 4:**

.cdecls C,LIST,"msp430fg4618.h"

**.sect** ".const"

.bss label, 4

**.word** 0x1234

strg2 .string "Laboratory #2 for EEL4742 embedded Systems"

**.byte** 0x0d,0x0a ;

**.byte** 0x00

**.text**

**.global** \_START

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

**START** **mov.w** #300h,SP

StopWDT **mov.w** #WDTPW+WDTHOLD,&WDTCTL

**call** #Init\_UART

Mainloop

**mov.b** #0x0,R9 ;setting up for loop counter

**mov.b** #0x0, R11

ForLoop

**call** #INCHAR\_UART

**call** #OUTA\_UART

**sub.b** #0x30,R4

**mov.b** R9, R10

decrement

**cmp.b** #0x7, R10

**jz** Next0

**inc** R10

**rla** R4 ;shift UARTinput until satisfied

**jmp** decrement

Next0 **inc** R9

**bis.b** R4,R11

**cmp.b** #0x8, R9

**jz** Next1

**jmp** ForLoop

Next1

**mov.b** R11, R7

**and.b** #0xF0, R7

**mov.b** R11, R8

**and.b** #0x0F, R8

**rra** R7

**rra** R7

**rra** R7

**rra** R7

LETTERCHECK **cmp.b** #0x0A, R7

**jge** letter1

**jmp** NUMCHECK

letter1 **cmp.b** #0x1F, R7

**jge** EXIT

**jmp** letter2

letter2 **add.b** #0x37, R7

**jmp** BSHIFT

NUMCHECK **cmp.b** #0x0, R7

**jge** number1

**jmp** EXIT

number1 **cmp.b** #0x9, R7

**jge** Mainloop

**jmp** number2

number2 **add.b** #0x30,R7

**jmp** BSHIFT

BSHIFT **mov.b** R7, R4

;call #OUTA\_UART

NEXT **cmp.b** #0x0A, R8

**jge** letter3

**jmp** NUMCHECK2

letter3 **cmp.b** #0x1F, R8

**jge** EXIT

**jmp** letter4

letter4 **add.b** #0x37, R8

**jmp** PRINT

NUMCHECK2 **cmp.b** #0x0, R8

**jge** number3

**jmp** EXIT

number3 **cmp.b** #0x9, R8

**jge** EXIT

**jmp** number4

number4 **add.b** #0x30,R8

**jmp** PRINT

PRINT

**mov.b** R7, R4

**call** #OUTA\_UART

**mov.b** R8, R4

**call** #OUTA\_UART

EXIT **jmp** Mainloop

OUTA\_UART

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

; prints to the screen the ASCII value stored in register 4 and

; uses register 5 as a temp value

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

; IFG2 register (1) = 1 transmit buffer is empty,

; UCA0TXBUF 8 bit transmit buffer

; wait for the transmit buffer to be empty before sending the

; data out

**push** R5

lpa **mov.b** &IFG2,R5

**and.b** #0x02,R5

**cmp.b** #0x00,R5

**jz** lpa

; send the data to the transmit buffer UCA0TXBUF = A;

**mov.b** R4,&UCA0TXBUF

**pop** R5

**ret**

INCHAR\_UART

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

; returns the ASCII value in register 4

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

; IFG2 register (0) = 1 receive buffer is full,

; UCA0RXBUF 8 bit receive buffer

; wait for the receive buffer is full before getting the data

**push** R5

lpb **mov.b** &IFG2,R5

**and.b** #0x01,R5

**cmp.b** #0x00,R5

**jz** lpb

**mov.b** &UCA0RXBUF,R4

**pop** R5

; go get the char from the receive buffer

**ret**

Init\_UART

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

; Initialization code to set up the uart on the experimenter board to 8 data,

; 1 stop, no parity, and 9600 baud, polling operation

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

;P2SEL=0x30;

; transmit and receive to port 2 b its 4 and 5

**mov.b** #0x30,&P2SEL

; Bits p2.4 transmit and p2.5 receive UCA0CTL0=0

; 8 data, no parity 1 stop, uart, async

**mov.b** #0x00,&UCA0CTL0

; (7)=1 (parity), (6)=1 Even, (5)= 0 lsb first,

; (4)= 0 8 data / 1 7 data, (3) 0 1 stop 1 / 2 stop, (2-1) --

; UART mode, (0) 0 = async

; UCA0CTL1= 0x41;

**mov.b** #0x41,&UCA0CTL1

; select ALK 32768 and put in software reset the UART

; (7-6) 00 UCLK, 01 ACLK (32768 hz), 10 SMCLK, 11 SMCLK

; (0) = 1 reset

;UCA0BR1=0;

; upper byte of divider clock word

**mov.b** #0x00,&UCA0BR1

;UCA0BR0=3; ;

; clock divide from a clock to bit clock 32768/9600 = 3.413

**mov.b** #0x03,&UCA0BR0

; UCA0BR1:UCA0BR0 two 8 bit reg to from 16 bit clock divider

; for the baud rate

;UCA0MCTL=0x06;

; low frequency mode module 3 modulation pater used for the bit

; clock

**mov.b** #0x06,&UCA0MCTL

;UCA0STAT=0;

; do not loop the transmitter back to the receiver for echoing

**mov.b** #0x00,&UCA0STAT

; (7) = 1 echo back trans to rec

; (6) = 1 framing, (5) = 1 overrun, (4) =1 Parity, (3) = 1 break

; (0) = 2 transmitting or receiving data

;UCA0CTL1=0x40;

; take UART out of reset

**mov.b** #0x40,&UCA0CTL1

;IE2=0;

; turn transmit interrupts off

**mov.b** #0x00,&IE2

; (0) = 1 receiver buffer Interrupts enabled

; (1) = 1 transmit buffer Interrupts enabled

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

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

; IFG2 register (0) = 1 receiver buffer is full, UCA0RXIFG

; IFG2 register (1) = 1 transmit buffer is empty, UCA0RXIFG

; UCA0RXBUF 8 bit receiver buffer, UCA0TXBUF 8 bit transmit

; buffer

**ret**

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

; Interrupt Vectors

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

**.sect** ".reset" ; MSP430 RESET Vector

**.short** **START** ;

.end