**UART Lab**

**Links to videos:**

**Task 1:** http://screencast.com/t/3yrRLX019c

**Task 2:** http://screencast.com/t/IpEamdDI0bw

**Task 3:** http://screencast.com/t/CCSLcDfBXKbK

**Task 1: Adding comments to original code.**

//video: http://screencast.com/t/3yrRLX019c

#include <stdint.h>

#include <stdbool.h>

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "driverlib/gpio.h"

#include "driverlib/pin\_map.h"

#include "driverlib/sysctl.h"

#include "driverlib/uart.h"

int main(void) {

SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ); //setup system clock 50 MHz

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0); //enable UART Port

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA); //enable port A

GPIOPinConfigure(GPIO\_PA0\_U0RX); //configure PA0 as Rx

GPIOPinConfigure(GPIO\_PA1\_U0TX); //configure PA1 as Tx

GPIOPinTypeUART(GPIO\_PORTA\_BASE, GPIO\_PIN\_0 | GPIO\_PIN\_1); //configure A0 and A1 as UART

UARTConfigSetExpClk(UART0\_BASE, SysCtlClockGet(), 115200, // Configure UART speed 115200, 8bits, 1 stop bit, no parity

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE | UART\_CONFIG\_PAR\_NONE));

UARTCharPut(UART0\_BASE, 'E'); //put these characters to serial port

UARTCharPut(UART0\_BASE, 'n');

UARTCharPut(UART0\_BASE, 't');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'r');

UARTCharPut(UART0\_BASE, ' ');

UARTCharPut(UART0\_BASE, 'T');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'x');

UARTCharPut(UART0\_BASE, 't');

UARTCharPut(UART0\_BASE, ':');

UARTCharPut(UART0\_BASE, ' ');

while (1) //infinite loop

{

if (UARTCharsAvail(UART0\_BASE)) UARTCharPut(UART0\_BASE, UARTCharGet(UART0\_BASE)); //get char from keyboard and send through serial port

}

}

**Task 2: Modify original code to print capital letters when small letters are entered and vice versa, and show on serial terminal.**

//video: http://screencast.com/t/IpEamdDI0bw

#include <stdint.h>

#include <stdbool.h>

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "driverlib/gpio.h"

#include "driverlib/pin\_map.h"

#include "driverlib/sysctl.h"

#include "driverlib/uart.h"

int main(void) {

SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ); //setup system clock 50 MHz

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0); //enable UART Port

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA); //enable port A

GPIOPinConfigure(GPIO\_PA0\_U0RX); //configure PA0 as Rx

GPIOPinConfigure(GPIO\_PA1\_U0TX); //configure PA1 as Tx

GPIOPinTypeUART(GPIO\_PORTA\_BASE, GPIO\_PIN\_0 | GPIO\_PIN\_1); //configure A0 and A1 as UART

UARTConfigSetExpClk(UART0\_BASE, SysCtlClockGet(), 115200, // Configure UART speed 115200, 8bits, 1 stop bit, no parity

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE | UART\_CONFIG\_PAR\_NONE));

UARTCharPut(UART0\_BASE, 'E'); //put these characters to serial port

UARTCharPut(UART0\_BASE, 'n');

UARTCharPut(UART0\_BASE, 't');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'r');

UARTCharPut(UART0\_BASE, ' ');

UARTCharPut(UART0\_BASE, 'T');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'x');

UARTCharPut(UART0\_BASE, 't');

UARTCharPut(UART0\_BASE, ':');

UARTCharPut(UART0\_BASE, ' ');

int serialChar; //will store character from serial port

while (1) //infinite loop

{

if (UARTCharsAvail(UART0\_BASE)){ //wait for next available char on serial port

serialChar = UARTCharGet(UART0\_BASE); //get char from serial port

if (serialChar >= 97 && serialChar <= 122) //check if char is lower case

serialChar -= 32; //if it is, subtract 32 to make upper case

else if(serialChar >= 65 && serialChar <= 90) //if char is upper case, add 32 to make lower case

serialChar += 32;

UARTCharPut(UART0\_BASE, serialChar); //put char on serial port

}

}

}

**Task 3: Continuously display the temperature of the internal temperature sensor on serial terminal.**

//video: http://screencast.com/t/CCSLcDfBXKbK

#include <stdint.h>

#include <stdbool.h>

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "driverlib/gpio.h"

#include "driverlib/pin\_map.h"

#include "driverlib/sysctl.h"

#include "driverlib/uart.h"

#include "driverlib/adc.h"

int main(void) {

//variables

int hundreds, tens, ones;

//temperature variables

uint32\_t ui32ADC0Value[4]; //will store 4 values from FIFO when using sequencer 1

volatile uint32\_t ui32TempAvg; //stores average of 4 sampled values

volatile uint32\_t ui32TempValueC; //stores temperature in Celsius

volatile uint32\_t ui32TempValueF; //stores temperature in Fahrenheit

SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ); //setup system clock 50 MHz

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0); //enable UART Port

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA); //enable port A

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0); //Enable ADC0

//configure ADC Sequencer 1

ADCSequenceConfigure(ADC0\_BASE, 1, ADC\_TRIGGER\_PROCESSOR, 0); //set sequencer 1 to trigger with CPU

ADCSequenceStepConfigure(ADC0\_BASE, 1, 0, ADC\_CTL\_TS); //configure step 0 from temp sensor

ADCSequenceStepConfigure(ADC0\_BASE, 1, 1, ADC\_CTL\_TS); //configure sequencer step 1 from temp sensor

ADCSequenceStepConfigure(ADC0\_BASE, 1, 2, ADC\_CTL\_TS); //configure sequencer step 2 from temp sensor

ADCSequenceStepConfigure(ADC0\_BASE,1,3,ADC\_CTL\_TS|ADC\_CTL\_IE|ADC\_CTL\_END); //configure sequencer step 3 from temp sensor and tell sequencer to finish

ADCSequenceEnable(ADC0\_BASE, 1);

//-----

GPIOPinConfigure(GPIO\_PA0\_U0RX); //configure PA0 as Rx

GPIOPinConfigure(GPIO\_PA1\_U0TX); //configure PA1 as Tx

GPIOPinTypeUART(GPIO\_PORTA\_BASE, GPIO\_PIN\_0 | GPIO\_PIN\_1); //configure A0 and A1 as UART

UARTConfigSetExpClk(UART0\_BASE, SysCtlClockGet(), 115200, // Configure UART speed 115200, 8bits, 1 stop bit, no parity

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE | UART\_CONFIG\_PAR\_NONE));

while (1) //infinite loop

{

UARTCharPut(UART0\_BASE, 'T'); //put these characters to serial port

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'm');

UARTCharPut(UART0\_BASE, 'p');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, 'r');

UARTCharPut(UART0\_BASE, 'a');

UARTCharPut(UART0\_BASE, 't');

UARTCharPut(UART0\_BASE, 'u');

UARTCharPut(UART0\_BASE, 'r');

UARTCharPut(UART0\_BASE, 'e');

UARTCharPut(UART0\_BASE, ' ');

UARTCharPut(UART0\_BASE, '(');

UARTCharPut(UART0\_BASE, 'F');

UARTCharPut(UART0\_BASE, ')');

UARTCharPut(UART0\_BASE, ':');

UARTCharPut(UART0\_BASE, ' ');

ADCIntClear(ADC0\_BASE, 1); //clear ADC0 interrupt

ADCProcessorTrigger(ADC0\_BASE, 1); //Trigger ADC0 sequencer 1

while(!ADCIntStatus(ADC0\_BASE, 1, false)) //wait for ADC conversion to finish

{

}

SysCtlDelay(6000000);

ADCSequenceDataGet(ADC0\_BASE, 1, ui32ADC0Value); //get data from FIFO and put into array

ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4; //calculate average temperature (+2/4 used for rounding)

ui32TempValueC = (1475 - ((2475 \* ui32TempAvg)) / 4096)/10; //calculate temp in Celsius

ui32TempValueF = ((ui32TempValueC \* 9) + 160) / 5; //calculate temp in Fahrenheit

//get hundreds, tens, and ones

hundreds = ui32TempValueF / 100;

tens = ui32TempValueF % 100 / 10;

ones = ui32TempValueF % 100 % 10;

//convert to ASCII numbers

hundreds += 48;

tens += 48;

ones += 48;

if(hundreds > '0') //if hundreds is 0, insert blank space to keep numbers alligned

UARTCharPut(UART0\_BASE, hundreds);

else

UARTCharPut(UART0\_BASE, ' ');

UARTCharPut(UART0\_BASE, tens); //send tens serially

UARTCharPut(UART0\_BASE, ones); //send ones serially

UARTCharPut(UART0\_BASE, '\n'); //new line

UARTCharPut(UART0\_BASE, '\r'); //carriage return

}

}