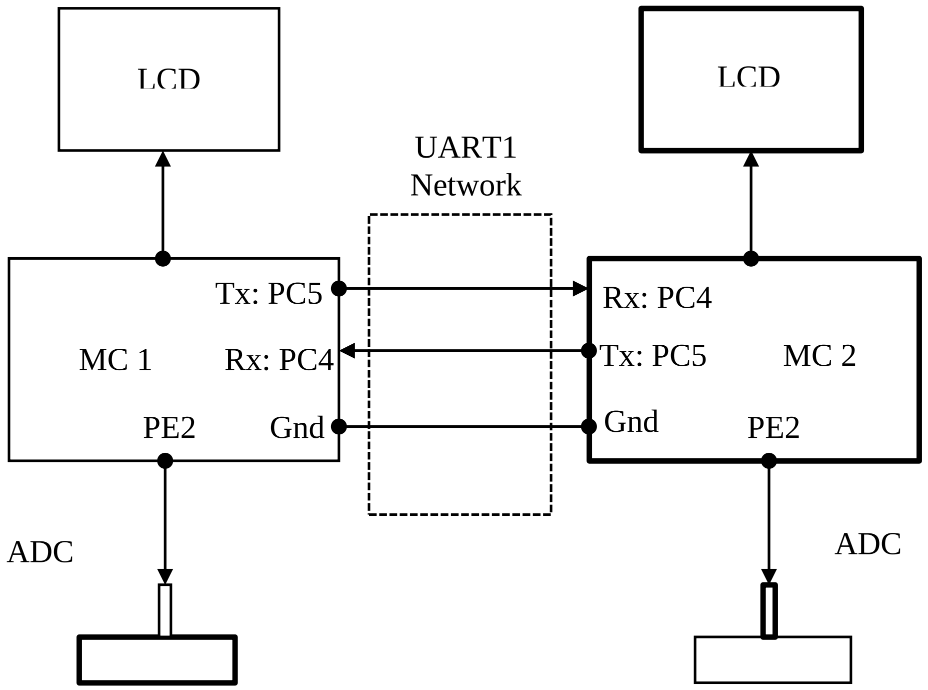
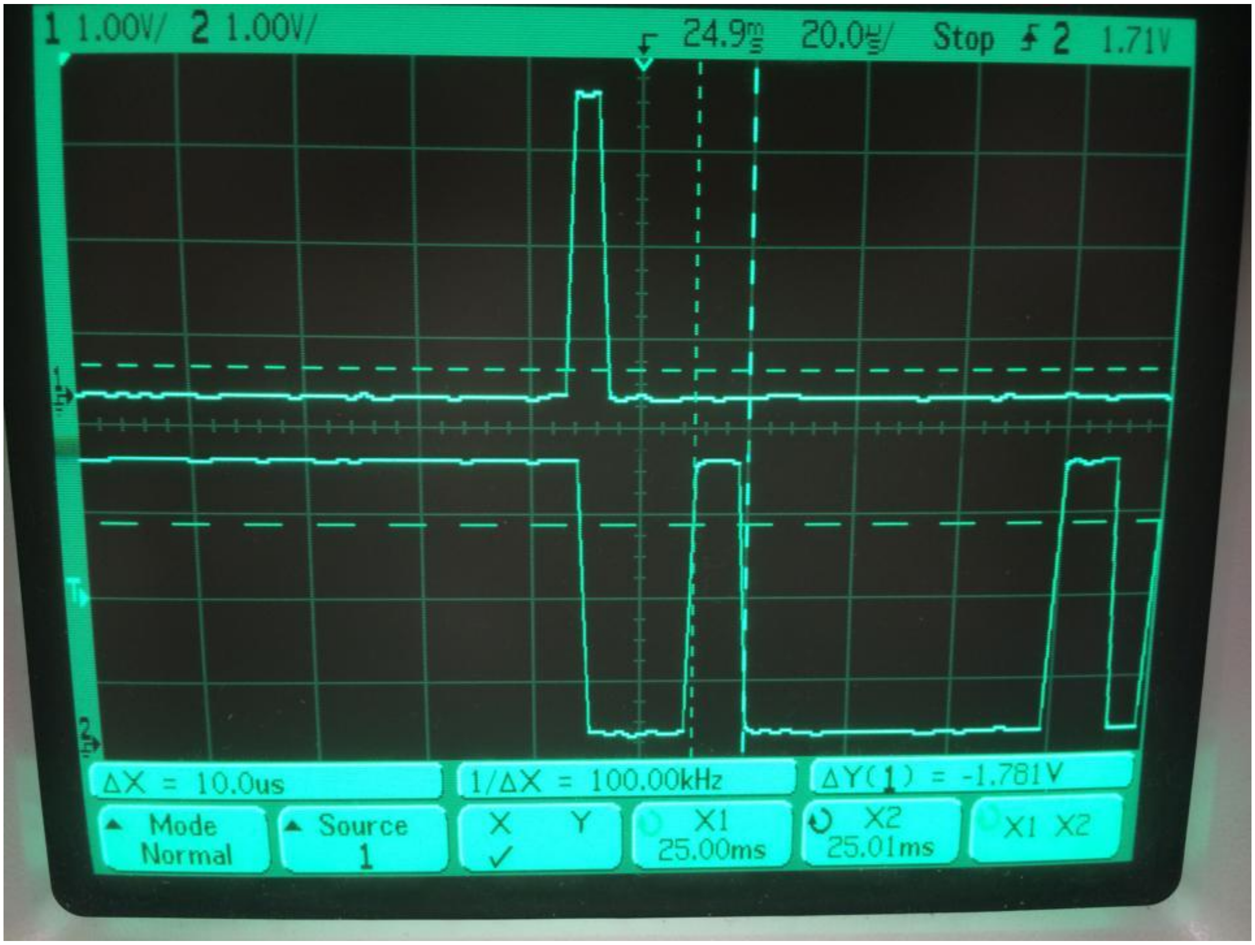
Ben Fu
Rochelle Roberts

Lab 9 Report



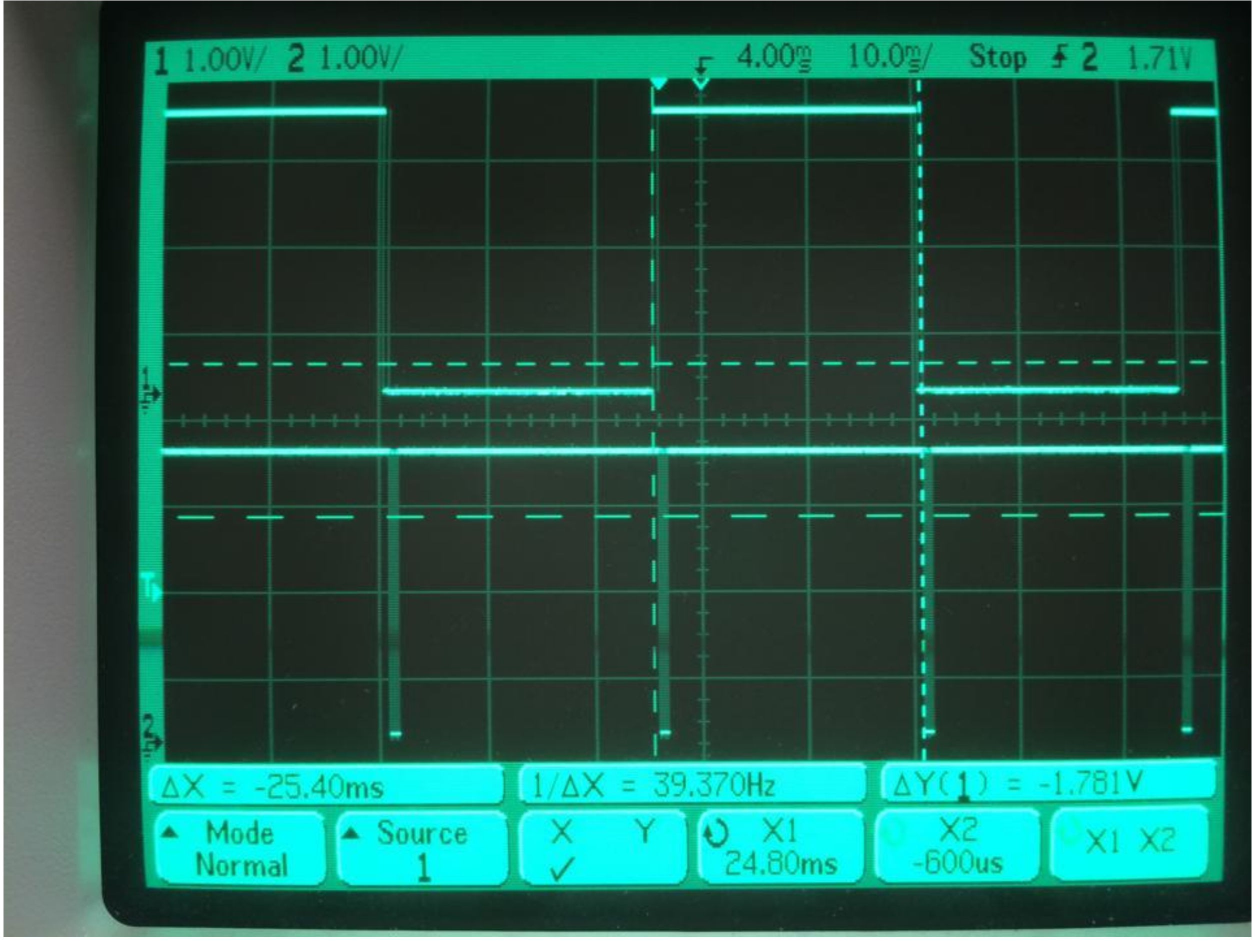
Ben Fu
Rochelle Roberts

Lab 9 Report



Ben Fu
Rochelle Roberts

Lab 9 Report



1 1.00V/ 2 1.00V/ 4.00ms 10.0ms/ Stop 2 1.71V

ΔX = -25.40ms
1/ΔX = 39.370Hz
ΔY(1) = -1.781V

Mode Normal
Source 1
X Y ✓
X1 24.80ms
X2 -600us
X1 X2

Ben Fu
Rochelle Roberts

Lab 9 Report

```
// Lab8.c
// Runs on LM4F120 or TM4C123
// Use the SysTick timer to request interrupts at a particular period.
// Ben Fu and Rochelle Roberts
// 4/7/14

// Analog Input connected to PE2=ADC1
// displays on Kentec EB-LM4F120-L35
// PF1 SysTick toggle


#include "lcd.h"
#include "pll.h"
#include "ADC.h"
#include "../inc/tm4c123gh6pm.h"
#include "UART.h"
#include "fifo.h"



void DisableInterrupts(void); // Disable interrupts
void EnableInterrupts(void);  // Enable interrupts
long StartCritical (void);    // previous I bit, disable interrupts
void EndCritical(long sr);    // restore I bit to previous value
void WaitForInterrupt(void);  // low power mode
#define PF1       (*((volatile unsigned long *)0x40025008))
#define PF2       (*((volatile unsigned long *)0x40025010))
#define PF3       (*((volatile unsigned long *)0x40025020))
void PortF_Init(void){unsigned long volatile delay;
      SYSCTL_RCGC2_R |= 0x00000020;   // 1) activate clock for Port F
  delay = SYSCTL_RCGC2_R;        //    allow time to stabilize
  GPIO_PORTF_DIR_R |= 0x04;      // 2) make PF2 output
  GPIO_PORTF_AFSEL_R &= ~0x04;    // 3) disable alt funct on PF2
  GPIO_PORTF_DEN_R |= 0x04;      // 4) enable digital I/O on PF2
}
void SysTick_Init(unsigned long period){
      NVIC_ST_CTRL_R =0;                                   //disable clock
during setup
      NVIC_ST_RELOAD_R = period - 1;            //reload value so interrupt
at 40Hz
      NVIC_ST_CURRENT_R =0;                           //write to clear
      NVIC_SYS_PRI3_R = (NVIC_SYS_PRI3_R & 0x00FFFFFF)|0x40000000; //Priotity
2
      NVIC_ST_CTRL_R = 0x00000007;  //enable with core clock and interrupts
}

//systick handler
unsigned long ADCMail;
unsigned char ADCStatus;
unsigned long TxCounter=0;    //initialize counter


//equation is POSITION = 0.4351(DATA)+180.35
unsigned long Convert(unsigned long input){unsigned long output;
      output = (4351*input+1803500)/10000;
      //output = 0.4351*(input)+180.35;
  return output;
}

void SysTick_Handler(void){
      PF2 ^=0x04;                   // 1.toggle heartbeat
      ADCMail = ADC_In();           // 2.sample ADC
      PF2 ^=0x04;                   // 3.toggle heartbeat
      ADCMail = Convert(ADCMail);   //4.convert fixed point
//send the 8-byte message
      UART1_OutChar(0x02);          //send first byte (STX)
      UART1_OutChar((ADCMail/1000)+0x30); //first digit
      UART1_OutChar(0x2E);          //.
      UART1_OutChar(((ADCMail%1000)/100)+0x30); //second digit
      UART1_OutChar((((ADCMail%1000)%100)/10)+0x30);        //third digit
      UART1_OutChar((((ADCMail%1000)%100)%10)+0x30);        //last digit
      UART1_OutChar(0x0D);          //send CR byte
      UART1_OutChar(0x03);          //send last byte (ETX)
      TxCounter++;                  // 6. Increment counter
      PF2 ^=0x04;                   // 7. toggle heartbeat
      ADCStatus = 1;          //ADCStatus=1 to indicate fresh sample in
ADCMail
}




unsigned long Data;      // 12-bit ADC
unsigned long Position;  // 32-bit fixed-point 0.001 cm

long letter;

int main1(){unsigned char i;
      PLL_Init();                   //initialize everything
      LCD_Init();
  LCD_SetTextColorRGB(YELLOW);
  PortF_Init();
      ADC_Init();
      UART1_Init();
      SysTick_Init(2000000);        //initialize interrupts
  EnableInterrupts();
      while(1){
            while(ADCStatus!=1){} //wait until fresh sample available
            ADCStatus = 0;                      //clear mailbox flag
            while(Fifo_Get(&letter) == 0){};    //if the buffer is empty do
nothing
            while(letter!=0x02){Fifo_Get(&letter);};        //wait until
letter is 0x02
            LCD_Goto(0,0);
            for(i=0;i<5;i++){
            Fifo_Get(&letter);            //read a bit
            LCD_OutChar(letter);    //output to display
            }
            LCD_OutString(" cm  ");
            //Position = ADCMail;
            //
    //LCD_OutFix(Position);
}
}

int main(){
      PLL_Init();                   //initialize everything
      LCD_Init();
  LCD_SetTextColorRGB(YELLOW);
  PortF_Init();
      ADC_Init();
      UART1_Init();
      SysTick_Init(2000000);
  EnableInterrupts();

      while(1){
            while(ADCStatus!=1){} //wait until fresh sample available
            ADCStatus = 0;                      //clear mailbox flag
            Position = ADCMail;
            LCD_Goto(0,0);
    LCD_OutFix(Position); LCD_OutString(" cm  ");
}
}

//tester
int Status[20]; // entries 0,7,12,19 should be false, others true
long GetData[10]; // entries 1 2 3 4 5 6 7 8 should be 1 2 3 4 5 6 7 8
int main3(void){
 Fifo_Init();
 for(;;){
 Status[0] = Fifo_Get(&GetData[0]); // should fail, empty
 Status[1] = Fifo_Put(1); // should succeed, 1
 Status[2] = Fifo_Put(2); // should succeed, 1 2
 Status[3] = Fifo_Put(3); // should succeed, 1 2 3
Status[4] = Fifo_Put(4); // should succeed, 1 2 3 4
 Status[5] = Fifo_Put(5); // should succeed, 1 2 3 4 5
 Status[6] = Fifo_Put(6); // should succeed, 1 2 3 4 5 6
 Status[7] = Fifo_Put(7); // should fail, 1 2 3 4 5 6
 Status[8] = Fifo_Get(&GetData[1]); // should succeed, 2 3 4 5 6
 Status[9] = Fifo_Get(&GetData[2]); // should succeed, 3 4 5 6
 Status[10] = Fifo_Put(7); // should succeed, 3 4 5 6 7
 Status[11] = Fifo_Put(8); // should succeed, 3 4 5 6 7 8
 Status[12] = Fifo_Put(9); // should fail, 3 4 5 6 7 8
 Status[13] = Fifo_Get(&GetData[3]); // should succeed, 4 5 6 7 8
 Status[14] = Fifo_Get(&GetData[4]); // should succeed, 5 6 7 8
 Status[15] = Fifo_Get(&GetData[5]); // should succeed, 6 7 8
 Status[16] = Fifo_Get(&GetData[6]); // should succeed, 7 8
 Status[17] = Fifo_Get(&GetData[7]); // should succeed, 8
 Status[18] = Fifo_Get(&GetData[8]); // should succeed, empty
 Status[19] = Fifo_Get(&GetData[9]); // should fail, empty
 }
}
```

Ben Fu
Rochelle Roberts

Lab 9 Report

```
//UART.c
// UART1_Int Transmitter
// Initialize UART1 Transmitter and set Baud Rate at 100,000 bps
//    call once

#include "../inc/tm4c123gh6pm.h"
#include "ADC.h"
#include "fifo.h"
#define PF1       (*((volatile unsigned long *)0x40025008))
#define PF2       (*((volatile unsigned long *)0x40025010))
#define PF3       (*((volatile unsigned long *)0x40025020))

unsigned long error;
unsigned long RxCounter;

void UART1_Init(void){
      Fifo_Init();
      SYSCTL_RCGC1_R |= 0x0002;                 //activate UART1
      SYSCTL_RCGC2_R |= 0x0004;                 //activate PortC
      UART1_CTL_R &= ~0x001;                    //disable UART during init
      UART1_IBRD_R = 50;                                    //bit
rate=(80000000/16*100000)=50
      UART1_FBRD_R = 0;
      UART1_LCRH_R = 0x0070;                    //8-bit word length, enable
FIFO
      //enable interrups
      UART1_IM_R |= 0x10;                                   //set RXIM
interrupt mask
      UART1_IFLS_R &= 0xFFFFFFD7;         //clear bits 5 and 3 of IFLS
      UART1_IFLS_R |= 0x10;                           //set bit 4 to enable
interrupts
      //set interrupr priority
      NVIC_PRI1_R = (NVIC_PRI1_R & 0xFF00FFFF)|0x00400000; //Priotity 2
      NVIC_EN0_R |= 0x40;                                   //priority for
UART1
      //enable UART
      UART1_CTL_R = 0x0301;                           //enable RXE, TXE and
UART
      GPIO_PORTC_PCTL_R = (GPIO_PORTC_PCTL_R&0xFF00FFFF)+0x00220000;
      //write 2
      GPIO_PORTC_AMSEL_R &= ~0x30;  //disable PC4,5 analog funtion
      GPIO_PORTC_AFSEL_R |= 0x30;         //enable PC4,5 alt function
      GPIO_PORTC_DEN_R |= 0x30;                 //enable PC4,5 digital I/O
}



// UART1_OutChar
// send 1-byte info,busy-wait synchronization
void UART1_OutChar(unsigned char data){
      while((UART1_FR_R&0x0020)!=0){}; //wait until TXFF=0
      UART1_DR_R=data;  //write data
}


void UART1_Handler(void){unsigned char read; unsigned char i;
```

```
      Fifo_Init();
      PF2 ^=0x04;                   //toggle heartbeat twice
      PF2 ^=0x04;
      while((UART1_FR_R&0x0010)!=0){}; //wait until RXFE is 0
      for(i=0;i<8;i++){
            read = UART1_DR_R&0xFF;       //get byte information into read
            if(Fifo_Put(read)==0)
                  error++;
      }
      RxCounter++;
      UART1_ICR_R=0x10; //clear RXRIS in the RIS register
      PF2 ^=0x04;
}
```

Ben Fu
Rochelle Roberts

Lab 9 Report

```
//fifo.c
unsigned long static PutI;    //0 to 8, points to empty location to put data
unsigned long static GetI; //0 to 8, points to the oldest data
long static FIFO[9];                      //initialize an 8 element array for
FIFO
void Fifo_Init(void){
      PutI = GetI = 8;                    //empty the queue
}
int   Fifo_Put(char data){
      if(PutI==(GetI+1)%9) return 0; //if the PutI conflicts with GetI
location, FIFO is full, so return a failure
      FIFO[PutI] = data;                  //not full, so put the data into
the address of PutI
      if(PutI==0) PutI = 8;         //wrap PutI if PutI<0
      PutI = (PutI-1);              //if not, just decrement PutI
      return      1;                                          //return a
success
}
int Fifo_Get(long *point){
      if(PutI == GetI) return 0;                //check for empty buffer
      *point = FIFO[GetI];                //get the data from GetI location
      if(GetI==0) GetI = 8;         //wrap GetI if GetI<0
      GetI = GetI-1;          //decrement the GetI pointer
      return      1;                            //return success
}
```