EMEBEDDED SYSTEMS(R22D6810)

LABORATORY MANUAL

M. Tech – I Year – I Sem. (VLSI & Embedded Systems)
2022-2023

DEPARTMENT OF ELECTRONICS AND COMMUNICATIONS ENGG

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

(Autonomous Institution –UGC, Govt. of India)

(Approved by AICTE- Accredited by NBA & NAAC- 'A' Grade-ISO 9001:2008 Certified)
Maisammaguda, Dhulapally, Secundrabad-500 100.

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

(Affiliated to JNTU, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Kompally), Secunderabad – 500100, Telangana State, India.

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

Department Vision & Mission

Vision

To strengthen the department into a centre of academic excellence with focus on advanced technologies & relevant research by delivering the best quality technical education to the students, so as to meet the current and future challenges along with emphasis on moral and ethical values.

Mission

To create and enrich academic environment with essential resources, so as to train and mould students in active learning & critical thinking with innovative ideas, so as to solve real-world problems in the field of Electrical & Electronics Engineering.

To motivate and strengthen the faculty to practice effective teaching & learning process and involve in advanced research & development work.

To enhance industry interaction and initiate best consultancy services.

Quality Policy

To develop, maintain and update global standards of excellence in all our areas of academic & research activities and facilities so as to impart a state of the art & value based technical education to the students commensurate with the rapidly changing industry needs.

To continuously adopt and implement concurrent & commensurate faculty development programmes towards achieving the institution's goals and objectives.

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

(Affiliated to JNTU, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Kompally), Secunderabad – 500100, Telangana State, India.

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

M.TECH – VLSI & EMBEDDED SYSTEMS

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO 1: TECHNICAL ACCOMPLISHMENTS

To pursue career in VLSI and Embedded Systems domain through state of the art learning and self directed approach on cutting edge technologies converging to substantial research work.

PEO 2: **PROFESSIONAL DEVELOPMENT**

To develop managerial skill and apply creative approaches in the domains of VLSI and Embedded Systems by incorporating automation, power consumption, miniaturization, sustainability leading to become a successful professional or an Entrepreneur.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1.

To acquire competency in areas of VLSI and Embedded Systems, Design, Testing, Verification IC Fabrication and prototype development with focus on applications.

PSO 2.

To integrate multiple sub-systems to develop System on Chip, optimize its performance and excel in industry sectors related to VLSI/ Embedded domain and to develop a start-up system.

LAB 2: ES LAB

Note:
The following programs are to be implemented on ARM based Processors/Equivalent.
Minimum of 10 programs are to be conducted.

The following Programs are to be implemented on ARM Processor

1. Simple Assembly Program for a. Addition | Subtraction | Multiplication | Division
b. Operating Modes, System Calls and Interrupts
c. Loops, Branches
2. Write an Assembly programs to configure and control General Purpose Input/Output (GPIO) port pins.
3. Write an Assembly programs to read digital values from external peripherals and execute them with the Target board.
4. Program for reading and writing of a file
5. Program to demonstrate Time delay program using built in Timer / Counter feature on IDE environment
6. Program to demonstrates a simple interrupt handler and setting up a timer
7. Program demonstrates setting up interrupt handlers. Press button to generate an interrupt and trace the program flow with debug terminal.
8. Program to Interface 8 Bit LED and Switch Interface
9. Program to implement Buzzer Interface on IDE environment
10. Program to Displaying a message in a 2 line x 16 Characters LCD display and verify the result in debug terminal.
11. Program to demonstrate I2C Interface on IDE environment
12. Program to demonstrate I2C Interface – Serial EEPROM
13. Demonstration of Serial communication. Transmission from Kit and reception from PC using Serial Port on IDE environment use debug terminal to trace the program.
14. Generation of PWM Signal
15. Program to demonstrate SD-MMC Card Interface.

Course Outcomes:

- To gain the working knowledge of various embedded tools.
- To develop sample programs to be implemented on ARM based Processors or equivalent.
- Create applications for using the ARM Processor on IDE Environment using RAM Tool chain &
- Library.
- Develop applications using the concept of Interfacing.
- Design advanced embedded applications using ARM Processor.

MALLAREDDY COLLEGE OF ENGINEERING AND TECHNOLOGY
MTech (VLSI&EMBEDDED SYSTEMS) EMBEDDED SYSTEMS LABORATORY
LAB PROGRAMS

PART-I

1. Simple Assembly Program for a. Addition | Subtraction | Multiplication | Division

Addition of 4 numbers : Total = A+B+C+D

Program:

```
start
MOV r0, r1; Make the first number the subtotal
ADD r0, r0, r2; Add the second number to the subtotal
ADD r0, r0, r3; Add the third number to the subtotal
ADD r0, r0, r4; Add the fourth number to the subtotal
stop B ; stop
```

SUBTRACTION of 4 numbers : Total = A-B-C-D

Program:

```
start
MOV r0, r1; Make the first number the subtotal
SUB r0, r0, r2; Add the second number to the subtotal
SUB r0, r0, r3; Add the third number to the subtotal
SUB r0, r0, r4; Add the fourth number to the subtotal
stop B ; stop
```

MULTIPLIATION of 4 numbers : Total = A X B X C X D

Program:

```
start
MOV r0, r1; Make the first number the subtotal
MUL r0, r0, r2; Add the second number to the subtotal
MUL r0, r0, r3; Add the third number to the subtotal
MUL r0, r0, r4; Add the fourth number to the subtotal
stop B ; stop
```

DIVISION of 4 numbers : Total = A/B/C/D

Program:

```
start
MOV r0, r1; Make the first number the subtotal
DIV r0, r0, r2; Add the second number to the subtotal
DIV r0, r0, r3; Add the third number to the subtotal
DIV r0, r0, r4; Add the fourth number to the subtotal
stop B ; stop
```

2. Write an Assembly programs to configure and control General Purpose Input/Output (GPIO) port pins.

```
/* Examples Program For "CP-JR ARM7 USB-LPC2148" */
/* Target MCU : Philips ARM7-LPC2148 */
/* : X-TAL : 12.00 MHz */
/* : Run Speed 60.00 MHz (With PLL) */
/* : PLL Setup = M(5),P(2) */
/* : VPB Clock = CPU Clock = 60.00 MHz */
/* Keil Editor : uVision3 V3.03a */
/* Compiler : Keil CARM V2.50a */
/* Function : Example Used Fast GPIO Function */
/*****************************************************/
// Connect P1.24 to LED For Test ON / OFF (Blink)
#include "LPC214x.H" // LPC2148 MPU Register
/* pototype section */
void delay(unsigned long int); // Delay
Time Function
int main(void)
{
// Enable GPIO Function
//SCS |= 0x00000001; //
Enable GPIO0 = Fast GPIO Mode
SCS |= 0x00000002; // Enable
GPIO1 = Fast GPIO Mode
// xxxx xxx1 xxxx xxxx xxxx xxxx xxxx xxxx
6
FIO1MASK = 0xFEFFFFFF; //
Enable GPIO1[24] = Fast GPIO Mode
FIO1DIR = 0x01000000; // Set GPIO-1[24] =
Output
FIO1SET = 0x01000000; // Set GPIO-1[24] Output Pin(OFF LED)
// Loop Test Output GPIO1.24
while(1)
// Loop Continue
{
FIO1CLR = 0x01000000; //
Clear Output GPIO1[24] Pin (ON LED)
```

```
delay(1000000); //
Display Delay
FIO1SET = 0x01000000; // Set Output GPIO1[24] Pin
(OFF LED)
delay(1000000);
// Display Delay
}
}
/************************/
/* Delay Time Function */
/* 1-4294967296 */
/************************/
void delay(unsigned long int count1)
{
while(count1 > 0) {count1--;} // Loop
Decrease Counter

}
```

3. Program to demonstrate Time delay program using built in Timer / Counter feature on IDE environment. (Demonstrate Time delay program using built in timer/Counter feature on IDE environment)

Timer functionality:

```
#include <LPC214X.H>
main()
{
// T0PR = 0x00000003;
PINSEL0=0x00000020;
T0CCR=0x00000004;
T0TCR = 0x00000001;
//T0MCR = 0x00000020;
//T0MR1 = 0X0000000f;
while(1);
}
```

Counter functionality:

```
#include <LPC214X.H>
void main()
{
PINSEL0=0X00000020;
```

```
T0TCR=0X01;
T0CTCR=0X03;
}
```

4. Write a Program to Interface 8 Bit LED and Switch Interface.

```
#include<lpc214x.h>
//////////////////////////// Delay Function ////////////////////////
void main_delay( unsigned int value )
{
unsigned int ui_temp1,ui_temp2; // Delay Variables
for(ui_temp1=0;ui_temp1<value;ui_temp1++) //Delay loop
for(ui_temp2=0;ui_temp2<5000;ui_temp2++); //Delay loop
}
int main()
{
IODIR0 = 0xffffffff;
IODIR1 = 0xffffffff;
while(1)
{
IOSET0= 0xffffffff;
IOSET1= 0xffffffff;
main_delay(5000);
IOCLR0= 0xffffffff;
IOCLR1= 0xffffffff;
main_delay(5000);
}
}
```

5. Program to implement Buzzer Interface on IDE environment

```
#include<lpc214x.h>
#define buzz 0x00000001
#define buzz_on IOCLR0=buzz
#define buzz_off IOSET0=buzz
void delay( unsigned int value )
{
unsigned int ui_temp1,ui_temp2;
for(ui_temp1=0;ui_temp1<value;ui_temp1++)
for(ui_temp2=0;ui_temp2<5000;ui_temp2++);
}
int main()
```

```
{
IODIR0 |=buzz;
while(1)
{
buzz_on;
delay(2000);
buzz_off;
delay(2000);
}
}
```

6. Program to Displaying a message in a 2 line x 16 Characters LCD display and verify the result in debug terminal.

```
#include<lpc214x.h> // LPC2148 MPU Registers
// Define LCD PinIO Mask
#define LCD_RS 0x00010000 // P1.16(0000 0000 0000
000x 0000 0000 0000 0000)
#define LCD_EN 0x00020000 // P1.17(0000 0000 0000
00x0 0000 0000 0000 0000)
#define LCD_D4 0x00040000 // P1.18(0000 0000 0000
0x00 0000 0000 0000 0000)
#define LCD_D5 0x00080000 // P1.19(0000 0000 0000
x000 0000 0000 0000 0000)
#define LCD_D6 0x00100000 // P1.20(0000 0000 000x
0000 0000 0000 0000 0000)
#define LCD_D7 0x00200000 // P1.21(0000 0000 00x0
0000 0000 0000 0000 0000)
//////////////////////////// Delay Function ////////////////////////
void main_delay( unsigned int value )
{
unsigned int ui_temp1,ui_temp2; // Delay Variables
for(ui_temp1=0;ui_temp1<value;ui_temp1++) //Delay loop
for(ui_temp2=0;ui_temp2<5000;ui_temp2++); //Delay loop
}
//////////////////////// LCD Command Sending Function//////////////////////
void lcd_cmd(unsigned char val)
{
unsigned int lcd_ch; // LCD Initial Data
unsigned int lcd_i; // LCD Initial Delay Count
IOCLR1 = LCD_RS ; // RS = 0
```

```
lcd_ch=((val>>4)&0x0F); // Strobe 4-Bit High-Nibble to LCD
IOCLR1 = (LCD_D7|LCD_D6|LCD_D5|LCD_D4); // Reset 4-Bit Pin Data
IOSET1 = (lcd_ch<<18); // Data Send to Respective Pins
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_i=0;lcd_i<10000;lcd_i++); //delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
lcd_ch=(val&0x0F); // Strobe 4-Bit Low-Nibble to LCD
IOCLR1 = (LCD_D7|LCD_D6|LCD_D5|LCD_D4); // Reset 4-Bit Pin Data
IOSET1 = (lcd_ch<<18); //
EN,0,RW,RS:DDDD:0000:0000:0000:0000:0000:0000
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_i=0;lcd_i<10000;lcd_i++); //delay
```



```
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_i=0;lcd_i<10000;lcd_i++); //delay
}
//////////////////////LCD Single Charecter Display
Function//////////////////////////////////////
void lcd_data(unsigned char val)
{
unsigned int lcd_ch; // LCD Initial Data
unsigned int lcd_i; // LCD Initial Delay Count
IOSET1 = LCD_RS ; // RS = 1
lcd_ch=((val>>4)&0x0F); // Strobe 4-Bit High-Nibble to LCD
IOCLR1 = (LCD_D7|LCD_D6|LCD_D5|LCD_D4); // Reset 4-Bit Pin Data
IOSET1 = (lcd_ch<<18); // Data Send to Respective Pins
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_i=0;lcd_i<10000;lcd_i++); // delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
lcd_ch=(val&0x0F); // Strobe 4-Bit Low-Nibble to LCD
IOCLR1 = (LCD_D7|LCD_D6|LCD_D5|LCD_D4); // Reset 4-Bit Pin Data
IOSET1 = (lcd_ch<<18); // Data Send to Respective Pins
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_i=0;lcd_i<10000;lcd_i++); //delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_i=0;lcd_i<10000;lcd_i++); //delay
}
//////////////////// LCD String Display Function ////////////////////
void lcd_puts(unsigned char* str)
{
```

```
while(*str) // Cheack Data is there or not
lcd_data(*str++); //LCD Single Charecter Display Function
}
//////////////// LCD Initialization Function ///////////////////
void lcd_init()
{
unsigned int lcd_main_i; // LCD Initial Delay Count
PINSEL2 |= 0x00000000; // GPIO1[31..16] = I/O Function
IODIR1 |= 0x003F0000 ; // GPIO1[21..16] = OUT Direction
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Power-On Delay (15 mS)
IOCLR1 = ((LCD_D7|LCD_D6|LCD_D5|LCD_D4|LCD_RS|LCD_EN)); // Reset
(RS,EN,4-Bit
Data) Pin
IOSET1 = (LCD_D5|LCD_D4); // Set D4,D5
```



```
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = ((LCD_D7|LCD_D6|LCD_D5|LCD_D4|LCD_RS|LCD_EN)); // Reset
(RS,EN,4-Bit
Data) Pin
IOSET1 = (LCD_D5|LCD_D4); // Set D4,D5
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // delay
IOCLR1 = ((LCD_D7|LCD_D6|LCD_D5|LCD_D4|LCD_RS|LCD_EN)); // Reset
(RS,RW,EN,4-Bit
Data) Pin
IOSET1 = (LCD_D5|LCD_D4); // Set D4,D5
IOSET1 = LCD_EN ; // EN = 1 (Enable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = ((LCD_D7|LCD_D6|LCD_D5|LCD_D4|LCD_RS|LCD_EN)); // Reset
(RS,RW,EN,4-Bit
Data) Pin
IOSET1 = (LCD_D5); // Set D4,D5
IOSET1 = LCD_EN ; // EN = 1 (Enable)
```

```
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
IOCLR1 = LCD_EN ; // EN = 0 (Disable)
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Delay
lcd_cmd(0x28); // LCD 4-Bit Mode
lcd_cmd(0x0c); // LCD Courser Blinking Stop
lcd_cmd(0x06); // LCD Shift Display Right
lcd_cmd(0x01); // LCD Clear Display
for (lcd_main_i=0;lcd_main_i<10000;lcd_main_i++); // Wait Command Ready
}
////////////////////////// MAIN Function ////////////////////
int main()
{
lcd_init(); // Initialization
lcd_puts("LPC2148 LCD DEMO"); //Display Text
lcd_cmd(0xc0); // Courser goto Secong Line
lcd_puts(" Program "); //Display Text
main_delay(5000); //delay
lcd_cmd(0x01); // LCD Clear Display
lcd_puts(" ELEGANT"); //Display Text
lcd_cmd(0xc0); // Courser goto Secong Line
lcd_puts(" Embedded Solu"); //Display Text
while(1); // Continuous Loop
}
```

7. Program to demonstrate I2C Interface – Serial EEPROM

```
#include <LPC214x.H> /* LPC214x definitions */
#include "LCD.h"
#include "i2c.h"
unsigned char buf_size;
unsigned char buf[70],bu[70],COUNT;
int main (void)
{
unsigned char val,val1,val2,val3,val4,val5,val6,val7,val8,val9;
lcd_init();
lcd_cmd(0x01);
lcd_puts("EEPROM WITH ");
lcd_cmd(0xc0);
lcd_puts(" LPC2148");
```

```
i2c_lpc_init(1); // Initialize I2C
i2c_delay(10000);
i2c_delay(10000);
lcd_cmd(0x01);
lcd_puts("DATA WRITING...");
eeprom_write(0,'9'); //WRITING THE DAT TO EEPROM
eeprom_write(1,'3');
eeprom_write(2,'9');
eeprom_write(3,'6');
eeprom_write(4,'6');
eeprom_write(5,'7');
eeprom_write(6,'1');
eeprom_write(7,'5');
eeprom_write(8,'4');
eeprom_write(9,'1');
i2c_delay(10000);
i2c_delay(10000);
lcd_cmd(0x01);
lcd_puts("DATA READING...");
i2c_delay(10000);
i2c_delay(10000);
val=eeprom_read(0); //READING DATA FROM EEPROM
val1=eeprom_read(1);
val2=eeprom_read(2);
val3=eeprom_read(3);
```



```
val4=eeprom_read(4);
val5=eeprom_read(5);
val6=eeprom_read(6);
val7=eeprom_read(7);
val8=eeprom_read(8);
val9=eeprom_read(9);
lcd_cmd(0x01);
lcd_data(val); //DISPLAY THE DATA ON LCD
lcd_data(val1);
lcd_data(val2);
lcd_data(val3);
lcd_data(val4);
lcd_data(val5);
lcd_data(val6);
```

```
lcd_data(val7);
lcd_data(val8);
lcd_data(val9);
}
```

8. Generation of PWM Signal

```
#include <lpc214x.h>
//#define PLOCK 0x00000400
#define PWMPRESCALE 60 //60 PCLK cycles to increment TC by 1 i.e 1 Micro-
second
void initPWM(void);
void main_delay( unsigned int value )
{
unsigned int ui_temp1,ui_temp2;
for(ui_temp1=0;ui_temp1<value;ui_temp1++)
for(ui_temp2=0;ui_temp2<5000;ui_temp2++);
}
int main(void)
{
initPWM(); //Initialize PWM
//IO0DIR = 0x1; This is not needed!
//Also by default all pins are configured as Inputs after MCU Reset.
while(1)
{
PWMMR1 = 1250; //T-ON=25% , Hence 25% Bright
PWMLER = (1<<1); //Update Latch Enable bit for PWMMR1
main_delay(2000);
PWMMR1 = 2500; //50% Bright
PWMLER = (1<<1);
main_delay(2000);
PWMMR1 = 3750; //75% Bright
PWMLER = (1<<1);
main_delay(2000);
PWMMR1 = 5000; //100% Bright
PWMLER = (1<<1);
main_delay(2000);
}
}
```

```
void initPWM(void)
{
/*Assuming that PLL0 has been setup with CCLK = 60Mhz and PCLK also =
60Mhz.*/
/*This is a per the Setup & Init Sequence given in the tutorial*/
PINSEL0 = (1<<1); // Select PWM1 output for Pin0.0
27
PWMPCR = 0x0; //Select Single Edge PWM - by default its single Edged so this
line can be
removed
PWMPR = PWMPRESCALE-1; // 1 micro-second resolution
PWMMR0 = 5000; // 10ms period duration
PWMMR1 = 2500; // 2.5ms - pulse duration i.e width (Brigtness level)
PWMMCR = (1<<1); // Reset PWMTC on PWMMR0 match
PWMLER = (1<<1) | (1<<0); // update MR0 and MR1
PWMPCR = (1<<9); // enable PWM output
PWMTCR = (1<<1) ; //Reset PWM TC & PR
//Now , the final moment - enable everything
PWMTCR = (1<<0) | (1<<3); // enable counters and PWM Mode
//PWM Generation goes active now - LED must be 25% Bright after Reset!!
//Now you can get the PWM output at Pin P0.0!
}
/*
#include<lpc214x.h>
#define PLOCK 0x00000400
#define PWMPRESCALE 60
int main()
{
PINSEL1 =0x00000400; //PWM channel 5 is selected
}*/
```

9. Demonstration of Serial communication. Transmission from Kit and reception from PC using Serial Port on IDE environment use debug terminal to trace the program.

```
#include<lpc214x.h> // LPC2148 MPU Registers
void uart0_init() // Uart0 Initialization Function
{
PINSEL0 =0x00000005; // GPIO1[1,0] = Uart Function // Select P0.0 =
TxD(UART0) // Select
```

```
P0.1 = RxD(UART0)
U0LCR =0X80; // Enable Programming of Divisor Latches
U0DLL =97; // Program Divisor Latch(391) for 9600 Baud
U0LCR =0X03; // Data Bit = 8 Bit
}
void uart0_putch(unsigned char val) //Write character to UART0
{
while(!(U0LSR & 0x20)); // Wait TXD Buffer Empty
U0THR =val; // Write Character
}
unsigned char uart0_getch() //Read character from UART0
{
while(!(U0LSR & 0x01)); // Wait RXD Receive Data Ready
return(U0RBR); // Get Receice Data & Return
}
void uart0_puts( char *stringptr) //String Display Function
{
while (*stringptr) // Cheack Data is there or not
uart0_putch(*stringptr++); //Write character to UART0
}
////////////////////////// MAIN Function ////////////////////
int main()
{
unsigned char i; //variable for recive data from uart buffer
uart0_init(); // Uart0 Initialization Function
uart0_putch(13); //next line command 13,10
uart0_putch(10);
uart0_puts("ELEGANT EMBEDDED SOLUTIONS"); //String Display on
Hyperterminal
uart0_putch(13); //next line command 13,10
uart0_putch(10);
uart0_puts("***** UART0 DEMO PROGRAM *****"); //String Display on
Hyperterminal
uart0_putch(13); //next line command 13,10
uart0_putch(10);
uart0_puts("***** Enter Some Test *****"); //String Display on Hyperterminal
uart0_putch(13); //next line command 13,10
uart0_putch(10);
while(1) // Continuous Loop
{
```

```
i=uart0_getch(); //get data from uart
uart0_putch(i); //print data on Hyperterminal
}
}
```