Sejal patil:

**1.UART0:**

#include<lpc214x.h>

void UART0Init() // INTIALIZE UART0

{

PINSEL0|=0x00000005;

U0FCR=0x07;

U0LCR=0x83;

U0DLM=0X00;

U0DLL=0X62;//BAUD RATE 9600

U0LCR=0x03;

}

unsigned char UART0\_PutChar(unsigned char ch) //TRANSMIT FRAME

{

if(ch=='\n')

{

while(!(U0LSR & 0x20));

U0THR=0x0D;//hex value equivalent to enter on keyboard(to start new line)

}

while(!(U0LSR & 0x20));

return (U0THR=ch);

}

unsigned char UART0\_GetChar(void) // RECEIVE CHARACTER FROM UART0

{

while(!(U0LSR & 0x01));

return (U0RBR);

}

void UART0\_Puts(unsigned char \*str) //DISPLAYING ON SERIAL WINDOW

{

while(1)

{

if(\*str =='\0')

break;

UART0\_PutChar(\*str++);

}

}

2**.UART1**

#include "lpc214x.h"

#include "stdio.h"

#include "string.h"

void UART1Init() // INTIALIZE UART0

{

PINSEL0|=0x00050000;

U1FCR=0x07;

U1LCR=0x83;

U1DLM=0X00;

U1DLL=0X62;//BAUD RATE 9600

U1LCR=0x03;

}

unsigned char UART1\_PutChar(unsigned char ch) //TRANSMIT FRAME

{

if(ch=='\n')

{

while(!(U1LSR & 0x20));

U1THR=0x0D;//hex value equivalent to enter on keyboard(to start new line)

}

while(!(U1LSR & 0x20));

return (U1THR=ch);

}

unsigned char UART1\_GetChar(void) // RECEIVE CHARACTER FROM UART0

{

while(!(U1LSR & 0x01));

return (U1RBR);

}

void UART1\_Puts(unsigned char \*str) //DISPLAYING ON SERIAL WINDOW

{

while(1)

{

if(\*str =='\0')

break;

UART1\_PutChar(\*str++);

}

}

**Serial communication:**

#include<lpc214x.h>

#include "UART0.C"

unsigned char Rx\_byte;

unsigned char SEND\_STRING[]="Welcome to Cummins College \n";

void delay (int n)

{

for(int i=0;i<n;i++)

for(int j=0;j<275;j++);

}

int main(void)

{

UART0Init();

while(1)

{

UART0\_Puts(SEND\_STRING);// TRANSMIT STRING

Rx\_byte= UART0\_GetChar();// RECEIVE SINGLE CHAR

delay(10);

UART0\_PutChar(Rx\_byte);

}

return 0;

}

**GSM:**  
#include<lpc214x.h>

#include "uart0.c"

#include "uart1.c"

unsigned char msg1[]="GSM program started uart intialized\n";

unsigned char AT\_cmd[]="AT\n\r";

unsigned char ATD\_string[]="ATD09868620340;\n\r";

unsigned char ATH\_string[]="ATH0\n\r";

unsigned char cmgf[]="AT+CMGF=1\n\r";

unsigned char cmgs[]="AT+CMGS=\"09868620340\"n\r";

unsigned char cntl\_z= 0x1a;

unsigned char text[]="Namaste";

unsigned char text1[]="\n\r";

unsigned char text2[]="Task Completed";

unsigned char rcv\_byte;

unsigned char rcv\_array[20];

void delay(unsigned int time)

{

unsigned int i,j;

for(i=0; i<time;i++)

for(j=0; j<5000;j++);

}

int main()

{

UART0Init();

UART1Init();

UART0\_Puts(msg1);

UART1\_Puts(AT\_cmd);

UART0\_Puts(AT\_cmd);

delay(1);

UART1\_Puts(ATD\_string);

UART0\_Puts(ATD\_string);

delay(30000);

UART1\_Puts(ATH\_string);

UART0\_Puts(ATH\_string);

delay(3000);

UART1\_Puts(cmgf);

UART0\_Puts(cmgf);

delay(3000);

UART1\_Puts(cmgs);

UART0\_Puts(cmgs);

delay(10000);

UART1\_Puts(text);

UART0\_Puts(text);

delay(3000);

UART1\_PutChar(cntl\_z);

UART0\_PutChar(cntl\_z);

UART1\_Puts(text1);

UART0\_Puts(text1);

UART0\_Puts(text2);

delay(100);

while(1);

return 0;

}

**GPS**

#include<lpc214x.h>

#include "UART0.c "

#include "UART1.c "

unsigned char message\_1[]="GPS program started uart initialised";

unsigned char message\_2[]="\n invalid GPS string";

unsigned char Lat\_string[]="\n Latitude=";

unsigned char Long\_string[]="\n Longitude=";

unsigned char rcv\_byte;

unsigned char rcv\_array[20];

int main()

{

unsigned char count=0;

UART0Init(); //USER INTERFACE

UART1Init(); //GPS

//transmit string on uart0 saying" GPS program started

UART0\_Puts(message\_1);

while (1)

{

// Wait for '$' character to indicate the start of a GPS string

while (rcv\_byte != '$')

{

rcv\_byte = UART1\_GetChar(); // Read a character from UART1

}

UART1\_PutChar(rcv\_byte);

rcv\_byte =UART1\_GetChar();// it will be ‘G’

UART1\_PutChar(rcv\_byte);

rcv\_byte = UART1\_GetChar();// it will be ‘P’

UART1\_PutChar(rcv\_byte);

rcv\_byte = UART1\_GetChar();

UART1\_PutChar(rcv\_byte);

if (rcv\_byte == 'R')// If the character is 'R'

{

for (count = 0; count < 14; count++)

{

rcv\_byte = UART1\_GetChar();

UART1\_PutChar(rcv\_byte);

}

rcv\_byte = UART1\_GetChar();

UART1\_PutChar(rcv\_byte);

if(rcv\_byte == 'V')//Invalid message

{

UART0\_Puts(message\_2);

}

else//valid if 'A'

{

rcv\_byte = UART1\_GetChar();//for comma discarding

UART1\_PutChar(rcv\_byte);

for (count =0; count < 24; count++)//characters stored in array

{

rcv\_byte = UART1\_GetChar();

rcv\_array[count]=rcv\_byte;

UART1\_PutChar(rcv\_byte);

}

UART0\_Puts(Lat\_string);

// Print the first 11 characters of rcv\_array through UART0

for (count = 0; count < 11; count++)

{

UART0\_PutChar(rcv\_array[count]);

}

UART0\_Puts(Long\_string);

// Print the first 11 characters of rcv\_array through UART0

for (count = 12; count < 24; count++)

{

UART0\_PutChar(rcv\_array[count]);

}

}

}

}

}

**lm35**

#include<lpc214x.h>

#include "uart0.c"

unsigned int data,adc\_data,count;

unsigned char data\_rcv[3];

unsigned char result;

unsigned char message\_1[]="\n\rADC data in HEX = ";

unsigned char crlf\_1[]="\n\r";

void adcdelay(unsigned int time)

{

unsigned int i,j;

for(i=0;i<time;i++)

{

for(j=0;j<10000;j++);

}

}

void Init\_ADC(void)

{

PINSEL1 |=0x01000000;

AD0CR=0x00200302;

}

unsigned int Read\_ADC\_1(void)

{

AD0CR=0x01200302;

while(!(AD0DR1 & 0x80000000));

data=((AD0DR1>>6)&0x3FF);

return data;

}

void hex\_ascii\_transmit()

{

for (count=0; count<3; count++)

{

if (data\_rcv[count]< 10)

{

result = data\_rcv[count]+ 0x30;

UART0\_PutChar(result);

}

else

{

result = data\_rcv[count]+ 0x37;

UART0\_PutChar(result);

}

}

UART0\_Puts(crlf\_1 );

}

void nibble\_sep(void)

{

data\_rcv[2] = (adc\_data & 0x0000000F); //lower

data\_rcv[1] = (adc\_data & 0x000000F0)>>4;//middle

data\_rcv[0] = (adc\_data & 0x00000F00)>>8;//upper

}

int main(void)

{

Init\_ADC();

UART0Init();

while(1)

{

adc\_data = Read\_ADC\_1();

adcdelay(1000);

nibble\_sep();

UART0\_Puts(message\_1 );

hex\_ascii\_transmit();

}

}

**DAC:  
TRIANGULAR**

#include<lpc214x.h>

#define DAC\_PinMask 1<<19

#define DAC\_DataMask 0x0000FFC0

#define Datashift 6

void delay(unsigned int n)

{

for(int i=0;i<n;i++)

{

for(int j=0;j<1000;j++)

{

}

}

}

void DAC\_init()

{

PINSEL1|=DAC\_PinMask;

DACR=1<<16;

}

int main()

{

unsigned int i;

DAC\_init();

while(1)

{

for(i=0;i<1023;i++){

DACR=(DAC\_DataMask & (i<<Datashift));

}

for(i=1023;i>0;i--){

DACR=(DAC\_DataMask & (i<<Datashift));

}

}

return 0;

}

**Square wave:**

#include <lpc214x.h>

#define DACPinMask 1<<19 // Mask for DAC pin (P0.19)

#define datamask 0x0000FFC0 // Data mask for DAC value

#define datashift 6 // Bit shift value for DAC data

// Function to create a delay using nested loops.

void delay(unsigned int n){

for(int i=0; i<n; i++){

for(int j=0; j<256; j++){

// Delay loop for generating time intervals.

}

}

}

void DACinit(){

PINSEL1 |= DACPinMask; // Set P0.19 as DAC output

DACR = 1 << 16; // Enable DAC output

}

int main(){

DACinit(); // Initialize DAC

while(1){

DACR = 0x00000000; // Set DAC data to 0 (0V)

delay(5); // Delay for a specified time

DACR = 0x0000FFC0; // Set DAC data to maximum (3.3V)

delay(5); // Delay for a specified time

}

}

**Stepcase :**

#include <lpc214x.h>

#define mask 0x0000FFC0

#define shift 6

void DAC\_INIT()

{

PINSEL1 |= 1<<19;

DACR = 1<<16;

}

void delay(unsigned int t)

{

unsigned int i,j;

for(i=0;i<275;i++){

for(j=0;j<t;j++){

}

}

}

void main()

{

unsigned int i;

DAC\_INIT();

while(1)

{

for(i=0;i<1024;i=i+341)

{

DACR = (i<<shift) & mask;

delay(10);

}

for(i=682;i>0;i=i-341)

{

DACR = (i<<shift) & mask;

delay(10);

}

}

}

**Sawtooth wave**:

#define shift 6

#define mask 0x0000FFC0

#include<LPC214x.h>

void dac\_init()

{

PINSEL1=1<<19;

DACR=1<<16;

}

int main()

{

dac\_init();

while(1)

{

for(int i=0;i<1024;i++)

{

DACR=(i<<shift)&mask;

}

}

return 0;

}

**glcd**

#include "lpc214x.h"

// Define constants for GPIO pins and control signals

#define LCD\_PORT 0x00FF0000

#define EN (1 << 31)

#define RS (1 << 27)

#define CS1 (1 << 26)

#define CS2 (1 << 25)

#define GRST (1 << 30)

#define RW (1 << 24)

#define LCD\_SHIFT 16

// Function to introduce a small delay

void delay(unsigned int time) {

int i, j;

for (i = 0; i < time; i++)

for (j = 0; j < 10; j++);

}

// Function to introduce a longer delay

void ldelay(unsigned int time) {

int i, j;

for (i = 0; i < time; i++)

for (j = 0; j < 5000; j++);

}

// Function to generate an ENABLE pulse for data latching

void LCD\_strobe(void) {

IO1SET = EN; // Set the EN (Enable) pin high

delay(5); // Delay to keep EN high

IO1CLR = EN; // Clear the EN (Enable) pin to create a pulse

delay(5); // Delay after the pulse

}

// Function to send data to the GLCD

void GLCD\_data(unsigned char ch) {

IO1CLR = LCD\_PORT; // Clear the data pins

IO1SET = ch << LCD\_SHIFT; // Set the data pins with the given data

IO1SET = RS; // Set RS (Register Select) to indicate data

LCD\_strobe(); // Call function to latch the data

}

// Function to send a command to the GLCD

void GLCD\_cmd(unsigned char ch) {

IO1CLR = LCD\_PORT; // Clear the data pins

IO1SET = ch << LCD\_SHIFT; // Set the data pins with the given command

IO1CLR = RS; // Clear RS (Register Select) to indicate a command

LCD\_strobe(); // Call function to send the command

}

// Function to initialize the GLCD

void GLCD\_Init() {

int i;

PINSEL0 = 0; // Set pins as GPIO

PINSEL1 = 0;

PINSEL2 = 0;

IODIR1 = LCD\_PORT | RS | EN | CS1 | CS2 | GRST | RW; // Set pins as output

IOSET1 = GRST | CS1 | CS2; // Set control pins

IOCLR1 = RW | RS | EN; // Clear other control pins

for (i = 0; i < 10; i++)

GLCD\_cmd(0x3F); // Display ON

GLCD\_cmd(0x40); // Set Y address as 0 (range 0-63)

GLCD\_cmd(0xB8); // Set X address as 0 (page address) (range 0-7)

}

// Function to display data on the GLCD

void GLCD\_disp(unsigned char \*temp1) {

int page, col;

for (page = 0; page < 8; page++) {

IO1SET = CS1; // Select chip CS1

IO1CLR = CS2; // Deselect chip CS2

GLCD\_cmd(0xB8 | page); // Set the page address

GLCD\_cmd(0x40); // Set the column address

for (col = 0; col < 64; col++) {

GLCD\_data(temp1[(page \* 128) + col]); // Send data for CS1

}

IO1CLR = CS1; // Deselect chip CS1

IO1SET = CS2; // Select chip CS2

GLCD\_cmd(0xB8 | page); // Set the page address

GLCD\_cmd(0x40); // Set the column address

for (col = 64; col < 128; col++) {

GLCD\_data(temp1[(page \* 128) + col]); // Send data for CS2

}

}

}

// Main function

int main() {

GLCD\_Init(); // Initialize the GLCD

int i, j, l, m;

// Draw vertical strips on the display

for (i = 1; i < 7; i++) {

IO1SET = CS1; // Select chip CS1

IO1CLR = CS2; // Deselect chip CS2

GLCD\_cmd(0xB8 | i);

for (j = 56; j < 64; j++) {

GLCD\_cmd(0x40 | j);

GLCD\_data(0x00); // Set pixel to 0x00

}

}

// Draw horizontal strips on the display

for (l = 3; l < 5; l++) {

GLCD\_cmd(0xB8 | l);

for (m = 37; m < 64; m++) {

GLCD\_cmd(0x40 | m);

GLCD\_data(0x00); // Set pixel to 0x00

}

}

// Repeat the same process for the other half of the display (CS2)

for (i = 1; i < 7; i++) {

IO1SET = CS2; // Select chip CS2

IO1CLR = CS1; // Deselect chip CS1

GLCD\_cmd(0xB8 | i);

for (j = 64; j < 72; j++) {

GLCD\_cmd(0x40 | j);

GLCD\_data(0x00); // Set pixel to 0x00

}

}

for (l = 3; l < 5; l++) {

GLCD\_cmd(0xB8 | l);

for (m = 64; m < 90; m++) {

GLCD\_cmd(0x40 | m);

GLCD\_data(0x00); // Set pixel to 0x00

}

}

}

**timer**

#include <lpc214x.h>

void delay(unsigned int milliseconds){

T0PR = 14999;

T0TC =0;

T0TCR = 0x01;

while(T0TC < milliseconds);

T0TCR = 0x00;

}

int main(){

PINSEL2 = 0x00000000;

IODIR1 = 0xFF000000;

while(1){

IOSET1 = 0xFF000000;

delay(1000);

IOCLR1 = 0xFF000000;

delay(1000);

}

}