```c
#include "LPC214x.h"
#include "lcd.h"
#include "delay.h"

#define PCLK 12000000


void ConfigI2c0(int BaudRate);
void WriteEeprom0(char SlaveAddress, unsigned char *Data,char len);
void ReadEeprom0(char SlaveAddress, unsigned char *Data, char len);

char WriteDataI2c0(char SlaveAddress, unsigned char *Data,unsigned char len);

char ReadDataI2c0(char SlaveAddress, unsigned char *Data, char len);

unsigned char addr= 0x00;

int main (void)
{
    unsigned char EepromBufA[]={0x00,'I','N','D','I','A'};
    unsigned char EepromBufB[2]={0x00,0};
    unsigned int No;

    edutechlcdinit();
    ConfigI2c0(100);

    WriteEeprom0(0xA0,EepromBufA,6);
    edutechlcdstring("Write Success",1,0);

    for(No=0; No<6; No++)
    {
    ReadEeprom0(0xA0,EepromBufB,1);
    edutechlcdstring(&EepromBufB[0],2,No);
    DelayMs(100);                       // This delay is important
    addr++;
    }
    while(1);
    return 0;
}


void ConfigI2c0(int BaudRate)
{

      PINSEL0 = 0x050;
      I2C0CONCLR = 0x6c; // disable I2C
      I2C0CONSET = 0x40; // Enable I2C

      I2C0SCLH = (PCLK/(2*(BaudRate*1000)));
      I2C0SCLL = (PCLK/(2*(BaudRate*1000)));
}


unsigned char I2C_WaitStatus0(unsigned char I2CStatus)
{
```

```
	unsigned int Time,I2C_WAIT_TIME_OUT;
	I2C_WAIT_TIME_OUT = 200;
	Time = 0;

		while (Time++ < I2C_WAIT_TIME_OUT)
		{
			if (I2C0CONSET & 0x08) // poll SI bit for Comm complete
			{
				if (I2C0STAT == I2CStatus)// read I2C status value
				{
					Time = 0;
					return 1;
				}
			}
		}
		return 0;
}


char WriteDataI2c0(char SlaveAddress, unsigned char *Data,unsigned char len)
{

	I2C0CONCLR = 0x2c; // Clear all acknowledgment
	I2C0CONSET = 0x40; // Enable I2C
	I2C0CONSET |= 0x20;	//Assert Start condition I2C0CONSET = 0x60

	if (!I2C_WaitStatus0(0x08)) // 0x08: status code to check if start condition
has been transmitted
	return 0;

	I2C0DAT = SlaveAddress;
	I2C0CONCLR = 0x2C;		// Clear all acknowledgment
	if (!I2C_WaitStatus0(0x18)) // status code to check if SLA+W has been
transmitted and acknowledgment has been received
	return 0;

	I2C0DAT = *Data++;
	I2C0CONCLR = 0x2C;                                                  // Clear all
acknowledgment
	if (!I2C_WaitStatus0(0x28)) // status code to check if data byte in I2C0DAT
has been transmitted and acknowledgment has been received
	return 0;

	while(len)
	{
	I2C0CONCLR=0x2C;

	if (!I2C_WaitStatus0(0x28)) // status code to check if data byte in I2C0DAT
has been transmitted and acknowledgment has been received
	return 0;

	I2C0DAT = *Data++ ;

	if(len>1)
```

```
	I2C0CONSET = 0x04 | 0x40;		//assert acknowledge flag and keep I2C
enabled
	else
	I2C0CONSET = (0x10 | 0x40);			//Send Stop condition

	len--;
	}

	I2C0CONSET = (0x10 | 0x40); // Send Stop condition
	I2C0CONCLR=0x2C;

	return(1);
}


char ReadDataI2c0(char SlaveAddress, unsigned char *Data, char len)
{

	I2C0CONCLR = 0x2c; // Clear Acknowledgment
	I2C0CONSET = 0x40; // Enable I2C
	I2C0CONSET |= 0x20;			//Assert Start Condition

	if (!I2C_WaitStatus0(0x08)) // 0x08: status code to check if start condition
has been transmitted
	return 0;

	I2C0DAT = SlaveAddress;
	I2C0CONCLR = 0x28;
	if (!I2C_WaitStatus0(0x40)) // status code to check if SLA+R has been
transmitted and acknowledgment has been received
	return 0;

	while(len)
	{
	I2C0CONCLR=0x2C;

	if (!I2C_WaitStatus0(0x58)) // status code to check if data byte has been
received and ACK has been returned
	return 0;

	*Data++ = I2C0DAT;

	if(len>1)
	I2C0CONSET = 0x04 | 0x40;		//keep I2C enabled and assert
acknowledge flag
	else
	I2C0CONSET = (0x10 | 0x40);			// Send Stop condition

	len--;
	}

	I2C0CONSET = (0x10 | 0x40); // Send Stop condition
	I2C0CONCLR=0x2C;

	return(1);
```

```
}


void WriteEeprom0(char SlaveAddress, unsigned char *Data,char len)
{
    WriteDataI2c0(SlaveAddress,Data,len);
}

void ReadEeprom0(char SlaveAddress, unsigned char *Data, char len)
{
       WriteDataI2c0(SlaveAddress,&addr,0);
       ReadDataI2c0((SlaveAddress + 0x01),Data,len);
}
```