#define F\_CPU 16000000UL

#include <avr/io.h>

#include <string.h>

#include "USART\_RS232\_H\_file.h" /\* add USART library \*/

#include "lcd\_16.h" /\* add LDC16x2 library \*/

#include "uart.h"

#include <util/delay.h>

int main()

{

int i,k=0,j=0,m=0,l=0,n=0,s=0,R=5;

unsigned char a[12]="2700130F5962",b[12]="270013125E78",c[12]="27001312CEE8",d[12]="27001310A98D",e[12]="270013111035",RFID[12];

char p[10];

USART\_Init(115200); /\* initialize USART with 9600 baud rate \*/

lcd\_init(LCD\_DISP\_ON); /\* initialize LCD16x2 display \*/

lcd\_clrscr();

lcd\_gotoxy(0,0); /\* Set row and column position at 0,0 \*/

memset(RFID,0,15);

lcd\_puts("items:");

while(1)

{ s++;

j=0,k=0,l=0,m=0,n=0;

for (i=0;i<12;i++)

{

RFID[i]=USART\_RxChar();

if(RFID[i]==a[i])

{

j++;

}

if(RFID[i]==b[i])

{

k++;

}

if(RFID[i]==c[i])

{

l++;

}

if(RFID[i]==d[i])

{

m++;

}

if(RFID[i]==e[i])

{

n++;

}

}

if(s==1)

{

lcd\_gotoxy(0,1);

lcd\_puts("RS 30 MAGGI");

}

if(s==2)

{

lcd\_gotoxy(0,1);

lcd\_puts("RS =40 BRUSH");

}

if(s==3)

{

lcd\_gotoxy(0,1);

lcd\_puts("RS=100 BOOKS");

}

if(s==4)

{

lcd\_gotoxy(0,1);

lcd\_puts("RS=500 SHOES");

}

if(s==5)

{

lcd\_gotoxy(0,1);

lcd\_puts("RS =30 DIARY");

}

if(s<=5)

{

itoa(s,p,10);

lcd\_gotoxy(6,0);

lcd\_puts(p);

}

unsigned char digital;

uart\_init();

\_delay\_ms(100);

uart\_string("AT+CWMODE=1\r\n");

\_delay\_ms(300);

uart\_string("AT+CWJAP=\"ni\",\"abc12345\"\r\n");

\_delay\_ms(500);

\_delay\_ms(500);

\_delay\_ms(500);

digital=s; // pin no. // digital is a variable

uart\_string("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n"); // this for connected thingspeak network

\_delay\_ms(300); // delay for n/w conncted

uart\_string("AT+CIPSEND=51\r\n"); // 51 byte of maximum data send

\_delay\_ms(100); // wait

uart\_string("GET /update?api\_key=MYPPNEI3V4TTKT2E="); // particular address of my channel

uart\_num(digital);

uart\_string("\r\n"); // new line

\_delay\_ms(100); // wait

uart\_string("AT+CIPCLOSE\r\n"); // connection close

\_delay\_ms(600); // wait

}

}