TEST CASES

# define F\_CPU 8000000ul

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

/\*\*MACROS/

#define USART\_BAUDRATE 9600

#define BAUD\_PRESCALE (((F\_CPU / (USART\_BAUDRATE \* 16UL))) - 1)

#define uchar unsigned char

#define uint unsigned int

#define LCDPORTDIR DDRB

#define LCDPORT PORTB

#define rs 0

#define rw 1

#define en 2

#define RSLow (LCDPORT&=~(1<<rs))

#define RSHigh (LCDPORT|=(1<<rs))

#define RWLow (LCDPORT&=~(1<<rw))

#define ENLow (LCDPORT&=~(1<<en))

#define ENHigh (LCDPORT|=(1<<en))

#define KeyPORTdir DDRA

#define key PINA

#define KeyPORT PORTA

After this, we have declared some variables and arrays for fingerprint command and response. We have also added some functions for fetching and setting data to RTC.

void RTC\_stp()

{

TWCR=(1<<TWINT)|(1<<TWEN)|(1<<TWSTO); //stop communication

}

void RTC\_read()

{

TWCR=(1<<TWINT)|(1<<TWSTA)|(1<<TWEN);

while((TWCR&0x80)==0x00);

TWDR=0xD0; //RTC write (slave address)

TWCR=(1<<TWINT)|(1<<TWEN);

while(!(TWCR&(1<<TWINT)));

TWDR=0x00; //RTC write (word address)

TWCR=(1<<TWINT)|(1<<TWEN);

while(!(TWCR&(1<<TWINT)));

TWCR=(1<<TWINT)|(1<<TWSTA)|(1<<TWEN); //start RTC communication again

while ((TWCR&0x80)==0x00);

TWDR=0xD1; // RTC command to read

TWCR=(1<<TWINT)|(1<<TWEN);

while(!(TWCR&(1<<TWINT)));

}

Then we have some functions for LCD which are responsible to drive the LCD. LCD driver function is written for 4-bit mode drive. Followed by that we also have some UART driver functions which are responsible for initializing UART and exchanging data between fingerprint sensor and microcontroller.

void serialbegin()

{

UCSRC = (1 << URSEL) | (1 << UCSZ0) | (1 << UCSZ1);

UBRRH = (BAUD\_PRESCALE >> 8);

UBRRL = BAUD\_PRESCALE;

UCSRB=(1<<RXEN)|(1<<TXEN)|(1<<RXCIE);

sei();

}

ISR(USART\_RXC\_vect)

{

char ch=UDR;

buf[ind++]=ch;

if(ind>0)

flag=1;

//serial1Write(ch);

}

void serialwrite(char ch)

{

while ((UCSRA & (1 << UDRE)) == 0);

UDR = ch;

}

void serialprint(char \*str)

{

while(\*str)

{

serialwrite(\*str++);

}

}

Now we have some more UART function but they are software UART. It is used for transferring saved data to the computer via serial terminal. These functions are delay-based and don’t use any type of interrupt. And for UART only tx signal will work and we have hardcoded baud rate for soft UART as 9600.

void SerialSoftWrite(char ch)

{

PORTD&=~(1<<7);

\_delay\_us(104);

for(int i=0;i<8;i++)

{

if(ch & 1)

PORTD|=(1<<7);

else

PORTD&=~(1<<7);

\_delay\_us(104);

ch>>=1;

}

PORTD|=(1<<7);

\_delay\_us(104);

}

void SerialSoftPrint(char \*str)

{

while(\*str)

{

SerialSoftWrite(\*str);

str++;

}

}

Followed by that we have functions that are responsible for displaying the RTC time in the LCD. The below given functions are used for writing attendance data to EEPROM and reading attendance data from EEPROM.

int eeprom\_write(unsigned int add,unsigned char data)

{

while(EECR&(1<<EEWE));

EEAR=add;

EEDR=data;

EECR|=(1<<EEMWE);

EECR|=(1<<EEWE);

return 0;

}

char eeprom\_read(unsigned int add)

{

while(EECR & (1<<EEWE));

EEAR=add;

EECR|=(1<<EERE);

return EEDR;

}

[1:15 PM, 4/24/2022] Sindhu M T: The below function is responsible for reading fingerprint image and convert them in template and matching with already stored image and show result over LCD.

void matchFinger()

{

// lcdwrite(1,CMD);

// lcdprint("Place Finger");

// lcdwrite(192,CMD);

// \_delay\_ms(2000);

if(!sendcmd2fp((char \*)&f\_detect[0],sizeof(f\_detect)))

{

if(!sendcmd2fp((char \*)&f\_imz2ch1[0],sizeof(f\_imz2ch1)))

{

if(!sendcmd2fp((char \*)&f\_search[0],sizeof(f\_search)))

{

LEDHigh;

buzzer(200);

uint id= data[0];

id<<=8;

id+=data[1];

uint score=data[2];

score<<=8;

score+=data[3];

(void)sprintf((char \*)buf1,"Id: %d",(int)id);

lcdwrite(1,CMD);

lcdprint((char \*)buf1);

saveData(id);

\_delay\_ms(1000);

lcdwrite(1,CMD);

lcdprint("Attendance");

lcdwrite(192,CMD);

lcdprint("Registered");

\_delay\_ms(2000);

LEDLow;

}

Followed by that we have a function that is used for enrolling a new finger and displaying the result or status on LCD. Then the below function is used for deleting stored fingerprint from the module by using id number and show status of the same.

void deleteFinger()

{

id=getId();

f\_delete[10]=id>>8 & 0xff;

f\_delete[11]=id & 0xff;

f\_delete[14]=(21+id)>>8 & 0xff;

f\_delete[15]=(21+id) & 0xff;

if(!sendcmd2fp(&f\_delete[0],sizeof(f\_delete)))

{

lcdwrite(1,CMD);

sprintf((char \*)buf1,"Finger ID %d ",id);

lcdprint((char \*)buf1);

lcdwrite(192, CMD);

lcdprint("Deleted Success");

}

else

{

lcdwrite(1,CMD);

lcdprint("Error");

}

\_delay\_ms(2000);

}

Below function is responsible for sending attendance data to serial terminal via soft UART pin PD7 and TTL to USB converter.

/function to show attendence data on serial moinitor using softserial pin PD7/

void ShowAttendance()

{

char buf[128];

lcdwrite(1,CMD);

lcdprint("Downloding....");

SerialSoftPrintln("Attendance Record");

SerialSoftPrintln(" ");

SerialSoftPrintln("S.No ID1 ID2 Id3 ID4 ID5 ");

//serialprintln("Attendance Record");

//serialprintln(" ");

//serialprintln("S.No ID1 ID2 Id3 ID4 ID5");

for(int cIndex=1;cIndex<=8;cIndex++)

{

sprintf((char \*)buf,"%d "

"%d:%d:%d %d/%d/20%d "

"%d:%d:%d %d/%d/20%d "

"%d:%d:%d %d/%d/20%d "

"%d:%d:%d %d/%d/20%d "

"%d:%d:%d %d/%d/20%d ",

cIndex,

eeprom\_read((cIndex\*6)),eeprom\_read((cIndex\*6)+1),eeprom\_read((cIndex\*6)+2),eeprom\_read((cIndex\*6)+3),eeprom\_read((cIndex\*6)+4),eeprom\_read((cIndex\*6)+5),

eeprom\_read((cIndex\*6)+48),eeprom\_read((cIndex\*6)+1+48),eeprom\_read((cIndex\*6)+2+48),eeprom\_read((cIndex\*6)+3+48),eeprom\_read((cIndex\*6)+4+48),eeprom\_read((cIndex\*6)+5+48),

eeprom\_read((cIndex\*6)+96),eeprom\_read((cIndex\*6)+1+96),eeprom\_read((cIndex\*6)+2+96),eeprom\_read((cIndex\*6)+3+96),eeprom\_read((cIndex\*6)+4+96),eeprom\_read((cIndex\*6)+5+96),

eeprom\_read((cIndex\*6)+144),eeprom\_read((cIndex\*6)+1+144),eeprom\_read((cIndex\*6)+2+144),eeprom\_read((cIndex\*6)+3+144),eeprom\_read((cIndex\*6)+4+144),eeprom\_read((cIndex\*6)+5+144),

eeprom\_read((cIndex\*6)+192),eeprom\_read((cIndex\*6)+1+192),eeprom\_read((cIndex\*6)+2+192),eeprom\_read((cIndex\*6)+3+192),eeprom\_read((cIndex\*6)+4+192),eeprom\_read((cIndex\*6)+5+192));

SerialSoftPrintln(buf);

//serialprintln(buf);

}

lcdwrite(192,CMD);

lcdprint("Done");

\_delay\_ms(2000);

}

Below function is used for deleting all the attendance data from the microcontroller’s EEPROM.

void DeleteRecord()

{

lcdwrite(1,CMD);

lcdprint("Please Wait...");

for(int i=0;i<255;i++)

eeprom\_write(i,10);

\_delay\_ms(2000);

lcdwrite(1,CMD);

lcdprint("Record Deleted");

lcdwrite(192,CMD);

lcdprint("Successfully");

\_delay\_ms(2000);

}

In the main function we will initialize all the used module and gpio pins. Finally, all-controlling event are performed in this as shown in below

[1:18 PM, 4/24/2022] Sindhu M T: while(1)

{

RTC();

// if(match == LOW)

// {

matchFinger();

// }

if(enrol == LOW)

{

buzzer(200);

enrolFinger();

\_delay\_ms(2000);

// lcdinst();

}

else if(delet == LOW)

{

buzzer(200);

getId();

deleteFinger();

\_delay\_ms(1000);

}

}

return 0;

}

The complete working set-up is shown in the video linked below. Hope you enjoyed the project and learnt something new. If you have any questions leave them in the comment section or use the forums for other technical questions.