A close up of text

Description automatically generated

ΑΘΗΝΑ 22 Νοεμβρίου 2024

**8η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ**

**ΓΙΑ ΤΟ ΜΑΘΗΜΑ “Εργαστήριο Μικροϋπολογιστών”**

**ΟΜΑΔΑ 23**

**Συνεργάτες**

Νικόλαος Αναγνώστου Νικόλαος Λάππας

03121818 03121098

**Ζήτημα 8.1:** Στην συγκεκριμένη άσκηση καλούμαστε να υλοποιήσουμε κώδικα σε C, ο οποίος να στέλνει μηνύματα στην USART και θα συνδέεται μέσω του ESP8266 στο δίκτυο του εργαστηρίου. Το πρόγραμμα περιέχει τις συναρτήσεις για την αρχικοποίηση της usart επικοινωνίας, της LCD με χρήση του PCA9555 καθώς και τρεις επιπλέον με τις οποίες κάνουμε τα εξής:

* Με την στέλνουμε ένα buffer που περιέχει την πληροφορία που θέλουμε να στείλουμε (παραπάνω από 1 byte συνήθως) και η συνάρτηση καλεί για κάθε χαρακτήρα του buffer την συνάρτηση προκειμένου να στείλουμε την πληροφορία ανά byte.
* Με την κάνουμε το δυαδικό ανάλογο με πριν, δεχόμαστε, δηλαδή, την πληροφορία που μας στέλνει ο server μέσω της συνάρτησης και την αποθηκεύουμε byte προς byte σε έναν δικό μας buffer προκειμένου να είμαστε σε θέση να το διαβάσουμε και να το τυπώσουμε. Έχουμε κάνει τις απαραίτητες ρυθμίσεις ώστε να αγνοούμε τον χαρακτήρα ‘ ” ’ και κάθε μήνυμα να λήγει με τον χαρακτήρα ‘ \n ’.
* Τέλος, με την στέλνουμε όλο το μήνυμα που θέλουμε να τυπώσουμε σε μορφή string, και αυτή καλεί την και τυπώνει στην οθόνη έναν προς έναν τους χαρακτήρες.

Έτσι, στην κάνουμε δυο ελέγχους μέσα σε επαναλαμβανόμενο βρόχο (μέχρι να επαληθευτεί με μήνυμα Success). Αρχικά, θέλουμε να συνδεθούμε στον server και στέλνουμε την εντολή . Περιμένουμε για την απάντηση από την server και αν αυτή διαφέρει από το τυπώνουμε αποτυχία και επαναλαμβάνουμε. Όταν στείλει επιτυχία, στέλνουμε την εντολή προκειμένου να θέσουμε το url στην κατάλληλη διεύθυνση και αναμένουμε το μήνυμα επιτυχίας ή αποτυχίας. Τα μηνύματα αυτά τα εκτυπώνουμε στην LCD οθόνη. Ακολουθεί ο κώδικας με τις νέες μόνο συναρτήσεις και την main():

#define F\_CPU 16000000UL

#include<avr/io.h>

#include<avr/interrupt.h>

#include<util/delay.h>

#include<stdbool.h>

#include<stdio.h>

#include<string.h>

//-------------------------------- USART ---------------------------------------

/\* Routine: usart\_init

Description: This routine initializes the usart as shown below.

------- INITIALIZATIONS -------

Baud rate: 9600 (Fck= 8MH)

Asynchronous mode

Transmitter on

Reciever on

Communication parameters: 8 Data ,1 Stop, no Parity

--------------------------------

parameters: ubrr to control the BAUD.

return value: None.

\*/

void usart\_init(unsigned int ubrr) {

UCSR0A=0;

UCSR0B=(1<<RXEN0)|(1<<TXEN0); // enable receiving and transmitting

UBRR0H=(unsigned char)(ubrr>>8);

UBRR0L=(unsigned char)ubrr;

UCSR0C=(3 << UCSZ00); // configure the frame size to 8 data

return;

}

/\* Routine: usart\_transmit

Description: This routine sends a byte of data using usart.

parameters:

data: the byte to be transmitted

return value: None.

\*/

void usart\_transmit(uint8\_t data) {

while(!(UCSR0A&(1<<UDRE0)));

UDR0=data;

}

/\* Routine: usart\_receive

Description: This routine receives a byte of data from usart.

parameters: None.

return value: the received byte

\*/

uint8\_t usart\_receive() {

while(!(UCSR0A&(1<<RXC0)));

return UDR0;

}

void transmit\_command(char \*data) {

int i = 0;

while (data[i] != '\0') {

usart\_transmit(data[i]);

i++;

}

}

void receive\_response(char \*response) {

char input;

int i = 0;

while(1){

input = usart\_receive();

if(input == '\n') {

response[i] = input;

i++;

break;

} else {

if (input == '"');

else {

response[i] = input;

i++;

}

}

}

for(i; i<=9; i++){

response[i] = " ";

}

}

void lcd\_print(const char\* str) {

while(\*str) {

lcd\_data(\*str++); // Send each character to the LCD

}

}

#define SIZE 10

int main() {

DDRC = 0x00;

twi\_init();

PCA9555\_0\_write(REG\_CONFIGURATION\_0, 0x00); // EXT\_PORT0 -> output

usart\_init(103); // for baud rate 9600

lcd\_init();

lcd\_clear\_display();

// Waiting till ESP connects ...

while(1) {

transmit\_command("ESP:connect\n");

char answer[SIZE];

receive\_response(answer);

if (strcmp(answer, "Success\n") == 0) {

lcd\_print("1.Success");

\_delay\_ms(2000);

lcd\_clear\_display();

break;

}

else lcd\_print("1.Fail");

\_delay\_ms(2000);

lcd\_clear\_display();

}

// Sending command for url ...

while(1) {

transmit\_command("ESP:url:\"http://192.168.1.250:5000/data\n\"");

char answer\_no2[SIZE];

receive\_response(answer\_no2);

if (strcmp(answer\_no2, "Success\n") == 0) {

lcd\_print("2.Success");

\_delay\_ms(2000);

lcd\_clear\_display();

break;

}

else lcd\_print("2.Fail");

\_delay\_ms(2000);

lcd\_clear\_display();

}

}

**Ζήτημα 8.2:** Στο συγκεκριμένο ερώτημα επεκτείναμε την παραπάνω υλοποίηση προκειμένου να παίρνουμε (προσομοίωση) τιμές πίεσης και θερμοκρασίας με χρήση του POT1 και του αισθητήρα DS18B20 αντίστοιχα. Επιπλέον, προστέθηκε η αλληλεπίδραση με το keypad προκειμένου να ανανεώνεται με την χρήση του το status σε ‘NURSE CALL’. Για τον σκοπό αυτό προσθέσαμε τις συναρτήσεις και την λογική από προηγούμενες εργαστηριακές εργασίες που είχαμε υλοποιήσει. Ακολουθεί ο κώδικας με την main().

#define SIZE\_ANSWER 10

#define SIZE\_STATUS 15

#define OFFSET 13

int main() {

DDRB = 0xFF;

DDRC = 0x00;

twi\_init();

PCA9555\_0\_write(REG\_CONFIGURATION\_0, 0x00); // EXT\_PORT0 -> output

usart\_init(103); // for baud rate 9600

lcd\_init();

lcd\_clear\_display();

// Waiting till ESP connects ...

while(1) {

transmit\_command("ESP:connect\n");

char answer[SIZE\_ANSWER];

receive\_response(answer);

if (strcmp(answer, "Success\n") == 0) {

lcd\_print("1.Success");

\_delay\_ms(2000);

lcd\_clear\_display();

break;

}

else lcd\_print("1.Fail");

\_delay\_ms(2000);

lcd\_clear\_display();

} // end loop for ESP connection

// Sending command for url ...

while(1) {

transmit\_command("ESP:url:\"http://192.168.1.250:5000/data\"\n");

char answer\_no2[SIZE\_ANSWER];

receive\_response(answer\_no2);

if (strcmp(answer\_no2, "Success\n") == 0) {

lcd\_print("2.Success");

\_delay\_ms(2000);

lcd\_clear\_display();

break;

}

else lcd\_print("2.Fail");

\_delay\_ms(2000);

lcd\_clear\_display();

} // end loop for ESP url

bool pressed\_no3 = false;

bool pressed\_hashtag = false;

while(1) { // temperature, H2O, no3, hashtag

// Getting the temperature

uint16\_t temperature = GetTemperature();

int result = 0;

int dekadika = 0;

if (temperature == 0x8000) { // NO Device 9 bits no need for extra line

lcd\_clear\_display();

lcd\_print("No Device");

}

else {

if ((temperature & 0x8000) > 0) { // Negative found

temperature = ~temperature + 1;

} //if negative convert to its value

if((temperature&0x01)==0x01) dekadika += 625;

temperature = temperature >> 1;

if((temperature&0x01)==0x01) dekadika += 1250;

temperature = temperature >> 1;

if((temperature&0x01)==0x01) dekadika += 2500;

temperature = temperature >> 1;

if((temperature&0x01)==0x01) dekadika += 5000;

temperature = temperature >> 1 ;

for (int i = 0; i <= 6; i++){ // Integer part -> 7 bits

if (temperature & 0x0001 > 0) result += (int)pow(2,i);

temperature = temperature >> 1;

} // result saves our temperature

result += OFFSET; // Took human temperature

}

// Getting the ADC-PWM

PWM\_init();

ADC\_init();

// Connection of ADC0 with POT1

uint16\_t ADC\_value = ADC\_conversion(); // Read POT1

int DC\_VALUE = ADC\_value;

//OCR1A = DC\_VALUE;

\_delay\_ms(100); // Small delay for better performance

int pressure = DC\_VALUE \* 20 / 1024; // Took human pressure

// Waiting for keypad ...

uint8\_t input = 0x00;

uint8\_t savor;

//bool pressed\_no3, pressed\_hashtag;

char status[SIZE\_STATUS];

// crucial for keypad activation with PCA

PCA9555\_0\_write(REG\_OUTPUT\_0,0x00);

PCA9555\_0\_write(REG\_OUTPUT\_1,0x00);

// Till now none of buttons 3 or # is pressed, so status is 'OK'

if (pressure > 12 | pressure < 4) { snprintf(status, sizeof(status), "CHECK PRESSURE"); }

else if (result > 37 | result < 34) { snprintf(status, sizeof(status), "CHECK TEMP"); }

else if (pressed\_no3 == true && pressed\_hashtag == false) { snprintf(status, sizeof(status), "NURSE CALL"); }

else snprintf(status, sizeof(status), "OK");

lcd\_clear\_display();

lcd\_print("H20:");

if (pressure/10 != 0) { lcd\_data(pressure/10 + '0'); }

lcd\_data(pressure%10 + '0');

lcd\_print(" Temp:");

if (result/10 != 0) { lcd\_data(result/10 + '0'); }

lcd\_data(result % 10 + '0');

lcd\_data('.');

dekadika = dekadika/1000;

lcd\_data('0' + dekadika);

lcd\_command(0xC0); // New line

lcd\_print(&status);

\_delay\_ms(50);

input = check(); // Check for the first pressed key

// Begin checking for pressed buttons

if (input == 0xFF) continue;

//first digit -> 3

else {

savor = input;

while(1){

\_delay\_ms(15);

input = check(); // Check again for debouncing

if (input == savor) continue;

else break;

}

if(savor == 0xB7) {

pressed\_no3 = true; // Found digit 3

pressed\_hashtag = false;

snprintf(status, sizeof(status), "NURSE CALL"); // Update status because symbol # found

lcd\_clear\_display();

lcd\_print(&status);

//break; // And go check for second -> #

}

else if (savor == 0xBE && pressed\_no3 == true) {

pressed\_hashtag = true;

pressed\_no3 = false;

}

}

}

}