Εθνικό **Μ**ετσόβιο **Π**ολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Εργαστήριο Μικροϋπολογιστών, ακ. έτος 2024 - 2025

Λύσεις στην 8η Εργαστηριακή Άσκηση

Όνομα: Ειρήνη	Όνομα: Γεώργιος
Επώνυμο: Σιμιτζή	Επώνυμο: Οικονόμου
ΑΜ: 03121063	ΑΜ: 03121103
Εργαστηριακή Ομάδα: 30	

Εισαγωγή

Πριν αναλύσουμε τον κώδικα της συνάρτησης main(), φροντίζουμε να εισάγουμε όλες τις βοηθητικές συναρτήσεις και ρουτίνες για τα διάφορα τμήματα και τα πρωτόκολλα της ntuAboard_G1, και από προηγούμενες εργαστηριακές, όπως τη ρουτίνα της USART και το πρωτόκολλο του TWI.

Για την κατάλληλη κατασκευή του payload σε μορφή JSON, εισάγουμε τη ρουτίνα payload_add, η οποία είναι η εξής:

```
void payload_add(const char *key, const char *value) {
   strcat(payload, "{\"name\": \"");
   strcat(payload, key);
   strcat(payload, "\",\"value\": \"");
   strcat(payload, value);
   strcat(payload, "\"},");
}
```

για το payload: `char payload[400];`.

Η συνάρτηση main() είναι:

```
int main ()
{
  twi_init();
  PCA9555_0_write(REG_CONFIGURATION_0, 0x00);
  PCA9555_0_write(REG_CONFIGURATION_1, 0xF0);
  lcd_init();

  // flag = 1 if already connected successfully, 0 otherwise
  int flag = 0;

  // ESP8266: Waiting for command, 'Waiting for command\0' = 20
  char restart_message[20];
  char connect_message[20];

  usart_init(103);
  usart_transmit_string("ESP:restart\n");
  usart_receive_string(restart_message);
  lcd_clear_display();

  while(1)
  {
     // lcd_clear_display();
     usart_transmit_string("ESP:connect\n");
     usart_receive_string(connect_message);

     if (strcmp(connect_message, "\"Success\"") == 0)
     {
        flag = 1;
        lcd_string("1.Success");
```

```
    _delay_ms(1000);
    lcd_clear_display();
}

else
{
    flag = 0;
    lcd_string("1.Fail");
    _delay_ms(1000);
    lcd_clear_display();
}

while(flag == 0)
{
    usart_transmit_string("ESP:connect\n");
    usart_receive_string(connect_message);

    // "\"Success\""
    if (strcmp(connect_message, "\"Success\"") == 0)
    {
        flag = 1;
        lcd_string("1.Success");
        _delay_ms(1000);
        lcd_clear_display();
    }

    else
    {
         flag = 0;
         lcd_string("1.Fail");
         _delay_ms(1000);
         lcd_clear_display();
    }
}

flag = 0;

// "ESP:url:\"http://192.168.1.250:5000/data\"" TO TEST WITH \n
usart_transmit_string("ESP:url:\"http://192.168.1.250:5000/data\"");
usart_receive_string(connect_message);

if (strcmp(connect_message, "\"Success\"") == 0)
{
    flag = 1;
    lcd_string("2.Success");
    _delay_ms(1000);
    lcd_clear_display();
}

else
```

```
{
   flag = 0;
   lcd_string("2.Fail");
   _delay_ms(1000);
   lcd_clear_display();
}

while(flag == 0)
{
   usart_transmit_string("ESP:url:\"http://192.168.1.250:5000/data\"\n");
   usart_receive_string(connect_message);

   if (strcmp(connect_message, "\"Success\"") == 0)
   {
      flag = 1;
      lcd_string("2.Success");
      _delay_ms(1000);
      lcd_clear_display();
   }

   else
   {
       flag = 0;
       lcd_string("2.Fail");
       _delay_ms(1000);
       lcd_clear_display();
   }
}

//flag = 0;
const char *status;
status = "OK";
int called=0;
while(1){

   // Temperature
   getvalue();

   if(r24==0x80 && r25==0x00)
   {
      // nop
      continue;
   }

   uint16_t temperature = (uint16_t)((r24 << 8) | r25);

   if(r24 & 10000000)
   {
      temperature=~temperature+1;
      // nop
```

```
   // lcd_data('-');
}

else
{
   //lcd_data('+');}
}

float temperature_new = temperature * 0.0625 * 1.47;
char temperature_string[5];
floatToString(temperature_new, temperature_string);
temperature_string[4] = '\0';

lcd_string(temperature_string);
lcd_data(223);   //code for degree symbol
lcd_data('C');
lcd_data(' ');

// Pressure
// MUX[3:0] = 0000 for ADC0
// REFS[1:0] = 01 for AVCC with external capacitor at AREF pin
ADMUX |= 0b01000000;
// ADIE = 1 => enable ADC interrupt,
// ADPS[2:0] = 111 => fADC = 16MHz/128 = 125KHz
ADCSRA |= 0b10001111;

ADCSRA |= (1<<ADSC);        // ADSC: ADC Start Conversion, Start ADC

while(ADCSRA & (1<<ADSC));
float pressure = ADC/50.0;   // 1024/50 = 20..

char pressure_string[5];
floatToString(pressure, pressure_string);
pressure_string[4] = '\0';

lcd_string(pressure_string);
lcd_string("cmH2O ");

// Keyboard
//const char *status;
//status = "OK";

if (keypad_to_ascii() == '0')
{
   status = "NURSE CALL";
   called=1;
}

if (keypad_to_ascii() == '#' && called==1)
{
```

```
    if ((temperature_new < 34) || (temperature_new > 37))
    {
       status = "CHECK TEMP";
    }

    else if ((pressure < 4.0) || (pressure > 12.0))
    {
       status = "CHECK PRESSURE";
    }

    else
    {
       status = "OK";
    }
}

// char team[] = "30";

// snprintf(payload, sizeof(payload),
// "ESP:payload:[{\"name\": \"temperature\",\"value\": \"%s\"},"
// "{\"name\": \"pressure\",\"value\": \"%s\"},"
// "{\"name\": \"team\",\"value\": \"%s\"},"
// "{\"name\": \"status\",\"value\": \"%s\"}]\n",
// temperature_string, pressure_string, team, status);

 // snprintf(payload, sizeof(payload),
 // "ESP:payload:[{\"name\":\"pressure\",\"value\":\"%s\"},"
 // "{\"name\":\"temperature\",\"value\":\"%s\"},"
 // "{\"name\":\"team\",\"value\":\"%s\"},"
 // "{\"name\":\"status\",\"value\":\"%s\"}]\n",
 // pressure_string, temperature_string, team, status);


strcpy(payload, "ESP:payload:[");
payload_add("temperature", temperature_string);
payload_add("pressure", pressure_string);
payload_add("team", "30");
payload_add("status", status);
payload[strlen(payload) - 1] = '\0';
strcat(payload, "]\n");

// lcd_string(pressure_string);
// lcd_string("cmH2O ");
lcd_command(0xC0);
lcd_string(status);

_delay_ms(1000);
lcd_clear_display();

// old version here
```

```
        usart_transmit_string(payload);
        usart_receive_string(connect_message);

        if (strcmp(connect_message, "\"Success\"") == 0)
        {
           flag = 1;
           lcd_string("3.Success");
           _delay_ms(1000);
           lcd_clear_display();
        }

        else
        {
           flag = 0;
           lcd_string("3.Fail");
           _delay_ms(1000);
           lcd_clear_display();
        }

        while(flag == 0)
        {
                  //usart_transmit_string("ESP:payload:[{\"name\":  \"team\",\"value\":
\"30\"}]\n");
           usart_transmit_string(payload);
           usart_receive_string(connect_message);

           if (strcmp(connect_message, "\"Success\"") == 0)
           {
              flag = 1;
              lcd_string("3.Success");
              _delay_ms(1000);
              lcd_clear_display();
           }

           else
           {
              flag = 0;
              lcd_string("3.Fail");
              _delay_ms(1000);
              lcd_clear_display();
           }
        }

        flag = 0;

        usart_transmit_string("ESP:transmit\n");
        usart_receive_string(connect_message);
        lcd_clear_display();
        lcd_string("4.");
        lcd_string(connect_message);
```

```
            lcd_command(0xC0);
            _delay_ms(2000);
        }
    }
}
```

Στο πλαίσιο του κώδικα αυτού, ο οποίος απαντά σταδιακά στα ερωτήματα 8.1, 8.2 και 8.3, έπειτα από τις απαραίτητες αρχικοποιήσεις, μπαίνουμε σε ένα while loop, όπου nested σε αυτό βρίσκονται όλες οι λειτουργίες που θέλουμε.

Η αποστολή του κάθε μηνύματος, ESP:connect, ESP://data και ESP:payload, από το AVR ακολουθεί την ίδια λογική. Αρχικά, αρχικοποιούμε τη μεταβλητή flag = 0 και στέλνουμε το μήνυμα μία πρώτη και δοκιμαστική φορά. Αν ο δέκτης ESP απαντήσει θετικά με "Success", τότε το flag = 1. Διαφορετικά, η ρουτίνα εισέρχεται σε ένα ατέρμονο while loop, όπου επιχειρούμε συνεχώς να λάβουμε "Success" από τον δέκτη και να ορίσουμε το flag = 1. Σε αυτή την περίπτωση, το while loop σπάει αφού έχει τυπωθεί το κατάλληλο μήνυμα στην LCD οθόνη.

Αφού έχουμε συνδεθεί στον server με την αποστολή του δεύτερου σταδίου, υπολογίζουμε τη θερμοκρασία και την πίεση για να κατασκευάσουμε το payload. Κλιμακώνουμε κατάλληλα τη θερμοκρασία, πολλαπλασιάζοντας την ποσότητα με *1.47, και εισάγουμε τα απαραίτητα if cases για το status.

Τέλος, στέλνουμε το payload με το ESP:transmit στον server και, εφόσον έχουν πάει όλα καλά, λαμβάνουμε απάντηση 200 OK.