ИНСТИТУТ ЗА МАТЕМАТИКУ И ИНФОРМАТИКУ

ПРИРОДНО-МАТЕМАТИЧКИ ФАКУЛТЕТ

УНИВЕРЗИТЕТА У КРАГУЈЕВЦУ

Систем за упозоравање о времену за узимање лекова

Предмет: Микропроцесорски системи

Студент:

Никола Рњак 80/2018

Професор:

др Александар Пеулић

Крагујевац, 2022. год.

Садржај

Увод

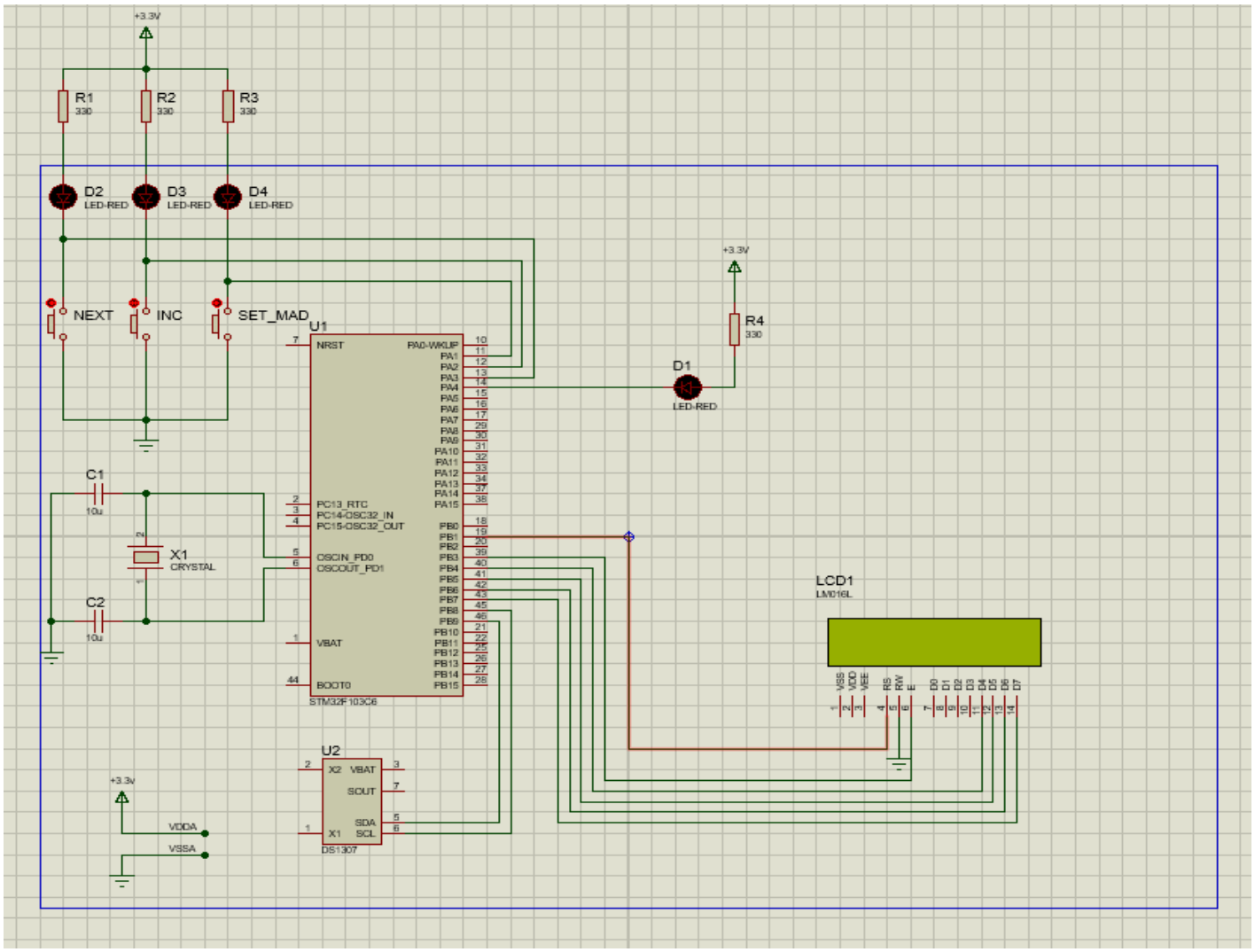
Циљ пројекта је реализација система који омогућава корисницима да добију информацију да треба да узму одређену терапију. Након што корисник унесе време за сваку до својих терапија, сваки дан ће алармом бити обавештен када је време за узимање терапије, а порука на екрану уређаја би га обавестила о којој терапији је реч. Развој обухвата како софтверски тако и хардверски предлог решења.

Proteus – шема пројекта

На слици је приказана шема пројекта, као замена за хардверску реализацију пројекта. Сем постојећих компоненти које можемо видети на слици, у хардверској реализацији бисмо користили и звучни сигнал приликом оглашавања аларма.

Три тастера SET_MAD, INC i NEXT служе за подешавање одговарајућих аларма. Повезана су са микроконтролером редом преко пинова 1-3 порта А.

LCD на ком се приказују тренутно време и датум, у случају оглашавања аларма приказиваће терапију коју корисник треба да узме и тренутно време. Повезан је са микроконтролером преко пинова 1-7 порта Б.



Диоде D2, D3 и D4 светле приликом притиска на одговарајући тастер. Притиском тастера

SET_MAD на LCD се приказује име подсетника и предефинисано време које корисник може да промени на жељено. Притиском NEXT тастера корисник се пребацује на подешавање минута тренутног аларма или уколико је већ подесио минуте прелази на подешавање сати наредног аларма. Тастер INC увећава сате (минуте) за један сат (минут). Када се огласи аларм диода D1, повезана на пин 4 порта A, почеће да блинка, а на LCD екрану појавиће се информације о терапији.

Објашњење појединачних делова кода

Фајл „lcd.c“

```
void Lcd_init(Lcd_HandleTypeDef * lcd)
{
    if(lcd->mode == LCD_4_BIT_MODE)
    {
            lcd_write_command(lcd, 0x33);
            lcd_write_command(lcd, 0x32);
            lcd_write_command(lcd, FUNCTION_SET | OPT_N);               // 4-bit mode
    }
    else
        lcd_write_command(lcd, FUNCTION_SET | OPT_DL | OPT_N);


    lcd_write_command(lcd, CLEAR_DISPLAY);                      // Clear screen
    lcd_write_command(lcd, DISPLAY_ON_OFF_CONTROL | OPT_D);     // Lcd-on, cursor-off, no-blink
    lcd_write_command(lcd, ENTRY_MODE_SET | OPT_INC);           // Increment cursor
}
```

Ова функција дефинише иницијално (почетно) стање lcd-а. Текст ће бити исписан у две линије, почевши од горњег левог угла. Курсор је искључен.

```
Lcd_HandleTypeDef Lcd_create(
        Lcd_PortType port[], Lcd_PinType pin[],
        Lcd_PortType rs_port, Lcd_PinType rs_pin,
        Lcd_PortType en_port, Lcd_PinType en_pin, Lcd_ModeTypeDef mode)
{
    Lcd_HandleTypeDef lcd;

    lcd.mode = mode;

    lcd.en_pin = en_pin;
    lcd.en_port = en_port;

    lcd.rs_pin = rs_pin;
    lcd.rs_port = rs_port;

    lcd.data_pin = pin;
    lcd.data_port = port;

    Lcd_init(&lcd);

    return lcd;
}
```

Овом функцијом креирамо lcd handler тако што дефинишемо пинове и портове које ће користити, а након тога позивамо функцију за иницијализацију.

```
void lcd_write_data(Lcd_HandleTypeDef * lcd, uint8_t data)
{
    HAL_GPIO_WritePin(lcd->rs_port, lcd->rs_pin, LCD_DATA_REG);         // Write to data register

    if(lcd->mode == LCD_4_BIT_MODE)
    {
        lcd_write(lcd, data >> 4, LCD_NIB);
        lcd_write(lcd, data & 0x0F, LCD_NIB);
    }
    else
    {
        lcd_write(lcd, data, LCD_BYTE);
    }

}

/**
 * Set len bits on the bus and toggle the enable line
 */
void lcd_write(Lcd_HandleTypeDef * lcd, uint8_t data, uint8_t len)
{
    for(uint8_t i = 0; i < len; i++)
    {
        HAL_GPIO_WritePin(lcd->data_port[i], lcd->data_pin[i], (data >> i) & 0x01);
    }

    HAL_GPIO_WritePin(lcd->en_port, lcd->en_pin, 1);
    DELAY(1);
    HAL_GPIO_WritePin(lcd->en_port, lcd->en_pin, 0);        // Data receive on falling edge
}
```

Функције којима се исписују одговарајући карактери на екран LCD-а.

```
void Lcd_int(Lcd_HandleTypeDef * lcd, int number)
{
    char buffer[11];
    sprintf(buffer, "%d", number);

    Lcd_string(lcd, buffer);
}
```

```
void Lcd_string(Lcd_HandleTypeDef * lcd, char * string)
{
    for(uint8_t i = 0; i < strlen(string); i++)
    {
        lcd_write_data(lcd, string[i]);
    }
}
```

Функције које као аргумент добијају број односно стринг и прослеђују један по један карактер функцијама за испис.

```
void Lcd_clear(Lcd_HandleTypeDef * lcd) {
    lcd_write_command(lcd, CLEAR_DISPLAY);
}
```

Функција којом чистимо сав садржај са екрана LCD-а.

```
void Lcd_send_command(Lcd_HandleTypeDef * lcd, uint8_t command)
{
    lcd_write_command(lcd, command);
}


void lcd_write_command(Lcd_HandleTypeDef * lcd, uint8_t command)
{
    HAL_GPIO_WritePin(lcd->rs_port, lcd->rs_pin, LCD_COMMAND_REG);      // Write to command register

    if(lcd->mode == LCD_4_BIT_MODE)
    {
        lcd_write(lcd, (command >> 4), LCD_NIB);
        lcd_write(lcd, command & 0x0F, LCD_NIB);
    }
    else
    {
        lcd_write(lcd, command, LCD_BYTE);
    }

}
```

Функције које извршавају одговарајуће команде. У овом пројекту то ће бити команде којима ћемо у ком реду на екрану LCD-а прослеђене информације треба да буду приказане.

Фајл „main.c“

```
#include "main.h"
#include "lcd.h"
#include "stm32f1xx_hal.h"

#define ALARM_LED GPIO_PIN_4

#define NEXT GPIO_PIN_3
#define INC GPIO_PIN_2
#define SET_MAD GPIO_PIN_1

#define MAX_REMINDERS 10
```

Макрои које смо дефинисали на почетку нашег кода.

```
void set_time (void)
{
      RTC_TimeTypeDef sTime;
      RTC_DateTypeDef sDate;
    /**Initialize RTC and set the Time and Date
    */
  sTime.Hours = 0x17;
  sTime.Minutes = 0x48;
  sTime.Seconds = 0x00;

  if (HAL_RTC_SetTime(&hrtc, &sTime, RTC_FORMAT_BCD) != HAL_OK)
  {
      Error_Handler();
  }

  sDate.WeekDay = RTC_WEEKDAY_SUNDAY;
  sDate.Month = RTC_MONTH_FEBRUARY;
  sDate.Date = 0x13;
  sDate.Year = 0x22;

  if (HAL_RTC_SetDate(&hrtc, &sDate, RTC_FORMAT_BCD) != HAL_OK)
  {
      Error_Handler();
  }

  HAL_RTCEx_BKUPWrite(&hrtc, RTC_BKP_DR1, 0x32F2);  // backup register
}
```

Функција којом инцијализујемо почетно време и датум нашег уређаја.

```
void get_time(void)
{
  RTC_DateTypeDef gDate;
  RTC_TimeTypeDef gTime;

  /* Get the RTC current Time */
  HAL_RTC_GetTime(&hrtc, &gTime, RTC_FORMAT_BIN);
  /* Get the RTC current Date */
  HAL_RTC_GetDate(&hrtc, &gDate, RTC_FORMAT_BIN);


  currentTime.Hour = gTime.Hours;
  currentTime.Min = gTime.Minutes;
  /* Display time Format: hh:mm:ss */
  sprintf((char*)time,"%02d:%02d:%02d",gTime.Hours, gTime.Minutes, gTime.Seconds);

  /* Display date Format: mm-dd-yy */
  sprintf((char*)date,"%02d-%02d-%2d",gDate.Date, gDate.Month, 2000 + gDate.Year);
}
```

Функција којом узимамо тренутно време и тренутни датум и чувамо их у глобалним стринг променљивама time и date.

```
void display_time (Lcd_HandleTypeDef *lcd)
{
    Lcd_send_command (lcd, 0x80);
    Lcd_string (lcd, "TIME: ");
    Lcd_string (lcd, time);
    Lcd_send_command (lcd, 0xc0);
    Lcd_string (lcd, "DATE: ");
    Lcd_string (lcd, date);
}
```

Функција којом исписујемо тренутно време и датум на екран LCD-а.

```
void beep(void)
{
    HAL_GPIO_WritePin(GPIOA, ALARM_LED, GPIO_PIN_RESET);
    HAL_Delay(200);
    HAL_GPIO_WritePin(GPIOA, ALARM_LED, GPIO_PIN_SET);
    HAL_Delay(100);
    HAL_GPIO_WritePin(GPIOA, ALARM_LED, GPIO_PIN_RESET);
    HAL_Delay(200);
    HAL_GPIO_WritePin(GPIOA, ALARM_LED, GPIO_PIN_SET);
    HAL_Delay(100);
}
```

Функција која ће приликом позива наизменично два пута да упали и угаси диоду D1, која симулира активацију аларма.

```
Lcd_PortType ports[] = {
    GPIOB, GPIOB, GPIOB, GPIOB
};

Lcd_PinType pins[] = {
    GPIO_PIN_4, GPIO_PIN_5, GPIO_PIN_6, GPIO_PIN_7
};

Lcd_HandleTypeDef lcd = Lcd_create(ports, pins, GPIOB, GPIO_PIN_1, GPIOB, GPIO_PIN_3, LCD_4_BIT_MODE);

if(HAL_RTCEx_BKUPRead(&hrtc, RTC_BKP_DR1) != 0x32F2)
{
  //inicijalizacija vremena i datuma
  set_time();
}

HAL_GPIO_WritePin(GPIOA, ALARM_LED, GPIO_PIN_SET);
```

У main делу програма након декларисања промењљивих иницијализујемо почетно стање LCD-а на коме ће да се приказују почетно време и датум.

```
while (1)
{
    /* USER CODE BEGIN 3 */
    //ucitaj trenutno vreme i datum
    get_time();
    //prikazi trenutno vreme i datum
    display_time(&lcd);

    //provera da li postoji neki alarm u ovo vreme
    for(i=0; i<medicineRemindersCount; i++){
        if(medicineReminders[i].Hour == currentTime.Hour
          && medicineReminders[i].Min == currentTime.Min){
            Lcd_clear(&lcd);

            Lcd_send_command(&lcd, 0x80);
            sprintf((char*)reminderName,"TAKE MEDICINE %02d", i + 1);
            Lcd_string(&lcd, reminderName);

            Lcd_send_command(&lcd, 0xc0);
            sprintf((char*)reminderTime,"ALARM TIME: %02d:%02d", medicineReminders[i].Hour,
                    medicineReminders[i].Min);
            Lcd_string(&lcd, reminderTime);

            beep();
            beep();
            beep();
        }
    }
    HAL_Delay(300);
```

Након иницијализације улазимо у бесконачну петљу, где се у свакој итерацији петље чита тренутно време и датум и исписује на екран. Сем тога стално проверавамо и да ли је

време да се неки од подешених аларма активирају. Ако је време да се активира неки аларм чистимо екран LCD-а и исписујемо одговарајућу поруку коју терапију корисник треба да узме, као и тренутно време. Након тога позивамо функцију beep() која ће да укључи наш аларм тј. да крене да блинка диода која представља аларм.

```
//provera da li je pritisnuto dugme za podesavanje alarma
if(HAL_GPIO_ReadPin(GPIOA, SET_MAD) == 0){
    while(HAL_GPIO_ReadPin(GPIOA, SET_MAD) == 0);

    tempTime.Hour = 0x00;
    tempTime.Min = 0x00;
    i = 0;

    Lcd_clear(&lcd);
    int ind = 0;
    //dok god ne pritisnemo opet dugme da potvrdimo unete alarme ili dok nije dostignut max broj alarma
    //podesavamo alarme
    while(HAL_GPIO_ReadPin(GPIOA, SET_MAD) != 0 && i < MAX_REMINDERS-1) {
        ind = 1;
        Lcd_send_command(&lcd, 0x80);
        sprintf((char*)reminderName,"REMINDER %02d", i + 1);
        Lcd_string(&lcd, reminderName);

        Lcd_send_command(&lcd, 0xc0);
        sprintf((char*)reminderTime,"%02d:%02d", tempTime.Hour, tempTime.Min);
        Lcd_string(&lcd, reminderTime);
```

Поред исписа тренутног времена и провере аларма, проверавамо и да ли је корисник притиснуо тастер SET_MAD тј. тастер за подешавање нових времена за узимање терапије. Уколико јесте постављамо предефинисано време на 0 часова и 0 минута и приказујемо на LCD екрану поруку са редним бројем терапије, а у другом реду приказујемо предефинисано време, које корисник може да измени. Док год опет не притиснемо тастер SET_MAD или док год се не унесе максимални број терапија које уређај може да запамти, наши аларми за терапије неће бити сачувани.

```
//uvecanje sata ili minuta za +1
if(HAL_GPIO_ReadPin(GPIOA, INC) == 0){
    while(HAL_GPIO_ReadPin(GPIOA, INC) == 0);
    if(hourIndicator == 1){
        tempTime.Hour++;
        if(tempTime.Hour == 0x18){
            tempTime.Hour = 0x00;
        }
    }
    else {
        tempTime.Min++;
        if(tempTime.Min == 0x3C){
            tempTime.Min = 0x00;
        }
    }
}
```

Када смо кренули са уносом терапија, предефинисане сате или минуте можемо увећати за један, тако што ћемо притиснути тастер INC. Овиме подешавамо време које нам одговара за постављање аларма.

```
//prelazak na minute ili na sledeci alarm
if(HAL_GPIO_ReadPin(GPIOA, NEXT) == 0){
    while(HAL_GPIO_ReadPin(GPIOA, NEXT) == 0);
    hourIndicator = -1 * hourIndicator;
    minIndicator = -1 * minIndicator;

    if(hourIndicator == 1){
       medicineReminders[i].Hour = tempTime.Hour;
       medicineReminders[i].Min = tempTime.Min;

       tempTime.Hour = 0x00;
       tempTime.Min = 0x00;

       ++i;
    }
}
```

Уколико смо подесили сате па сад желимо минуте, на њих се пребацујемо тако што притиснемо тастер NEXT. Када завршимо и са подешавањем минута, поновним притиском тастера NEXT чувамо тренутно подешену терапију тј. аларм и прелазимо на креирање нове при чему се време враћа на предефинисано, а то је 0 сати и 0 минута.

```
if(ind && i < MAX_REMINDERS-1){
    medicineReminders[i].Hour = tempTime.Hour;
    medicineReminders[i].Min = tempTime.Min;
    ++i;
}

//kada zavrsimo sa unosom, broj unetih cuvamo
medicineRemindersCount = i;
Lcd_clear(&lcd);
```

Када смо унели максималан број терапија или када смо притиснули поново тастер SET_MAD како бисмо потврдили унос нових терапија, чува се последња терапија, чисти се екран LCD-а и на екрану се поново исписују тренутно време и датум.