

Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki



Elektronika Biomedyczna - projekt
17. Układ mierzący czas reakcji

Studia niestacjonarne, sem. VII / NS1, EiT4/E1, rok akad. 2022/2023

Prowadzący: **dr hab. inż. Tomasz Przybyła**

Skład sekcji:

Marcin Szoltysek

Damian Śnieguła

SPIS TREŚCI

1 CEL	2
2 OPIS TECHNICZNY	2
3 SCHEMAT IDEOWY	2
4 KOD PROGRAMU ATMEGA328P	3
Wyświetlanie LCD:	3
Program główny:	4
5 WYKAZ ELEMENTÓW	8
6 PROTOTYP	8

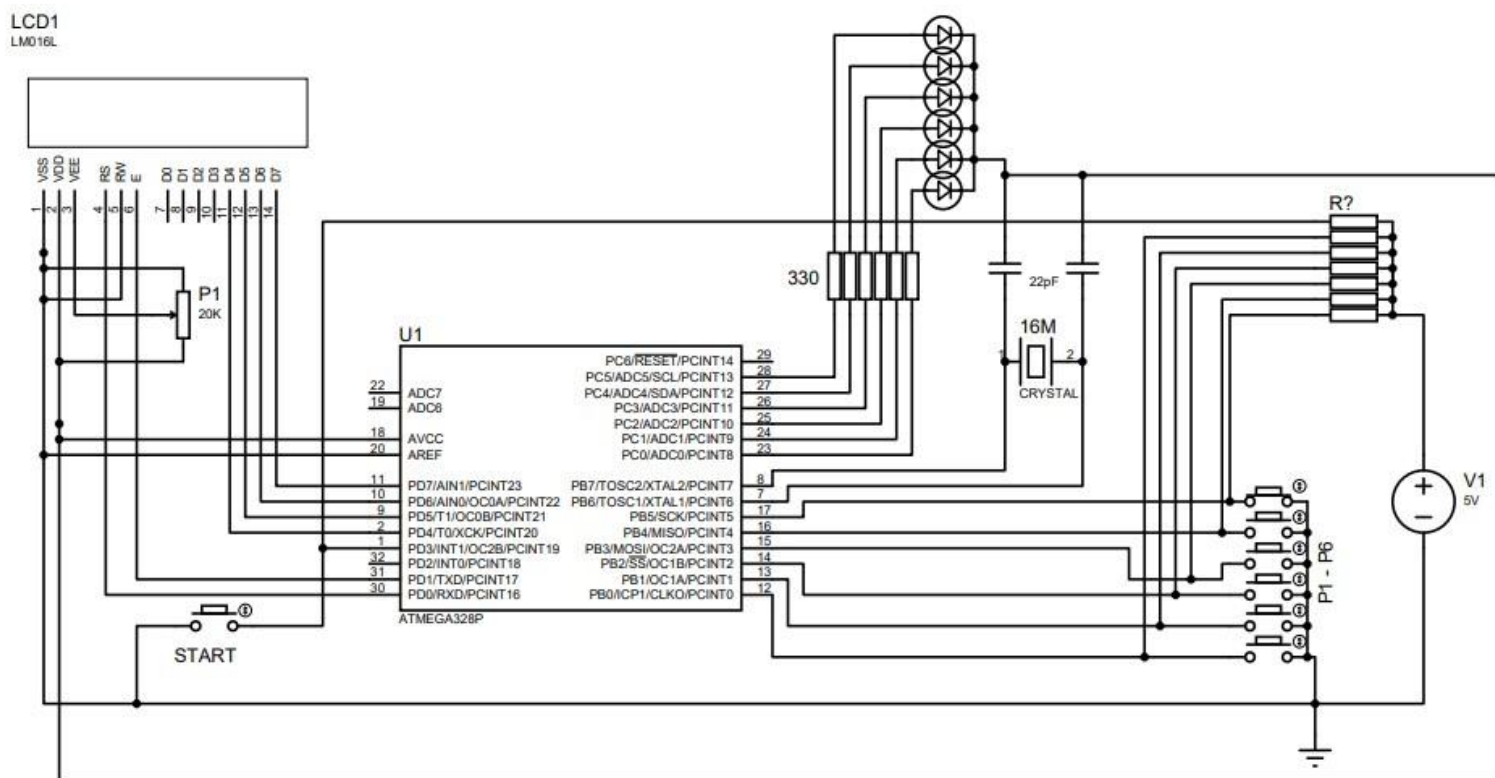
1 CEL

Celem projektu jest wykonanie urządzenia elektronicznego, który będzie mierzył czas reakcji pacjentów. Urządzenie powinno działać w taki sposób żeby mierzył średnią, najmniejszą, największą wartość czasu reakcji oraz ilość popełnionych błędów podczas badania.

2 OPIS TECHNICZNY

Urządzenie zostało zaprojektowane w oparciu o mikrokontroler *ATMEGA328P*, a informacje o przebiegu badania pojawiają się na wyświetlaczu LCD *LMD16L*. Pacjent po wciśnięciu przycisku "START" ma 3 sekundy na przygotowanie się. Następnie tak szybko jak to możliwe powinien wciskać przyciski nad którymi zaświecają się diody. Pod koniec badania używając przycisku "START" można przechodzić między wynikami pomiaru.

3 SCHEMAT IDEOWY



4 KOD PROGRAMU ATMEGA328P

Wyświetlanie LCD:

```
#define LCD_Dir DDRD          // PRZYPISANIE DDR WYŚWIETLACZA
#define LCD_Port PORTD        // PRZYPISANIE PORT WYŚWIETLACZA
#define RS PD0                // PRZYPISANIE PIN RS
#define EN PD1                // PRZYPISANIE PIN EN

void LCD_komenda( unsigned char cmnd )
{
    LCD_Port = (LCD_Port & 0x0F) | (cmnd & 0xF0);    // WYSYŁANIE 4 BARDZIEJ ZNACZĄCYCH BITÓW
    LCD_Port &= ~(1<<RS);                            // RS = 0
    LCD_Port |= (1<<EN);                              // EN = 1
    _delay_us(1);
    LCD_Port &= ~(1<<EN);                            // EN = 0
    _delay_us(200);

    LCD_Port = (LCD_Port & 0x0F) | (cmnd << 4);    // WYSYŁANIE 4 MNIEJ ZNACZĄCYCH BITÓW
    LCD_Port |= (1<<EN);                            // EN = 1
    _delay_us(1);
    LCD_Port &= ~(1<<EN);                            // EN = 0
    _delay_ms(2);
}

void LCD_napis( unsigned char data )
{
    LCD_Port = (LCD_Port & 0x0F) | (data & 0xF0);    // WYSYŁANIE 4 BARDZIEJ ZNACZĄCYCH BITÓW
    LCD_Port |= (1<<RS);                            // RS = 1
    LCD_Port |= (1<<EN);                              // EN = 1
    _delay_us(1);
    LCD_Port &= ~(1<<EN);                            // EN = 0
    _delay_us(200);
    LCD_Port = (LCD_Port & 0x0F) | (data << 4);    // WYSYŁANIE 4 MNIEJ ZNACZĄCYCH BITÓW
    LCD_Port |= (1<<EN);                            // EN = 1
    _delay_us(1);
    LCD_Port &= ~(1<<EN);                            // EN = 0
    _delay_ms(2);
}

void LCD_start (void)
{
    LCD_Dir = 0xFF;                                // USTAWIENIE PORTU JAKO WYJŚCIA
    _delay_ms(20);
    LCD_komenda(0x02);                            // 4 BITOWA INICJALIZACJA
    LCD_komenda(0x28);                            // 2 WIERZSZE, MACIERZ 5x7 W TRYB 4 BIT
    LCD_komenda(0x0c);                            // WYŁĄCZENIE KURSORA
    LCD_komenda(0x06);                            // PRZESUNIĘCIE KURSORA W PRAWO
    LCD_komenda(0x01);                            // CZYSZCZENIE WYŚWIETLANIA
    _delay_ms(2);
}

void LCD_wyswietl( char *str)
{
    int i;
    for(i=0;str[i]!=0;i++)                        // WYŚLIJ KAZDY ZNAK ŁAŃCUCHA
    {
        LCD_napis (str[i]);
    }
}

void LCD_wyswietl_pozycja (char wiersz, char kolumna, char *napis)
{
    if (kolumna == 0 && kolumna<16)
        LCD_komenda((kolumna & 0x0F)|0x80);
    else if (wiersz == 1 && kolumna<16)
        LCD_komenda((kolumna & 0x0F)|0xC0);
    LCD_wyswietl(napis);
}

void LCD_wyczyszc()
{
    LCD_komenda (0x01);                            // CZYSZCZENIE WYŚWIETLACZA
    _delay_ms(2);
    LCD_komenda (0x80);                            // KURSOR W DOMYŚLNEJ POZYCJI
}

}
```

```
uint32_t x=0, losowa_liczba, najwiekszy_czas=0, najmniejszy_czas=9999, suma_czasow=0, sredni_czas, liczba_bledow, y=15, z=1;
uint32_t czas_ms[] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}, czas_start;
volatile uint32_t zliczanie=0, pomocnicza=0;
```

4

```

    case 0:
    {
        zliczanie=0;
        PORTC |= (1<<PC0);
        while (bit_is_set(PINB,0))
    {
        if(bit_is_clear(PINB, 1))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
        if(bit_is_clear(PINB, 2))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
        if(bit_is_clear(PINB, 3))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
        if(bit_is_clear(PINB, 4))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
        if(bit_is_clear(PINB, 5))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
    };

        czas_ms[x] = zliczanie/4;
        break;
    }

    case 1:
    {
        zliczanie=0;
        PORTC |= (1<<PC1);
        while (bit_is_set(PINB,1))
    {
        if(bit_is_clear(PINB, 0))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
        if(bit_is_clear(PINB, 2))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
        if(bit_is_clear(PINB, 3))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
        if(bit_is_clear(PINB, 4))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
        if(bit_is_clear(PINB, 5))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
    };

        czas_ms[x] = zliczanie/4;
        break;
    }

    case 2:
    {
        zliczanie=0;
        PORTC |= (1<<PC2);
        while (bit_is_set(PINB,2))
    {
        if(bit_is_clear(PINB, 1))
        {liczba_bledow++;LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
        if(bit_is_clear(PINB, 0))
        {liczba_bledow++;LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
        if(bit_is_clear(PINB, 3))
        {liczba_bledow++;LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
        if(bit_is_clear(PINB, 4))
        {liczba_bledow++;LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
        if(bit_is_clear(PINB, 5))
        {liczba_bledow++;LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
    }

        czas_ms[x] =zliczanie/4;
        break;
    }

```

while

case 3:

```

        {
            zliczanie=0;
            PORTC |= (1<<PC3);
            while (bit_is_set(PINB,3))

        {
            if(bit_is_clear(PINB, 1))
            {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
            if(bit_is_clear(PINB, 2))
            {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
            if(bit_is_clear(PINB, 0))
            {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
            if(bit_is_clear(PINB, 4))
            {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
            if(bit_is_clear(PINB, 5))
            {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
        }

        czas_ms[x] = zliczanie/4;
        break;
    }

    case 4:
    {
        zliczanie=0;
        PORTC |= (1<<PC4);
        while (bit_is_set(PINB,4))

    {
        if(bit_is_clear(PINB, 1))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
        if(bit_is_clear(PINB, 2))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
        if(bit_is_clear(PINB, 3))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
        if(bit_is_clear(PINB, 0))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
        if(bit_is_clear(PINB, 5))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
    }

        czas_ms[x] = zliczanie/4;
        break;
    }

    case 5:
    {
        zliczanie=0;
        PORTC |= (1<<PC5);
        while (bit_is_set(PINB,5))

    {
        if(bit_is_clear(PINB, 1))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,1)); _delay_ms(20);}
        if(bit_is_clear(PINB, 2))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,2)); _delay_ms(20);}
        if(bit_is_clear(PINB, 3))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,3)); _delay_ms(20);}
        if(bit_is_clear(PINB, 4))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,4)); _delay_ms(20);}
        if(bit_is_clear(PINB, 0))
        {liczba_bledow++; LCD_wyswietl_pozycja(1, 6, "Blad!"); while (bit_is_clear(PINB,5)); _delay_ms(20);}
    }

        czas_ms[x] = zliczanie/4;
        break;
    }

} //koniec switch

```

```

PORTC &= ~(1<<PC0);
PORTC &= ~(1<<PC1);

```

```

PORTC &= ~(1<<PC2);
PORTC &= ~(1<<PC3);
PORTC &= ~(1<<PC4);
PORTC &= ~(1<<PC5);
_delay_ms(500);
x++;
} //koniec while(x<20)

x=0;
while (x<y)
{
    if (czas_ms[x] > najwiekszy_czas)
    {najwiekszy_czas = czas_ms[x];}
    if (czas_ms[x] < najmniejszy_czas)
    {najmniejszy_czas = czas_ms[x];}
    suma_czasow = suma_czasow + czas_ms[x];
    x++;
}

sredni_czas = suma_czasow/y;
LCD_wyczyszc();
LCD_wyswietl("KONIEC BADANIA!");
_delay_ms(1000);

char najwiekszy_czas_tekst[7];
itoa(najwiekszy_czas, najwiekszy_czas_tekst, 10);
LCD_wyczyszc();
LCD_wyswietl("MAKS.");
LCD_wyswietl_pozycja(0, 10, najwiekszy_czas_tekst);
LCD_wyswietl_pozycja(0, 14, "ms");
LCD_wyswietl_pozycja(1, 1, "WCISNIJ START>>");
_delay_ms(500);
while (bit_is_set(PIND,2)) {};

char najmniejszy_czas_tekst[7];
itoa(najmniejszy_czas, najmniejszy_czas_tekst, 10);
LCD_wyczyszc();
LCD_wyswietl("MIN.");
LCD_wyswietl_pozycja(0, 10, najmniejszy_czas_tekst);
LCD_wyswietl_pozycja(0, 14, "ms");
LCD_wyswietl_pozycja(1, 1, "WCISNIJ START>>");
_delay_ms(500);
while (bit_is_set(PIND,2)) {};

char sredni_czas_tekst[7];
itoa(sredni_czas, sredni_czas_tekst, 10);
LCD_wyczyszc();
LCD_wyswietl("SREDN.");
LCD_wyswietl_pozycja(0, 10, sredni_czas_tekst);
LCD_wyswietl_pozycja(0, 14, "ms");
LCD_wyswietl_pozycja(1, 1, "WCISNIJ START>>");
_delay_ms(500);
while (bit_is_set(PIND,2)) {};

char liczba_bledow_tekst[7];
itoa(liczba_bledow, liczba_bledow_tekst, 10);
LCD_wyczyszc();
LCD_wyswietl("BLEDY");
LCD_wyswietl_pozycja(0, 10, liczba_bledow_tekst);
LCD_wyswietl_pozycja(1, 1, "WCISNIJ START>>");
_delay_ms(500);
while (bit_is_set(PIND,2)) {};
z=1;

} //koniec if

} //koniec while (1)

return 0;
} //koniec int main

```


5 WYKAZ ELEMENTÓW

Projekt wykonano z użyciem dwóch płytek stykowych oraz przewodów połączeniowych. Do wgrzywania programu posłużono się konwerter USB-UART FTDI.

Element	Ilość
Mikrokontroler AVR - ATmega328P-PU DIP	1
Rezonator kwarcowy 16MHz - HC49	1
Kondensator ceramiczny 22pF/50V	2
1602 Wyświetlacz LCD 2x16 znaków	1
Potencjometr montażowy leżący 20kΩ	1
Tact Switch 6x6mm / 5mm	7
Dioda LED 5mm czerwona	6
Rezystor węglowy 1/4W 330kΩ	6
Rezystor węglowy 1/4W 10kΩ	7

6 PROTOTYP

