# Politechnika Opolska

# L A B O R A T O R I U M

PRZEDMIOT: ***Technika mikroprocesorowa***

KIERUNEK STUDIÓW: ***Automatyka i Robotyka*** ROK STUDIÓW: ***III***

SPECJALNOŚĆ: **-**

SEMESTR: **VI** ROK AKADEMICKI: ***2019/2020***

*Nr ćwiczenia:* **1**

*Temat ćwiczenia:*

***Obsługa portów - MSP430.***

*Ćwiczenie wykonali:*

| *Nazwisko:* | *Imię:* | *Nazwisko:* | *Imię:* |
|---|---|---|---|
| **1. Dziembowski** | **Mateusz** | **2. --** | **--** |
| **3. --** | **--** | **4.** | |

| *Uwagi:* | *Data:* | *Ocena za sprawozdanie:* |
|---|---|---|
| | | |

# 1. Założenia

Wykorzystując zestaw MSP-EXP430G2 produkcji Texas Instruments, napisać program realizujący sygnalizator optyczny reagujący na kolejne naciśnięcia przycisku. Stan programu ( numer kolejnego naciśnięcia przycisku S2) odzwierciedlany będzie zachowaniem pokładowych diod świetlnych (LED1, LED2) w konfiguracji podanej w tabeli ***1.1*** poniżej ***:***

**Tabela 1-1** *Tabela stanów sygnalizatora optycznego*

| stan programu | LED1 czerwona | LED2 zielona |
|---|---|---|
| uruchomienie | OFF | OFF |
| pierwsze naciśnięcie przycisku S2 | OFF | ON |
| drugie naciśnięcie przycisku S2 | ON | ON |
| trzecie naciśnięcie przycisku S2 | OFF | OFF |
| czwarte naciśnięcie przycisku S2 | OFF | PULS |
| piąte naciśnięcie przycisku S2 | PULS | OFF |
| szóste naciśnięcie przycisku S2 | PULS | PULS |
| | synchronicznie | |
| siódme naciśnięcie przycisku S2 | PULS | PULS |
| | naprzemiennie | |
| ósme naciśnięcie przycisku S2 | OFF | OFF |
| | kasacja zliczania | |

Ponadto , w programie zawarty zostanie fragment odpowiedzialny za reagowanie tylko na zadaną długość czasu naciśnięcia przycisku. Krótkie naciśnięcia przycisku S2 ( fałszywe alarmy ) nie będą zmieniały stanu programu.

# 2. Opis kodu programu

W linii nr 6 programu głównego, włączany jest ***Stop Watchdog Timer.*** Następnie w liniach 7 i 8 następuje ustawienie portu nr 1 Bit0 oraz Bit6 jako wyjście ( P1.0 – LED1 , P1.6 – LED2) oraz w liniach 9 i 10 ustawienie wyjść na off. Linie 11-13 dotyczą przycisku S2 podłączonego do P1.3 , zadeklarowania wejścia oraz włączenia rezystora podciągającego. Opisywany fragment znajduje się poniżej :

```
4  int main(void)
5  {
6      WDTCTL = WDTPW + WDTHOLD;
7      P1DIR |= BIT0;                  // ustawienie  P1.0 (czerwona-LED) jako wyjście
8      P1DIR |= BIT6;                  // ustawienie  P1.6 (zielona-LED) jako wyjście
9      P1OUT &= ~BIT6;                 // ustawienie  P1.6 na off
10     P1OUT &= ~BIT0;                 // ustawienie  P1.0 na off
11     P1DIR &= ~BIT3;                 // ustawienie  P1.3 (przycisk S2) jako wejście
12     P1REN |=  BIT3;                 // wlaczenie rezystora pull-up na P1.3 (przycisk S2)
13     P1OUT |= BIT3;                  // gdy przycisk jest w górze
```

Następnie zadeklarowano kilka zmiennych w typie ***integer*** wykorzystywanych pomocniczo w dalszych częściach programu. Zmienna ***czulosc*** odpowiada za ustalenie minimalnej długości czasu naciśnięcia przycisku S2. Podlegające możliwym zmianom są również zmienne ***t_on*** oraz ***t_off*** odpowiadające za przebieg pulsacji.

```
14      int a=0;                    // zmienna pomocnicza
15      int b=0;                    // zmienna pomocnicza
16      int i;                      // zmienna pomocnicza
17      int j;                      // zmienna pomocnicza
18      int czulosc=1000;           // ustalenie dlugosci nacisniecia przycisku do dzialania
19      int t_on=30000;             // czas stanu wysokiego w pulsach
20      int t_off=30000;            // czas stanu niskiego w pulsach
```

Program wykonywany jest w pętli ***while*** zadeklarowanej w linii 23. Następnie w pętli ***do-while*** w liniach 28-39 z warunkiem naruszenia przycisku S2 inkrementowana jest zmienna **a** tak długo jak długo naciśnięty jest włącznik. Z chwilą gdy inkrementowane **a** zrówna się ustawionej czułości ( linia 32 ) następuje zwiększenie zmiennej **b** o 1 co narzuci wykonanie danego kroku w dalszej części . Opisywany fragment poniżej :

```
23    while(1)
24    {
25        /* --------- petla do-while ----------------------------------
26       pomocniczo do interpretacji nacisniecia i jego dlugosci na s2
27       ------------------------------------------------------------*/
28      do
29      {
30          if( (P1IN & BIT3 ) == 0)       // gdy S2 wcisniety gdy bit 3 == 0
31            {  a = a+1;                  // inkrementacja zmiennej a gdy wcisniety s2
32                if (a==czulosc)
33                    {b=b+1;}            // zwiększanie wartości zmiennej b gdy inkrementowane a
34                                        //       jest równe ustawionej czulosci
35            }
36          else
37            {   a=0; }                 // wyzerowanie a , gdy s2 nie jest wcisniety
38       }
39      while((P1IN & BIT3 ) == 0);
40
```

Jeżeli zmienna **b** przyjmie wartość **b=1** wówczas nastąpi załączenie LED2 :

```
40
41      /*-------warunki i akcje ----------------------------------------*/
42
43        if (b==1)                    // pierwsze nacisniecie - dioda 2 on
44        { P1OUT |=BIT6; }
```

Jeżeli zmienna **b** przyjmie wartość **b=2** wówczas nastąpi załączenie LED1 (LED2 pozostaje włączone z poprzedniego kroku) :

```
45        if (b==2)                    // drugie nacisniecie - dioda 1 on
46        { P1OUT |=BIT0; }
```

Jeżeli zmienna **b** przyjmie wartość **b=3** wówczas nastąpi wyłączenie obu diod :

```
47        if (b==3)                    // trzecie nacisniecie - obie diody off i przypisanie b=0;
48        { P1OUT &= ~BIT6;P1OUT &= ~BIT0; }
```

Jeżeli zmienna **b** przyjmie wartość **b=4** wówczas nastąpi pulsowanie diody LED2 . Pulsowanie w liniach 52-55 wykonane jest w oparciu o pętle ***for*** liczące do nastaw zmiennych **t_on** i **t_off** . W komentarzu (linie 57-60) zawarto alternatywny sposób rozwiązania problemu z wykorzystaniem ***delay_cyclec.*** Opisywany fragment poniżej:

```
49        i=0;
50        j=0;
51        if (b==4)                     // czwarte nacisniecie - pulsowanie diody 2 (zielona)
52        {   for(i;i<t_on;i++)        // czas wlaczenia
53            {P1OUT |=BIT6;}          // czas wylaczenia
54            for(j;j<t_off;j++)
55            {P1OUT &= ~BIT6;}
56
57            /*P1OUT |=BIT6;
58             delay_cycles(300000);
59            P1OUT &= ~BIT6;
60             delay_cycles(300000);*/
61
62
63        }
```

Jeżeli zmienna **b** przyjmie wartość **b=5** wówczas nastąpi pulsowanie diody LED1 :

```
64        if (b==5)                     // piate nacisniecie - pulsowanie diody 1 (czerwona)
65        {   for(i;i<t_on;i++)        // czas wlaczenia
66            {P1OUT |=BIT0;}
67            for(j;j<t_off;j++)        // czas wylaczenia
68            {P1OUT &= ~BIT0;}
69        }
```

Jeżeli zmienna **b** przyjmie wartość **b=6** wówczas nastąpi pulsowanie synchroniczne obu diod LED :

```
70        if (b==6)                    // szoste nacisniecie - pulsowanie obu diód synchronicznie
71        {    for(i;i<t_on;i++)       // czas wlaczenia
72             {P1OUT |=BIT6;P1OUT |=BIT0;}
73             for(j;j<t_off;j++)      // czas wylaczenia
74             {P1OUT &= ~BIT6;P1OUT &= ~BIT0;}
75        }
```

Jeżeli zmienna **b** przyjmie wartość **b=7** wówczas nastąpi pulsowanie naprzemienne obu diod LED :

```
76        if (b==7)                    // siódme nacisniecie - pulsowanie obu diód naprzemiennie
77        {    for(i;i<t_on;i++)       // czas wlaczenia
78             {P1OUT |=BIT6;P1OUT &= ~BIT0;}
79             for(j;j<t_off;j++)      // czas wylaczenia
80             {P1OUT &= ~BIT6;P1OUT |=BIT0;}
81        }
```

Jeżeli zmienna **b** przyjmie wartość **b=8** wówczas nastąpi wyłaczenie obu diod LED i przypisanie zmiennej b wartości **b=0**:

```
82        if (b==8)                    // wylaczenie
83        {  P1OUT &= ~BIT6; P1OUT &= ~BIT0; b=0;}
84
85    }
86
87
88    return 0;
89}
```

# 3. Pełen kod programu

```
/* PROGRAM SYGNALIZATORA OPTYCZNEGO 04032020*/
/* AiR_ns 2019/2020  */
/* Mateusz Dziembowski */

#include <msp430.h>
#include <intrinsics.h>

int main(void)
{
   WDTCTL = WDTPW + WDTHOLD;
   P1DIR |= BIT0;                // ustawienie  P1.0 (czerwona-LED) jako wyjście
   P1DIR |= BIT6;                // ustawienie  P1.6 (zielona-LED) jako wyjście
   P1OUT &= ~BIT6;               // ustawienie  P1.6 na off
   P1OUT &= ~BIT0;               // ustawienie  P1.0 na off
   P1DIR &= ~BIT3;               // ustawienie  P1.3 (przycisk S2) jako wejście
   P1REN |=  BIT3;               // wlaczenie rezystora pull-up na P1.3 (przycisk S2)
   P1OUT |= BIT3;                // gdy przycisk jest w górze
    int a=0;                   // zmienna pomocnicza
    int b=0;                   // zmienna pomocnicza
    int i;                     // zmienna pomocnicza
    int j;                     // zmienna pomocnicza
    int czulosc=1000;          // ustalenie dlugosci nacisniecia przycisku do dzialania
    int t_on=30000;            // czas stanu wysokiego w pulsach
    int t_off=30000;           // czas stanu niskiego w pulsach


   while(1)
   {
     /* --------- petla do-while ----------------------------------
     pomocniczo do interpretacji nacisniecia i jego dlugosci na s2
     ------------------------------------------------------------*/
    do
    {
      if( (P1IN & BIT3 ) == 0)     // gdy S2 wcisniety gdy bit 3 == 0
        {  a = a+1;              // inkrementacja zmiennej a gdy wcisniety s2
           if (a==czulosc)
              {b=b+1;}         // zwiększanie wartości zmiennej b gdy inkrementowane a
                          //     jest równe ustawionej czulosci
        }
       else
        {   a=0; }           // wyzerowanie a , gdy s2 nie jest wcisniety
    }
    while((P1IN & BIT3 ) == 0);

    /*-------warunki i akcje ----------------------------------------*/

     if (b==1)                // pierwsze nacisniecie - dioda 2 on
     { P1OUT |=BIT6; }
     if (b==2)                // drugie nacisniecie - dioda 1 on
     { P1OUT |=BIT0; }
     if (b==3)               // trzecie nacisniecie - obie diody off i przypisanie b=0;
     { P1OUT &= ~BIT6;P1OUT &= ~BIT0; }
     i=0;
     j=0;
     if (b==4)                // czwarte nacisniecie - pulsowanie diody 2 (zielona)
     {  /*for(i;i<t_on;i++)      // czas wlaczenia
        {P1OUT |=BIT6;}        // czas wylaczenia
        for(j;j<t_off;j++)
        {P1OUT &= ~BIT6;}*/

        P1OUT |=BIT6;
        _delay_cycles(300000);
        P1OUT &= ~BIT6;
        _delay_cycles(300000);


     }
     if (b==5)                // piate nacisniecie - pulsowanie diody 1 (czerwona)
     {   for(i;i<t_on;i++)      // czas wlaczenia
        {P1OUT |=BIT0;}
        for(j;j<t_off;j++)      // czas wylaczenia
        {P1OUT &= ~BIT0;}
     }
     if (b==6)                // szóste nacisniecie - pulsowanie obu diód synchronicznie
     {   for(i;i<t_on;i++)      // czas wlaczenia
        {P1OUT |=BIT6;P1OUT |=BIT0;}
        for(j;j<t_off;j++)      // czas wylaczenia
        {P1OUT &= ~BIT6;P1OUT &= ~BIT0;}
     }
     if (b==7)                // siódme nacisniecie - pulsowanie obu diód naprzemiennie
     {   for(i;i<t_on;i++)      // czas wlaczenia
        {P1OUT |=BIT6;P1OUT &= ~BIT0;}
        for(j;j<t_off;j++)      // czas wylaczenia
        {P1OUT &= ~BIT6;P1OUT |=BIT0;}
     }
     if (b==8)                // wylaczenie
     {  P1OUT &= ~BIT6; P1OUT &= ~BIT0; b=0;}

   }


              return 0;
}
```