Politechnika Opolska

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

PRZEDMIOT:	Technika mikroprocesorowa

KIERUNEK STUDIÓW:	***AUTOMATYKA I ROBOTYKA***	ROK STUDIÓW:	***3***
SPECJALNOŚĆ:	**-**		
SEMESTR:	**6**	ROK AKADEMICKI:	***2019/2020***

Temat ćwiczenia:

Wykorzystanie P1.3 do zmiany stanów diód na płytce MSP430

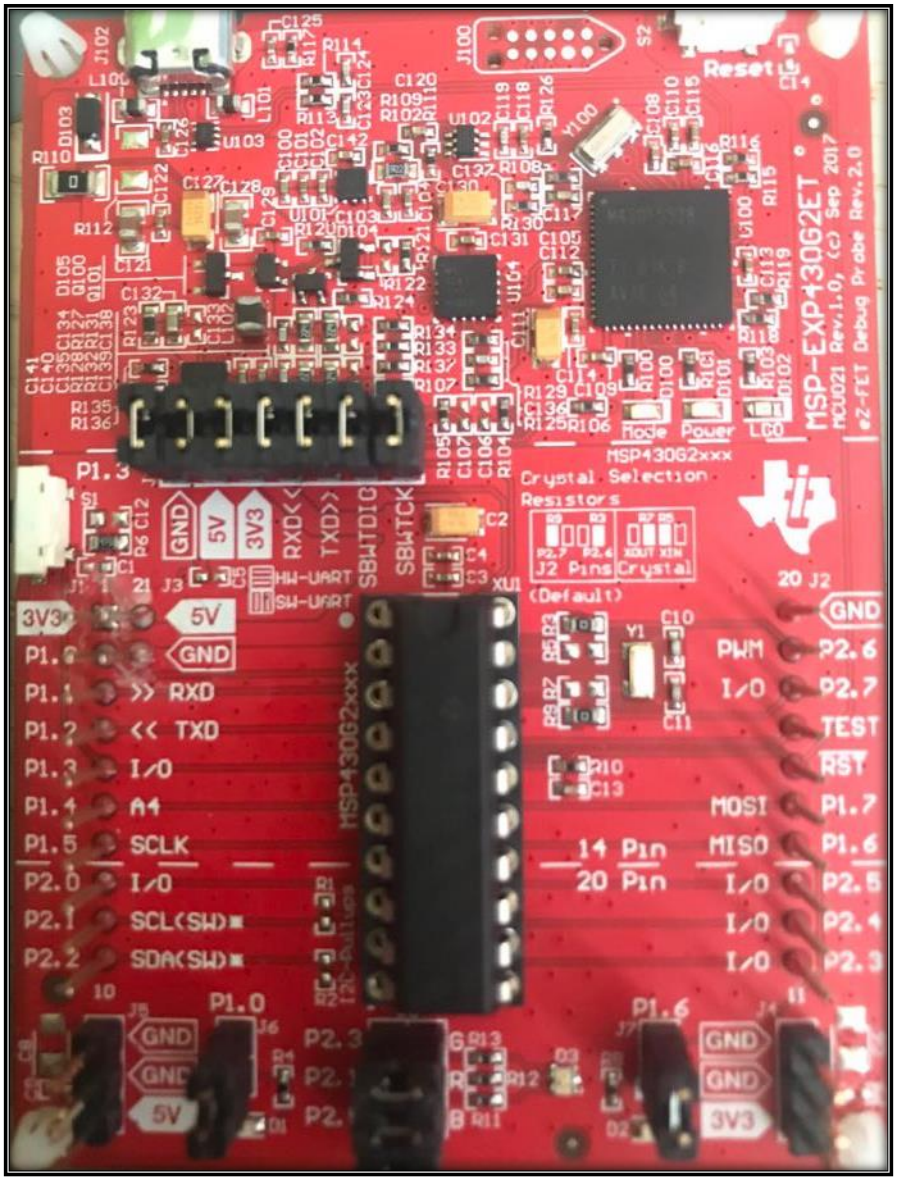
Ćwiczenie wykonali:					
Nazwisko:		*Imię:*	*Nazwisko:*		*Imię:*
1.	**Janas**	**Kamil**	**2.**	**Indyk**	**Dominik**
3.	**Dittrich**	**Sebastian**			

Uwagi:	*Data:*	*Ocena za sprawozdanie:*

Termin zajęć:					
Data:	07.03.2020	Dzień tygodnia:	SOBOTA	Godzina:	15:40

1. Cel ćwiczenia.

Celem ćwiczenia było użycie układu MSP430G2553 do zaprogramowania własnego programu oraz zaimplementowanie go do układu. Do stworzenia tego projektu wykorzystano oprogramowanie CodeComposer, który pozwala na podświetlanie składni oraz implementacje kodu do mikrokontrolera. Układ jest przedstawiony na zdjęciu 1.



Zdj.1. Mikrokontroler MSP430G2553 wykorzystany w ćwiczeniu.

2. Założenia ćwiczenia.

Program w założeniu dzięki umiejscowionemu na mikrokontrolerze przycisku P1.3 ma zmieniać kolory świecenia diód LED D1, D2, D3 w danej sekwencji określonej programowo. Do stworzenia sekwencji można użyć instrukcji warunkowych if lub switch case. W tym projekcie zastosowano drugą opcję dzięki której zaprogramowano 5 wariantów świecenia diód. Program jest opisany kodem w języku C przedstawiony poniżej:

```
#include <msp430.h>

#define SW BIT3          // Przycisk P1.3
#define LEDG BIT0         // Zielona dioda P1.0
#define LEDR BIT6        // Czerwona dioda P1.6

int x,i;
void main (void)
{
    WDTCTL = WDTPW | WDTHOLD;           // Stop watchdog timer

    P1DIR |= LEDR+LEDG;
    P2DIR |= BIT5+BIT3+BIT1;
    P1DIR &= ~SW;
    P1REN |= SW;
    P1OUT |= SW;

    x=1;

    while(1)
    {
        if(!(P1IN & SW))                // Jeśli przycisk jest wciśnięty
        {
            __delay_cycles(20000);      // Poczekaj 20ms
            if(!(P1IN & SW))            // Sprawdza czy przycisk jest dalej
                                        // wciśniety
            {                           // Ignoruj wciśnięcie, które jest krótsze
                                        // niż 20ms
                while(!(P1IN & SW));    // Jak puścimy przycisk
                x = x+1;0;              // Zmień wartość x
        }
        }



      switch(x){
        case 1:                         // Świeci się dioda P1.6 i P2.1

            P1OUT &= ~(LEDG+LEDR);      // Negacja diody P1.0 i P1.6
            P2OUT &= ~(BIT3+BIT5);      // Negacja diody P2.3 I P2.5
            P2OUT |= BIT1;              // Ustawienie logicznej 1 na P2.1 (dioda
                                        // czerwona)
            P1OUT |= LEDR;              // Ustawienie logicznej 1 na P1.6 (dioda
                                        // czerwona)
            break;
```

```
        case 2:                         // Świeci się dioda P1.0 i P2.3

             P1OUT &= ~LEDR;             // Negacja diody P1.6
             P2OUT &= ~BIT1;             // Negacja diody P2.1
             P1OUT |= LEDG;              // Ustawienie logicznej 1 na P1.0 (dioda
                                         // zielona)
             P2OUT |= BIT3;              // Ustawienie logicznej 1 na P2.3 (dioda
                                         // zielona)
             break;
         case 3:                         // Migająca dioda P2.5 (niebieska), stałe
                                         // świecenie P1.0 i P1.6
             P2OUT &= ~BIT3;             // Negacja diody P2.3
             P1OUT |=LEDR+LEDG;          // Ustawienie logicznej 1 na P1.0 i P1.6
             P2OUT |= BIT5;              // Ustawienie logicznej 1 na P2.5 (dioda
                                         // niebieska)
             for(i = 0; i<20000; i++);   // Opóźnienie zmiany stanu P2.5 20ms
             P2OUT &= ~BIT5;             // Negacja diody P2.5 (dioda niebieska)
             for(i = 0; i<20000; i++);   // Opóźnienie zmiany stanu P2.5 20ms
             break;
         case 4:                         // Miganie na przemienne diód czerwonych i
                                         // zielonych
                                         // P2.1 i P1.6 po zmianie P2.3 i P1.0
                                         // opóźnienie 20 ms
             P1OUT &= ~LEDR;             // Negacja diody P1.6 (czerwonej)
             P1OUT |= LEDG;              // Ustawienie logicznej 1 dla diody P1.0
                                         // (zielonej)
             P2OUT &= ~BIT1;             // Negacja diody P2.1 (czerwonej)
             P2OUT |= BIT3;              // Ustawienie logicznej 1 dla diody P2.3
                                         // (zielonej)
             for(i = 0; i<20000; i++);   // Opóźnienie zmiany stanu 20ms
             P1OUT &= ~LEDG;             // Negacja diody P1.0 (zielonej)
             P1OUT |= LEDR;              // Ustawienie logicznej 1 na diodę P1.6
                                         // (czerwonej)
             P2OUT &= ~BIT3;             // Negacja diody P2.3 (zielonej)
             P2OUT |= BIT1;              // Ustawienie logicznej 1 na diodę P2.1
                                         // (czerwonej)
             for(i = 0; i<20000; i++);   // Opóźnienie zmiany stanu 20ms
             break;
         case 5:
             x=0;                        // Reset instrukcji Switch case
             break;
         default:                        // Wyłączenie wszystkich diód
             P1OUT &= ~LEDG+LEDR;        // Negacja P1.0 i P1.6
             P2OUT &= ~BIT1+BIT3+BIT5;   // Negacja P2.1; P2.3; P2.5
         }
    }
}
```

3. Działanie programu.

Powyższy program działa w następującej sekwencji przy uruchomieniu mikrokontrolera ustawiana jest 1 logiczna na P1.6 oraz P2.1 co powoduje zapalenie się dwóch diód czerwonych. Przyciśnięcie przycisku P1.3 powoduje przejście do drugiego warunku, który kasuje poprzednie ustawienie oraz ustawia 1 logiczną na P1.0 i P2.3. Drugie wciśnięcie odpowiada trzeciemu warunkowi, który określa, że negowany jest P2.3, a ustawiany jest natomiast bit P1.6 i P2.5, gdzie dodatkowo ustawiono, że P2.5 (dioda niebieska) będzie migała z opóźnieniem 20ms. Kolejne kliknięcie przycisku spowoduje zanegowanie P2.5 oraz miganie naprzemienne dwóch kolorów P1.0 i P2.3 (diody zielone) oraz P1.6 i P2.1 (diody czerwone), opóźnienie zmian stanów wynosi 20ms. Ostanie kliknięcie powoduje kasowanie wszystkich stanów wszystkich diód, czyli ich wyłączenie. Po przejściu wszystkich sekwencji następne wciśnięcie przycisku spowoduje zresetowanie zliczanie warunków, dzięki czemu sekwencja będzie zliczana od nowa. Dodatkowo do programu został dodany warunek określający, że jeśli przycisk będzie wciśnięty zbyt krótko nie nastąpi przełączenie warunku, zostało to dodane w celu zabezpieczenia programu przed nieoczekiwanym rezultatem działania mikrokontrolera.

4. Wnioski podsumowujące.

Powyższe zastosowanie mikrokontrolera pozwala stwierdzić, że przy wykorzystaniu środowiska programistycznego oraz znajomości języka C, układ MSP430G2553 diody znajdujące się na płytce można zaprogramować dowolnie ustawiając częstotliwość migania lub kombinacje świecenia diód, a przy dodaniu przycisku można wedle uznania ustawiać wiele rozwiązań, przykładem takiego rozwiązania jest powyżej przedstawiony kod, lub można również zaprogramować płytkę w taki sposób, aby po kolejnych wciśnięciach przycisku P1.3 częstotliwość migania wybranej diody rosła lub malała.