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:
Zapoznanie się z układem MSP430G2

Ćwiczenie wykonali:					
Nazwisko:		*Imię:*	*Nazwisko:*		*Imię:*
1.	**Halek**	**Krzysztof**	**2.**	**Heisig**	**Henryk**
3.	**Pryszcz**	**Krzysztof**	**4.**		

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

Termin zajęć:					
Data:	7.03.2020	Dzień tygodnia:	Sobota	Godzina:	15:40

1. Cel ćwiczenia

Celem ćwiczenia było zapoznanie się ze środowiskiem programistycznym Code Composer Studio, które służy do tworzenia aplikacji dla wbudowanych procesorów Texas Instruments. Elementem wykonawczym programu jest mikrokontroler MSP430G2553. Program został napisany z użyciem języka programowania C.

2. Zakres ćwiczenia

Naszym zadaniem było napisanie programu, który ma polegać na obsłudze wejść oraz wyjść układu MSP430G2553. Program jest realizowany w nieskończonej pętli while. Polega on na zmianie kombinacji zapalonych diod LED po wciśnięciu przycisku. Każdorazowe wciśnięcie przycisku powoduje zwiększenie wartości zmiennej „a" o jeden. Jeżeli zmienna ta ma zdefiniowaną jakąś wartość na płytce zostają zaświecone diody o konfiguracji odpowiadającej wartości tej zmiennej. Liczba możliwych kombinacji wynosi 7. W celu poprawienia działania programu zastosowano instrukcje „for" aby nadać opóźnienie, które powoduje inkrementacje zmiennej „a" w taki sposób aby przy jednym naciśnięciu przycisku warunek był wykonywany tylko raz. Jeżeli wartość zmiennej jest większa lub równa 8 następuje zmiana jej wartości na 1.

3. Kod programu

```
1.  #include <msp430.h>

2.  #define SW  BIT3                    // definiujemy zmienną SW która
                                        będzie
                                        odpowiadała za Bit3. BIT3 odpowaida za
                                        fizyczny pin P1.3 na płytce prototypowej
3.  #define RED BIT3                    // Zdefiniujemy bit3 jako kolor
                                        czerwony diody RGB
4.  #define GRN BIT1                    // Zdefiniujemy bit1 jako kolor zielony
                                        diody RGB
5.  #define BLU BIT5                    // Zdefiniujemy bit5 jako kolor
                                        niebieski diody RGB

6.  unsigned int i,a;                   // rezerwujemy zmienną "i" oraz "a"jako
                                        unsigned integer

7.  void main(void) {
8.  WDTCTL = WDTPW | WDTHOLD;           // Zatrzymanie zegara watchdog'a

9.  P2DIR |= (RED+GRN+BLU);             // Ustawienie wyjść na pinach P2.3
                                        (Czerwona LED), P2.1 (Zielona LED),
                                        P2.5(Niebieska LED)
10. P1DIR &= ~SW;                        // Ustawienie wejścia na pinie P1.3
                                        (SW2)
11. P1REN |= SW;                        // włączenie rezoystora podciągającego
12. P1OUT |= SW;                        // ustawienie rezystora podciągającego
                                        jako pull-up

13. a=1;                                // zapisanie wartości 1 dla zmiennej "a"

14. while(1){                           // rozpoczęcie nieskończonej pętli
```

```
                                        programu, który przy każdym naciśnięciu
                                        przycisku SW2 inkrementuje wartość
                                        zmiennej "a"
15. if(!(P1IN & SW)){                   // wykonanie kodu jeżeli przycisk
                                        SW2 jest naciśnięty
16. a=a+1;                              // inkrementacja zmiennej "a"
17. for(i = 0; i<10000; i++);           // opóźnienie, aby przy jednym naciśnięciu
                                        inkrementacja zmiennej "a" była wykonana jeden
                                        raz
18. }

19. if(a==1){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 1
20. P2OUT |= GRN;
21. P2OUT |= RED;                       // ustawienie wyjść dla odpowiednich pinów
                                        do których jest podłączona dioda RGB
22. P2OUT &= ~BLU;                      // w tej konfiguracji zaświecony jest kolor
                                        czerwony i zielony
23. }

24. if(a==2){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 2
25. P2OUT &= ~GRN;
26. P2OUT |= RED;                       // ustawienie wyjść dla odpowiednich pinów
                                        do których jest podłączona dioda RGB
27. P2OUT |= BLU;                        // w tej konfiguracji zaświecony jest kolor
                                        niebieski i zielony
28. }

29. if(a==3){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 3
30. P2OUT |= GRN;
31. P2OUT |= RED;                       // ustawienie wyjść dla odpowiednich pinów
                                        do których jest podłączona dioda RGB
32. P2OUT |= BLU;                       // w tej konfiguracji zaświecony jest kolor
                                        czerwony, zielony i niebieski
33. }

34. if(a==4){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 4
35. P2OUT |= GRN;
36. P2OUT &= ~RED;                      // ustawienie wyjść dla odpowiednich pinów
                                        do których jest podłączona dioda RGB
37. P2OUT |= BLU;                        // w tej konfiguracji zaświecony jest kolor
                                        zielony i niebieski
38. }

39. if(a==5){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 5
40. P2OUT |= GRN;
41. P2OUT &= ~RED;                      // ustawienie wyjść dla odpowiednich pinów
                                        do których jest podłączona dioda RGB
42. P2OUT &= ~BLU;                      // w tej konfiguracji zaświecony jest kolor
                                        zielony
43. }

44. if(a==6){                           // warunek, który jest spełniony jeśli
                                        zmienna "a" jest równa 6
45. P2OUT &= ~GRN;
```

```
46. P2OUT |= RED;                    // ustawienie wyjść dla odpowiednich
                                     pinów do których jest podłączona dioda
                                     RGB
47. P2OUT &= ~BLU;                   // w tej konfiguracji zaświecony jest
                                     Kolor czerwony
48. }

49. if(a==7){                        // warunek, który jest spełniony jeśli
                                     zmienna "a" jest równa 7
50. P2OUT &= ~GRN;
51. P2OUT &= ~RED;                   // ustawienie wyjść dla odpowiednich
                                     pinów
                                     do których jest podłączona dioda RGB
52. P2OUT |= BLU;                    // w tej konfiguracji zaświecony jest
                                     kolor niebieski
53. }

54. if(a>=8){                        // Jeśli zmienna "a" jest większa lub
                                     równa 8
55. a = 1;                           // następuje ustawienie wartości zmiennej
                                     "a" na 1
56. }
57. }
```

4. Wnioski

Za pomocą środowiska Code Composer Studio można z łatwością nawiązać komunikację z mikroprocesorem MSP430G2553 i wgrać programy napisane w języku „C”. Napisany pierwszy program do obsługi GPIO nie sprawił problemu, a uruchomiony kod działa poprawnie. Dzięki wbudowanym rezystorom podciągającym „Pull Up” można było skonfigurować przycisk jako sygnał wejściowy. Skonfigurowanie odpowiednich pinów jako wyjście pozwalało na zapalanie odpowiednich kolorów wbudowanej diody RGB. Natknęliśmy się jednak na przeszkodę typu zbyt szybką inkrementację zmiennej „a” przez co dołożyliśmy opóźnienie zrealizowanie na pętli „for”.