|  |  |
| --- | --- |
| logo_PO | **Politechnika Opolska** |

**LABORATORIUM**

|  |
| --- |
| **Technika Mikroprocesorowa** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Kierunek studiów: | AiR Ns | | Rok studiów: | | III |
| Semestr: | VI | Rok akademicki: | | 2019/2020 | |

|  |
| --- |
| *Temat ćwiczenia:* |
| Programowanie płytki MSP-EXP430G2 |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| *Projekt wykonali:* | | | |
| *Nazwisko i imię:* | | *Nazwisko i imię:* | |
| **1.** | Szymon Słaboń | **2.** | Rychel Konrad |
| **3.** | Syguła Dariusz | **4.** |  |

|  |  |  |
| --- | --- | --- |
| *Ocena:* | *Data:* | *Uwagi:* |
|  |  |  |

**Wstęp**

Postawione przed nami zadanie polegało a napisaniu kodu do mikrokontrolera MSP-EXP430G2 firmy Texas Instruments. Działanie programu zostało ściśle określone i należało się go trzymać.

**Opis programu**

Po uruchomieniu programu i wciśnięciu przycisku zaświeca się dioda, która miga z określoną częstotliwością. Po upływie pewnego czasu gaśnie i zaświeca się czerwona, a następnie znowu znów zaświeca się zielona, a czerwona gaśnie. Trwa to do momentu, w którym wewnętrzny licznik nie osiągnie wartości 5, która to wartość zmienia się przy każdej zmianie świecenia diod. Po osiągnięciu tej wartości dioda zielona gaśnie, a czerwona miga dwukrotnie po czym gaśnie.

Poniższy program oparty jest na instrukcjach przerwania. Na początku zdefiniowane zostały diody i przycisk. Diody zostały wyłączone na samym początku, a do przycisku została dołączona flaga przerwania, przerwanie to będzie wywoływane po „puszczeniu” wciśniętego przycisku. Następnie zadeklarowaliśmy taimery, określiliśmy ich kierunek tzn. mają odliczać w górę, oraz ilość tankowań czyli do jakiej wartości będą zliczały te timery. Należało jeszcze przypisać do nich wykonywanie przerwań.

Teraz przejdźmy do głównej części programu, pierwsza instrukcja przerwania jest wywoływana przez przycisk i zawarte są w niej tylko dwie instrukcje, pierwsza to uruchomienie timera, a właściwie przerwania uruchamianego przez ten timer. Druga instrukcja przerywania (druga w kolejności wykonywania, w programie umieszczona została na samym końcu) zawiera właśnie instrukcje warunkowe odpowiedzialne za kolejność zaświecania diod. Zastosowano tu dwie zadeklarowane wcześniej zmienne, zmienna takt odpowiada za zliczanie ile razy wykonała się instrukcja, a licznik jak sama nazwa wskazuje jest tym licznikiem inkrementowanym z każdym przełączeniem diody. Po określonej liczbie taktowań, sprawdzana jest wartość licznika. Przy wartości 0 czyli przy pierwszym uruchomieniu załączana jest dioda zielona, później załączane są one w opisany wcześniej sposób. Kiedy licznik osiąga wartość 5 diody są gaszone licznik i takt zerowane, podobnie jak flaga przerwania timera 0 i załączone zostaje przerwanie timera 1.

Ostatnia instrukcja przerwania odpowiada za dwukrotne mignięcie diody czerwonej przed zakończeniem działania programu. Tutaj też zmienna takt zlicza ilość wykonanych przerwań, instrukcja sprawdzająca bada czy upłynęła już odpowiednia ilość taktów, jeżeli tak pętla for czterokrotnie zmienia stan na diodzie czerwonej (z wyłączonej na włączoną). Po wyjściu z pętli dioda czerwona zostaje całkowicie wyłączona, następuje zatrzymanie timera 1 i zakończenie funkcji przerwania.

**Skrypt programu:**

|  |
| --- |
| #include "msp430g2553.h"  #define RED\_LED BIT6 // Red LED -> P1.6  #define GREEN\_LED BIT0 // Green LED -> P1.0  #define SW BIT3 // Switch -> P1.3  void main(void){  WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer  // Ustawianie diod i przycisku  P1DIR |= RED\_LED;  P1OUT &= ~RED\_LED;  P1DIR |= GREEN\_LED;  P1OUT &= ~GREEN\_LED;  P1DIR &= ~SW;  P1REN |= SW; // wlaczenie rezystora podciagajacego  P1OUT |= SW; // ustawienie rezystora podciagajacego jako pull-up    P1IES &= ~SW; // Wybranie wybranie puszczenia przycisku jako wyzwalanie przerwania  P1IE |= SW; // Wlczenie przerwania na przycisku  TA0CCTL0 |= CCIE; // Ustawienie wywo³ania przerwania timera0 po odliczeniu ustawionego czasu - timer 0  TA0CTL |= TASSEL\_2 + TACLR + ID\_3; // Ustawienie liczenia - "w gore", Zegar -> ACLK, Zerowanie stanu czasu - timer 0  TA1CCTL0 |= CCIE; // Ustawienie wywo³ania przerwania timera1 po odliczeniu ustawionego czasu - timer 1  TA1CTL |= TASSEL\_2 + TACLR + ID\_3; // Ustawienie liczenia - "w gore", Zegar -> ACLK, Zerowanie stanu czasu - timer 1  TA0CCR0 = 37500; // Ustawienie Timera 0 na 3 sekundy  TA1CCR0 = 12500; // Ustawienie Timera 1 na 1 sekunde  \_\_bis\_SR\_register(LPM0\_bits + GIE); // Wejscie w tryb LPM3 (Low-power Mode) oraz wlaczenie przerwania CPU  }  unsigned int licznik=0; // deklaracja zmiennej licznik jako integer  unsigned int takt=0; // deklaracja zmiennej licznik jako takt  unsigned int y; // deklaracja zmiennej y  // Przerwanie wykonane przez uruchomienie przycisku  #pragma vector=PORT1\_VECTOR  \_\_interrupt void Port\_1(void){  TACTL |= MC\_1; // Uruchomienie przerwania timera 0  P1IFG &= ~SW; // Czyszczenie flagi przerwania na przycisku  }  // Przerwanie wykonane przez timer 1  #pragma vector = TIMER1\_A0\_VECTOR // Przerwanie po odliczeniu czasu timera 1  \_\_interrupt void CCR1\_ISR(void){  takt++; // inkrementacja zmiennej zliczajacej ilosc taktowan  if (takt==10){  for(y=0; y<3; y++){ // wykonanie 4 zmian stanu diody czerownej  P1OUT ^= RED\_LED; // zmiana stanu diody czerwonej  \_\_delay\_cycles(100000); // opóznienie wykonania petli for  }  P1OUT &= ~RED\_LED; // zmiana stanu diody czerwonej  TA1CTL &= ~MC\_3; // zatrzymanie wykonania timera 1  \_\_bic\_SR\_register\_on\_exit( LPM4\_bits ); // zakończenie funkcji przerwania  }  }  // Przerwanie wykonane przez timer 0  #pragma vector = TIMER0\_A0\_VECTOR // Przerwanie po odliczeniu czasu timera 0  \_\_interrupt void CCR0\_ISR(void){  takt++; // inkrementacja zmiennej zliczajacej ilosc taktowan  if(takt==10){ // wykonanie kodu po zliczeniu 10 przerwan  if (licznik==0)  {  P1OUT |= GREEN\_LED; // zapalenie diody zielonej  }    else if (licznik!=5)  {  P1OUT ^= GREEN\_LED; // przelaczenie stanu diody zielonej  P1OUT ^= RED\_LED; // przelaczenie stanu diody czerwonej  }    else if(licznik==5)  {  licznik=0; // ustawienie wartsci zmiennej licznik na "0"  TA0CTL &= ~MC\_3; // Zatrzymanie timera 0  TA1CTL |= MC\_1; // Uruchomienie timera 1  P1OUT &= ~GREEN\_LED; // wylaczenie led zielonej  P1OUT &= ~RED\_LED; // wylazenie led czerwonej  takt=0;  }  licznik++; // inkrementacja wartosci zmiennej licznik  takt=0;  }  } |