Systemy Wbudowane

Sprawozdanie

Paweł Idzikowski
Praca w duecie z Dorota Gil

26 maja 2019

Spis treści

1 Zadanie 1

Treść zadania:
Stworzyć program "z dziewięcioma podprogramami". Przemieszczanie między programami ma się odbywać za pomocą RD6(przejście do przodu) i RD13(przejście do tyłu).

- Licznik binarny 8 bitowy zliczający w górę
- Licznik binarny 8 bitowy zliczający w dół
- Licznik bitowy w kodzie Graya zliczający w górę
- Licznik bitowy w kodzie Graya zliczający w dół
- Licznik 2x4 bit - BCD 0-99 zliczający w górę
- Licznik 2x4 bit - BCD 0-99 zliczający w dół
- Wężyk 3-diodowy w lewo
- Wężyk 3-diodowy w prawo
- Pseudogenerator liczb losowych PRNG 6 bitowy

Podsumowanie:
Pierwsze podejście do programowania mikrokontrolera. Udało nam się zrealizować 8 podprogramów. Do dziewiątego nie wiedzieliśmy jak podejść - brakowało pomysłu. Najwięcej czasu zajęło nam rozwiązanie zadań z licznikiem BCD - Dorota znalazła podpowiedzi jak zrobić taki licznik na stronie stackoverflow. Implementacja wraz z komentarzami znajduje się poniżej.

```
#include <p24fj128ga010.h>

//Konfiguracja dla Explorer 16 z progr. icd2
_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_OFF & COE_OFF & FWDTEN_OFF)
_CONFIG2(FCKSM_CSDCMD & OSCIOFNC_ON & POSCMOD_HS & FNOSC_PRI)

# define SCALE 600 L

int main(void) {
  unsigned long i;
  unsigned char display = 0;
  unsigned int liczba = 0; // przechowywanie aktualnej wartosci
  unsigned int jednosci = 0; // do licznika BCD
  unsigned int dziesiatki = 0; // do licznika BCD
```

```
unsigned int temp = 0; // przechowywanie tymczasowych danych

int numerProgramu = 0; // do obslugi zmiany programu

//inicjalizacja
PORTA = 0x0000;
TRISA = 0xFF00;
TRISD = 0xFFFF;

//niekonczaca sie petla
again:
  Nop();
PORTA = (unsigned int) display;

for (i = 0; i < 500 L * SCALE; i++) Nop();

// ****************************************** //
// obsluga zmiany programu

// jezeli wcisnieto RD6 i numerProgramu jest mniejszy od 9
if (PORTDbits.RD6 == 0 && numerProgramu < 9) {
  // zwieksz numerProgramu o 1
  numerProgramu = numerProgramu + 1;
// lub jesli wcisnieto RD13 i numerProgramu jest wiekszy od 0
} else if (PORTDbits.RD13 == 0 && numerProgramu > 0) {
  // zmniejsz numerProgramu o 1
  numerProgramu = numerProgramu - 1;
// lub jesli wcisnieto RD6 i numerProgramu osiagnal wartosc 9
} else if (PORTDbits.RD6 == 0 && numerProgramu == 9) {
  // ustaw numerProgramu na 0
  numerProgramu = 0;
// lub jesli wcisnieto RD13 i numerProgramu osiagnal wartosc 0
} else if (PORTDbits.RD13 == 0 && numerProgramu == 0) {
  // ustaw numerProgramu na 8
  numerProgramu = 8;
}

// ****************************************** //

// jezeli wartosc numerProgramu jest 0
// program1
if (numerProgramu == 0) {
  // zwieksz biezaca wartosc display o 1
  display = display + 1;
}
```

```
// lub jezeli wartosc numerProgramu jest 1
// program2
else if (numerProgramu == 1) {
  // zmniejsz biezaca wartosc display o 1
  display = display - 1;
}

// lub jezeli wartosc numerProgramu jest 2
// program3
else if (numerProgramu == 2) {
  // przypisz pod PORTA wartosc spod zmiennej liczba
  PORTA = liczba;
  // wykonaj na wartosci spod zmiennej liczba przesuniecie bitowe w prawo o 1
  // a nastepnie operacje XOR miedzy wartoscia zm. liczba a uzyskanym wynikiem
  display = liczba ^ (liczba >> 1);
  // zwieksz biezaca wartosc liczba o 1
  liczba = liczba + 1;
}

// lub jezeli wartosc numerProgramu jest 3
// program4
else if (numerProgramu == 3) {
  // przypisz pod PORTA wartosc spod zmiennej liczba
  PORTA = liczba;
  // wykonaj na wartosci spod zmiennej liczba przesuniecie bitowe w prawo o 1
  // a nastepnie operacje XOR miedzy wartoscia zm. liczba a uzyskanym wynikiem
  display = liczba ^ (liczba >> 1);
  // zmniejsz biezaca wartosc liczba o 1
  liczba = liczba - 1;
}

// lub jezeli wartosc numerProgramu jest 4
// program5
else if (numerProgramu == 4) {
  // jezeli wartosc zm. liczba osiagnela 99
  if (liczba == 99)
    // przypisz pod zm. liczba wartosc 0
    liczba = 0;

  // reszte z dzielenia liczba przez 10 przypisz do zm. jednosci
  jednosci = liczba % 10;
  // wynik dzielenia liczba przez 10 przypisz do zm. dziesiatki
  dziesiatki = liczba / 10;
  // przypisz do zm. temp wynik przesuniecia bitowego
  // w lewo o 4 wartosci spod zm. dziesiatki
  temp = dziesiatki << 4;
```

```
  // przypisz do zm. display wynik sumy temp + jednosci
  display = temp + jednosci;
  // zwieksz biezaca wartosc liczby o 1
  liczba = liczba + 1;
}

// lub jezeli wartosc numerProgramu jest 5
// program6
else if (numerProgramu == 5) {
  // jezeli wartosc zm. liczba osiagnela 0
  if (liczba == 0)
    // przypisz pod zm. liczba wartosc 99
    liczba = 99;

  // reszte z dzielenia liczba przez 10 przypisz do zm. jednosci
  jednosci = liczba % 10;
  // wynik dzielenia liczba przez 10 przypisz do zm. dziesiatki
  dziesiatki = liczba / 10;
  // przypisz do zm. temp wynik przesuniecia bitowego
  // w lewo o 4 wartosci spod zm. dziesiatki
  temp = dziesiatki << 4;
  // przypisz do zm. display wynik sumy temp + jednosci
  display = temp + jednosci;
  // zmniejsz biezaca wartosc liczby o 1
  liczba = liczba - 1;
}

// lub jezeli wartosc numerProgramu jest 6
// program7
else if (numerProgramu == 6) {
  // jezeli wartosc zm. liczba wynosi 0
  if (liczba == 0)
    // przypisz zmiennej liczba wartosc 1
    liczba = 1;
  // lub jesli wartosc zm. liczba wynosi 1
  else if (liczba == 1)
    // przypisz zmiennej liczba wartosc 3
    liczba = 3;
  // lub jesli wartosc zm. liczba wynosi 3
  else if (liczba == 3)
    // przypisz zmiennej liczba wartosc 7
    liczba = 7;
  // a jezeli zadne z powyzszych to
  else {
    // przypisz zmiennej liczba wynik operacji liczba * 2
    liczba = liczba * 2;
```

```
  }

  // przypisz do zm. display wartosc zm. liczba
  display = liczba;

  // jezeli wartosc zm. liczba jest wieksza od 1000
  if (liczba > 1000)
    // przypisz do zmiennej liczba wartosc 0
    liczba = 0;
}

// lub jezeli wartosc numerProgramu jest 7
// program8
else if (numerProgramu == 7) {
  // jezeli wartosc zm. liczba wynosi 255
  if (liczba == 255)
    // przypisz zmiennej liczba wartosc 128
    liczba = 128;
  // lub jesli wartosc zm. liczba wynosi 128
  else if (liczba == 128)
    // przypisz zmiennej liczba wartosc 192
    liczba = 192;
  // lub jesli wartosc zm. liczba wynosi 192
  else if (liczba == 192)
    // przypisz zmiennej liczba wartosc 112
    liczba = 112;
  // a jezeli zadne z powyzszych to
  else {
    // przypisz do zm. liczba wynik operacji przesuniecia
    // bitowego wartosci spod zm. liczba w prawo o 1
    liczba = liczba >> 1;
  }

  // przypisz do zm. display wartosc zm. liczba
  display = liczba;

  // jezeli wartosc zm. liczba jest mniejsza bądz rowna zeru
  if (liczba <= 0)
    // przypisz zmiennej liczba wartosc 255
    liczba = 255;
}

// lub jezeli wartosc numerProgramu jest 8
// program9
else if (numerProgramu == 8) {
  // PRNG (brak)
```

```
  }

  goto again;
}
```

2 Zadanie 2

Treść zadania:
W oparciu o ćwiczenie 4c zrobić czujnik temperatury. Jeżeli temperatura >25 stopni Celsjusza (nastawa pierwotna) to przez pierwsze 3 sekundy miga pojedyncza dioda po czym zapalają się wszystkie. RD6 - zmiana nastawy na bieżącą temperature + 1. RD13 - wyłącza alarm.

Podsumowanie:
Największym problemem w tym zadaniu było aby je zacząć. Nie wiedzieliśmy jak ze zmiennej temp odczytać wartość temperatury. Znaleźliśmy na stronie arudino wzorek aby odczytać ADC i w praktyce odczyt był poprawny. Na jednych ćwiczeniach prowadzący zauważył ten schemat i podpowiedział, że aby odczytać temperaturę wystarczy podzielić wartość temp przez 10 i zrzutować na double więc to zamieniliśmy. Z dalszym etapem zadania sobie poradziliśmy. Implementacja kodu wraz z komentarzem znajduje się poniżej.

```
#include <p24fj128ga010.h>

_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_OFF & COE_OFF & FWDTEN_OFF)
_CONFIG2(FCKSM_CSDCMD & OSCIOFNC_ON & POSCMOD_HS & FNOSC_PRI)

# define SCALE 308 L

/* numer którym dostajemy się do czujnika temperatury */

# define TSENS 4# define AINPUTS 0xffcf
void ADCinit(int amask)

{
  AD1PCFG = amask;
  AD1CON1 = 0x00e0;
  AD1CSSL = 0;
  AD1CON2 = 0;
  AD1CON3 = 0x1f02;
  AD1CON1bits.ADON = 1;
}

int readADC(int ch)
```

```
{
  AD1CHS = ch;
  AD1CON1bits.SAMP = 1;
  while (!AD1CON1bits.DONE);
  return ADC1BUF0;
}

// funkcja aby opoznic dzialanie programu
// przy etapie migania diod
void Delay() {
  unsigned long i;

  for (i = 100000 L; i > 0; i--) {};
}

int main(void) {
  TRISA = 0xFF00;
  TRISD = 0xFFFF;

  unsigned long i;
  unsigned char display = 0;
  int temp;
  unsigned int szybkosc = 500 L;

  // temperatura poczatkowa na ktora mamy reagowac alarmem
  unsigned int nastawa = 25;

  // kontrolka do przelaczania alarmu
  unsigned int alarm = 0;

  // zmienna ktora kontroluje wykonanie migania diody
  unsigned int przelaczDiody = 0;

  double wynik;
  PORTA = 0x0000;
  ADCinit(AINPUTS); /*inicjalizacja konwertera AD*/

  display = 1;

  while (1) {

    Nop();

    PORTA = display;
```

```
for (i = szybkosc * SCALE; i > 0; i--) Nop();

temp = readADC(TSENS);

// wykonaj iloraz temp przez 10 i zrzutuj do double
// wynik operacji przechowaj w zmiennej wynik
wynik = temp / (double) 10;

// przypisz do zmiennej display wartosc zm. wynik
display = wynik;

// jezeli obecna temperatura wieksza niz prog
if (wynik > nastawa) {
  // ustaw zmienna alarm na 1
  alarm = 1;
}
// lub jezeli biezacy odczyt temperatury
// jest mniejszy badz rowny od nastawy
// to wylacz alarm i zresetuj przelaczDiody
else if (wynik <= nastawa) {
  alarm = 0;
  przelaczDiody = 0;
}

// jezeli wcisnieto przycisk RD6
if (PORTDbits.RD6 == 0) {
  // przypisz do nastawa wynik sumy
  // biezaca temperatura(wynik) + 1
  nastawa = wynik + 1;
}
// lub jezeli wcisnieto przycisk RD7
// wlacz alarm recznie
else if (PORTDbits.RD7 == 0) {
  // ustaw alarm na 1
  alarm = 1;
}
// lub jezeli wcisnieto przycisk RD13
// wylacz alarm recznie
else if (PORTDbits.RD13 == 0) {
  // ustaw zmienna alarm na 0
  alarm = 0;
  // ustaw zmienna przelaczDiody na 0
  przelaczDiody = 0;
}

// jezeli wartosc alarm wynosi 1
```

```
if (alarm == 1) {
  // jezeli wartosc przelaczDiody jest ustawiona na 0
  if (przelaczDiody == 0) {

    // wywolaj Delay w celu opoznienia iteracji
    Delay();

    // wywolaj Delay w celu opoznienia iteracji
    Delay();

    // odczyt temperatury
    temp = readADC(TSENS);
    wynik = temp / (double) 10;

    // sprawdz czy wynik(temperatura) <= nastawa
    // - jezeli tak to przerwij etap migania
    if (wynik <= nastawa) {
      continue;
    }

    // ustaw wartosc 1 pod zm. display
    display = 1;
    // przypisz display do PORTA
    PORTA = display;

    // wywolaj Delay w celu opoznienia iteracji
    Delay();

    // odczyt temperatury
    temp = readADC(TSENS);
    wynik = temp / (double) 10;

    // jezeli wynik <= nastawa przerwij
    // etap migania alarmu
    if (wynik <= nastawa) {
      continue;
    }

    // przypisz do display wartosc 0
    display = 0;
    // przypisz pod zm. PORTA wartosc zm. display
    PORTA = display;

    // wywolaj Delay w celu opoznienia iteracji
    Delay();
```

```
// odczyt temperatury
temp = readADC(TSENS);
wynik = temp / (double) 10;

// jezeli wynik(biezaca temperatura) jest
// mniejsza od nastawa przerwij procedure
// migania
if (wynik <= nastawa) {
  continue;
}

// przypisz do display wartosc 1
display = 1;
// przypisz do PORTA wartosc zm. display
PORTA = display;

// wywolaj Delay w celu opoznienia iteracji
Delay();

// odczyt temperatury
temp = readADC(TSENS);
wynik = temp / (double) 10;

// jezeli wynik(biezaca temperatura) jest
// mniejsza od nastawa przerwij procedure
// migania
if (wynik <= nastawa) {
  continue;
}

// przypisz do display wartosc 0
display = 0;
// przypisz do PORTA wartosc zm. display
PORTA = display;

// wywolaj Delay w celu opoznienia iteracji
Delay();

// odczyt temperatury
temp = readADC(TSENS);
wynik = temp / (double) 10;

// jezeli wynik(biezaca temperatura) jest
// mniejsza od nastawa przerwij procedure
// migania
if (wynik <= nastawa) {
```

```
          continue;
        }

        // przypisz do display wartosc 1
        display = 1;
        // przypisz do PORTA wartosc zm. display
        PORTA = display;

        // wywolaj Delay w celu opoznienia iteracji
        Delay();

        // ustaw przelaczDiody na 1
        przelaczDiody = 1;
      // lub jesli przelaczDiody jest ustawione na 1
      } else if (przelaczDiody == 1) {
        // odczyt temperatury
        temp = readADC(TSENS);
        wynik = temp / (double) 10;

        // jezeli wynik(biezaca temperatura) jest
        // mniejsza od nastawa przerwij procedure
        // alarmu
        if (wynik <= nastawa) {
          continue;
        }

        // przypisz do display wartosc 255
        display = 255;
        // przypisz do PORTA wartosc zm. display
        PORTA = display;
      }
    }
  }
}
```

3 Zadanie 3

Treść zadania:
Sterowanie prędkością zmiany stanu licznika przy pomocy potencjometru R6 wykorzystując dwa programy wykonane w zadaniu pierwszym. RD6 przełącza program do przodu, RD13 do tyłu.

Podsumowanie:
Zadanie 3 miało należeć do najłatwiejszych - i tak było... - ale my się dowiedzie-

liśmy o tym trochę później. Przechodząc do niego nie wiedzieliśmy czy potencjometr R6 to na pewno to 'pokrętło' bo jego przemieszczanie nie dawało żadnych zmian... Później dowiedzieliśmy się od prowadzącego, że z racji, że sprzęt nie jest już 'pierwszej świeżości' to trzeba nacisnąć i przytrzymać potencjometr aby zamknąć obwód i aby wartość z potencjometru została załapana. Jak już to wiedzieliśmy to dalsza implementacja poszła bez przeszkód. Kod z komentarzami umieszczony został poniżej.

```
#include <p24fj128ga010.h>

_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_OFF & COE_OFF & FWDTEN_OFF)
_CONFIG2(FCKSM_CSDCMD & OSCIOFNC_ON & POSCMOD_HS & FNOSC_PRI)

/* numer ktÃ?rym dostajemy siÄ? do potencjometru */
# define TVOLT 5# define AINPUTS 0xffcf

void ADCinit(int amask) {
  AD1PCFG = amask;
  AD1CON1 = 0x00e0;
  AD1CSSL = 0;
  AD1CON2 = 0;
  AD1CON3 = 0x1f02;
  AD1CON1bits.ADON = 1;
}

int readADC(int ch) {
  AD1CHS = ch;
  AD1CON1bits.SAMP = 1;
  while (!AD1CON1bits.DONE);
  return ADC1BUF0;
}

int main(void) {
  unsigned long SCALE = 308 L;
  unsigned long SCALEprogram = 308 L;
  unsigned long i;
  unsigned char display = 0;
  unsigned int liczba = 0;
  int numerProgramu = 0;
  int volt;

  //maksymalna wartoĹ?Ä? jakÄ? moĹźna odczytaÄ? to ok 1123. na jej
  //podstawie przeskalujemy wynik
  int maxOdczyt = 1123;

  //inicjalizacja
```

```
PORTA = 0x0000;
TRISA = 0xFF00;
TRISD = 0xFFFF;
ADCinit(AINPUTS); //inicjalizacja konwertera AD

while (1) {
  Nop();
  PORTA = (unsigned int) display;

  for (i = SCALEprogram * SCALE; i > 0; i--) Nop();

  volt = readADC(TVOLT); //Odczytanie potencjometru

  // ****************************************** //
  // obsluga zmiany programu

  // jezeli wcisnieto RD6 i numerProgramu przechowuje wartosc mniejsza od 4
  if (PORTDbits.RD6 == 0 && numerProgramu < 4) {
    // zwieksz biezaca wartosc numerProgramu o 1
    numerProgramu = numerProgramu + 1;
  // jezeli wcisnieto RD13 i numerProgramu przechowuje wartosc wieksza od 0
  } else if (PORTDbits.RD13 == 0 && numerProgramu > 0) {
    // zmniejsz biezaca wartosc numerProgramu o 1
    numerProgramu = numerProgramu - 1;
  }

  // ****************************************** //
  // obsluga zmiany predkosci

  // jezeli wartosc pod zm. volt jest >= maxOdczyt
  if (volt >= maxOdczyt) {
    // przypisz pod SCALEprogram wartosc 300 L
    SCALEprogram = 300 L;
  // lub jezeli wartosc volt jest mniejsza od 200
  } else if (volt < 200) {
    // przypisz pod SCALEprogram wartosc 500 L
    // ok. 0,5s
    SCALEprogram = 500 L;
  // lub jezeli wartosc volt jest wieksza od 200
  // i jednoczesnie mniejsza od 350
  } else if (volt > 200 && volt < 350) {
    // przypisz pod SCALEprogram wartosc 2000 L
    // ok. 2 sekundy
    SCALEprogram = 2000 L;
  // lub jezeli wartosc volt jest wieksza od 350
  // i jednoczesnie mniejsza od 450
```

```
} else if (volt > 350 && volt < 450) {
  // przypisz pod SCALEprogram wartosc 5000 L
  // ok. 5 sekund
  SCALEprogram = 5000 L;
// lub jezeli wartosc volt jest wieksza od 450
// i jednoczesnie mniejsza od 600
} else if (volt > 450 && volt < 600) {
  // przypisz pod SCALEprogram wartosc 8000 L
  // ok. 8 sekund
  SCALEprogram = 8000 L;
// lub jezeli wartosc volt jest wieksza od 600
// i jednoczesnie mniejsza od 800
} else if (volt > 600 && volt < 800) {
  // przypisz pod SCALEprogram wartosc 10000 L
  // ok. 10 sekund
  SCALEprogram = 10000 L;
// lub jezeli wartosc volt jest wieksza od 800
// i jednoczesnie mniejsza badz rowna 1122
} else if (volt > 800 && volt <= 1122) {
  // przypisz pod SCALEprogram wartosc 15000 L
  // ok. 15 sekund
  SCALEprogram = 15000 L;
}

// jezeli wartosc numerProgramu osiagnela 4
if (numerProgramu == 4) {
  // ustaw numerProgramu na 0
  numerProgramu = 0;
// lub jezeli wartosc numerProgramu osiagnela -1
} else if (numerProgramu == -1) {
  // ustaw numerProgramu na 3
  numerProgramu = 3;
}
// ************************************** //

// jezeli numerProgramu wynosi 0
// program1
if (numerProgramu == 0) {
  // zwieksz biezaca wartosc display o 1
  display = display + 1;
}

// lub jezeli numerProgramu wynosi 1
// program2
else if (numerProgramu == 1) {
  // zmniejsz biezaca wartosc display o 1
```

```
      display = display - 1;
    }

    // lub jezeli numerProgramu wynosi 2
    // program3
    else if (numerProgramu == 2) {
     // przypisz pod PORTA wartosc spod zmiennej liczba
     PORTA = liczba;
     // wykonaj na wartosci spod zmiennej liczba przesuniecie bitowe w prawo o 1
     // a nastepnie operacje XOR miedzy wartoscia zm. liczba a uzyskanym wynikiem
     display = liczba ^ (liczba >> 1);
     // zwieksz biezaca wartosc liczba o 1
     liczba = liczba + 1;
    }

    // lub jezeli numerProgramu wynosi 3
    // program4
    else if (numerProgramu == 3) {
     // przypisz pod PORTA wartosc spod zmiennej liczba
     PORTA = liczba;
     // wykonaj na wartosci spod zmiennej liczba przesuniecie bitowe w prawo o 1
     // a nastepnie operacje XOR miedzy wartoscia zm. liczba a uzyskanym wynikiem
     display = liczba ^ (liczba >> 1);
     // zmniejsz biezaca wartosc liczba o 1
     liczba = liczba - 1;
    }
  }

  return 0;
}
```

4 Zadanie 4

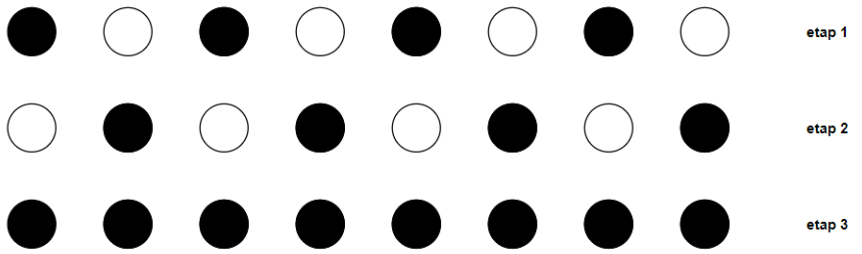
Treść zadania:
Kontroler lampek choinkowych (przynajmniej 6 układów)

Nie byliśmy niestety tymi szczęściarzami, którzy dostali mikrokontroler z działającym ekranem więc przeszliśmy do zadania z kontrolerem lampek choinkowych. Zanim zaczęliśmy cokolwiek umieszczać w kodzie to zaprojektowaliśmy koncepty, które chcielibyśmy wyświetlić na diodach.

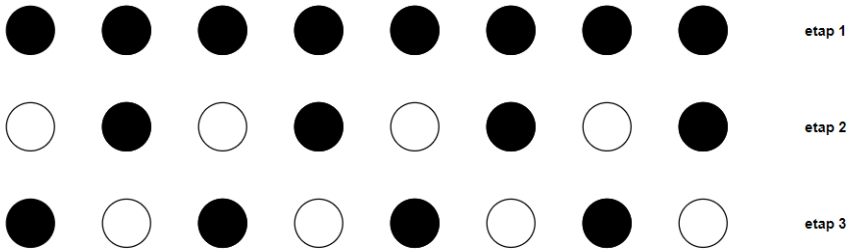
Schematy zaprezentowane są poniżej. Koła w czarnym tle reprezentują uruchomione diody w danym etapie pokazywania schematu natomiast koła z białym tłem - wyłączone. Z racji, że nie udało nam się zrobić PRNG, koncepty nie będą uruchamiane w sposób losowy. Jest natomiast alternatywa w postaci

iteracji. Po określonej w kodzie liczbie iteracji nastąpi zmiana schematu, który będzie pokazywany. Nie jest to może ten wyczekiwany efekt ale chcieliśmy aby po określonym czasie nastąpiła zmiana schematu.

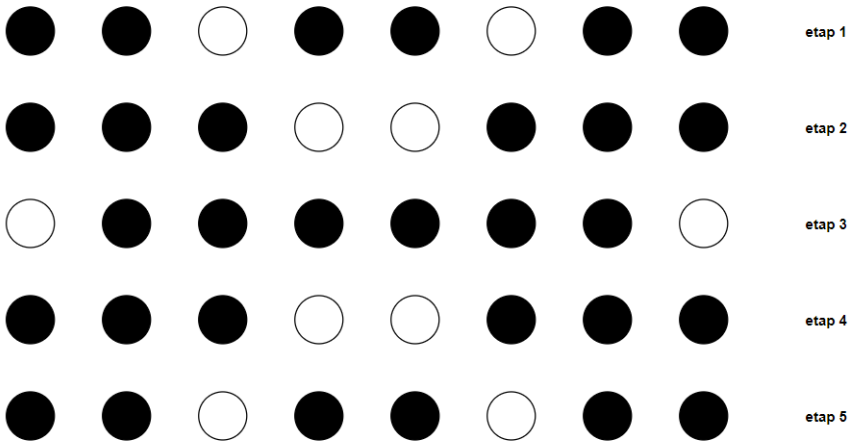
Pod schematami umieszczona jest natomiast implementacja wraz z komentarzami.



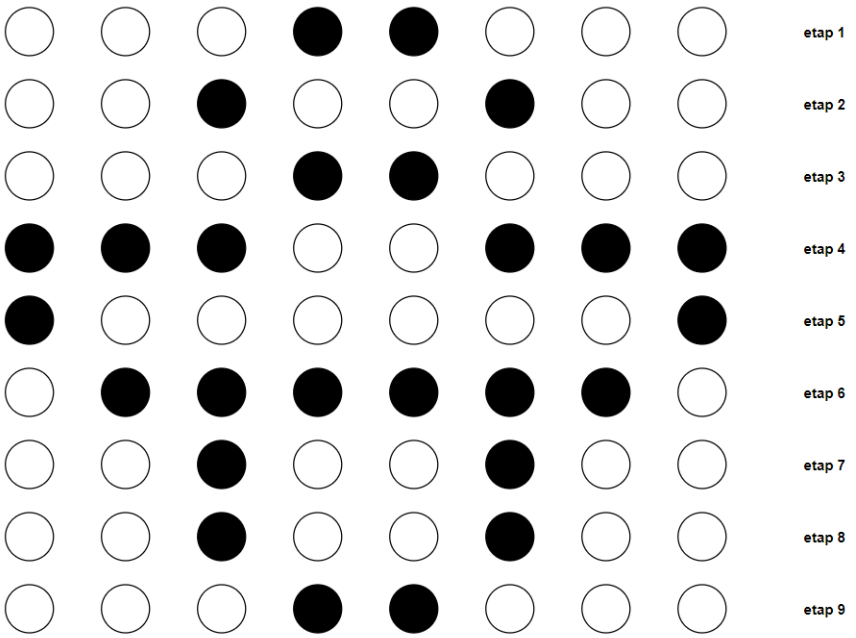
Rysunek 1: Koncept 1 - dziwne coś 1



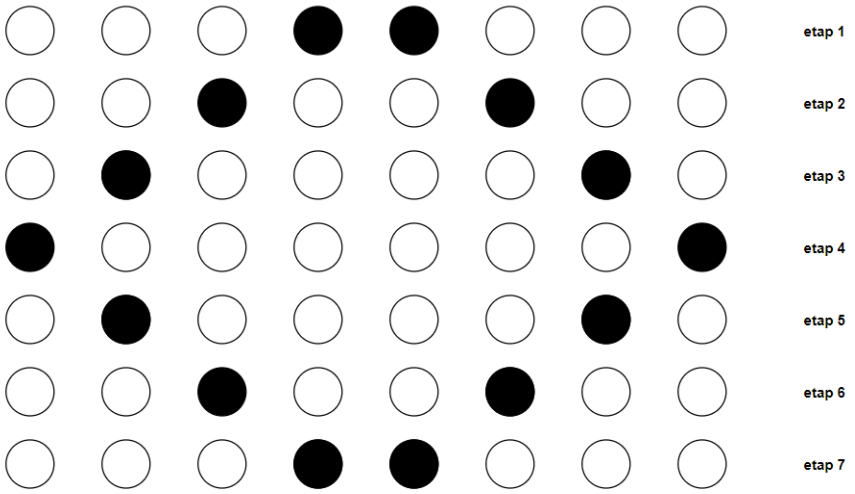
Rysunek 2: Koncept 2 - dziwne coś 2



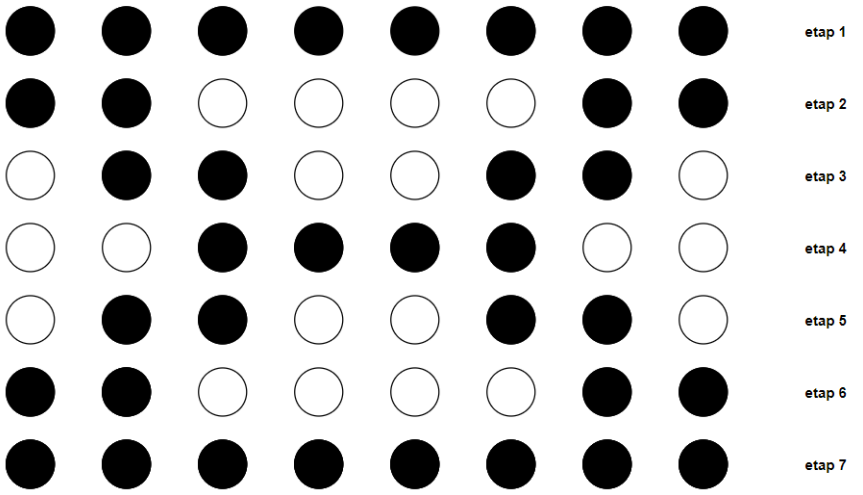
Rysunek 3: Koncept 3 - cukierek



Rysunek 4: Koncept 4 - anioł



Rysunek 5: Koncept 5 - dywan



Rysunek 6: Koncept 6 - klepsydra

Implementacja wygląda następująco:

```
#include <p24fj128ga010.h>

_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_OFF & COE_OFF & FWDTEN_OFF)
```

```
_CONFIG2(FCKSM_CSDCMD & OSCIOFNC_ON & POSCMOD_HS & FNOSC_PRI)

/* numer ktÃ†rym dostajemy siÄ? do potencjometru */
# define TVOLT 5# define AINPUTS 0xffcf

void ADCinit(int amask) {
  AD1PCFG = amask;
  AD1CON1 = 0x00e0;
  AD1CSSL = 0;
  AD1CON2 = 0;
  AD1CON3 = 0x1f02;
  AD1CON1bits.ADON = 1;
}

int readADC(int ch) {
  AD1CHS = ch;
  AD1CON1bits.SAMP = 1;
  while (!AD1CON1bits.DONE);
  return ADC1BUF0;
}

int main(void) {
  unsigned long SCALE = 308 L;
  unsigned long SCALEprogram = 308 L;
  unsigned long i;
  unsigned char display = 0;

  int numerProgramu = 0;
  int volt;
  int iterator = 0;

  //maksymalna wartoĹ?Ä? jakÄ? moĹźna odczytaÄ? to ok 1123. na jej
  //podstawie przeskalujemy wynik
  int maxOdczyt = 1123;

  //inicjalizacja
  PORTA = 0x0000;
  TRISA = 0xFF00;
  TRISD = 0xFFFF;
  ADCinit(AINPUTS); //inicjalizacja konwertera AD

  while (1) {
    Nop();
    PORTA = (unsigned int) display;

    for (i = SCALEprogram * SCALE; i > 0; i--) Nop();
```

```
volt = readADC(TVOLT); //Odczytanie potencjometru

// ****************************************** //
// obsluga zmiany programu
// jezeli wcisnieto RD6 i numerProgramu jest mniejszy od 6
if (PORTDbits.RD6 == 0 && numerProgramu < 6) {
  // zwieksz biezaca wartosc numerProgramu o 1
  numerProgramu = numerProgramu + 1;
// lub jesli wcisnieto RD13 i numerProgramu jest >= 0
} else if (PORTDbits.RD13 == 0 && numerProgramu >= 0) {
  // zmniejsz biezaca wartosc numerProgramu o 1
  numerProgramu = numerProgramu - 1;
}

// ****************************************** //
// obsluga zmiany predkosci

// jezeli wartosc pod zm. volt jest >= maxOdczyt
if (volt >= maxOdczyt) {
  // przypisz pod SCALEprogram wartosc 300 L
  SCALEprogram = 300 L;
// lub jezeli wartosc volt jest mniejsza od 200
} else if (volt < 200) {
  // przypisz pod SCALEprogram wartosc 200 L
  // ok. 0,2s
  SCALEprogram = 200 L;
// lub jezeli wartosc volt jest wieksza od 200
// i jednoczesnie mniejsza lub rowna 350
} else if (volt > 200 && volt <= 350) {
  // przypisz pod SCALEprogram wartosc 500 L
  // ok. 0,5 sekundy
  SCALEprogram = 500 L;
// lub jezeli wartosc volt jest wieksza od 350
// i jednoczesnie mniejsza lub rowna 450
} else if (volt > 350 && volt <= 450) {
  // przypisz pod SCALEprogram wartosc 1000 L
  // ok. 1 sekundy
  SCALEprogram = 1000 L;
// lub jezeli wartosc volt jest wieksza od 450
// i jednoczesnie mniejsza lub rowna 600
} else if (volt > 450 && volt <= 600) {
  // przypisz pod SCALEprogram wartosc 2000 L
  // ok. 2 sekundy
  SCALEprogram = 2000 L;
// lub jezeli wartosc volt jest wieksza od 600
```

```
// i jednoczesnie mniejsza lub rowna 800
} else if (volt > 600 && volt <= 800) {
  // przypisz pod SCALEprogram wartosc 3000 L
  // ok. 3 sekundy
  SCALEprogram = 3000 L;
// lub jezeli wartosc volt jest wieksza od 800
// i jednoczesnie mniejsza lub rowna 1122
} else if (volt > 800 && volt <= 1122) {
  // przypisz pod SCALEprogram wartosc 5000 L
  // ok. 5 sekund
  SCALEprogram = 5000 L;
}

// jezeli wykonane zostalo 50 iteracji zmien program
if (iterator == 50) {
  // przypisz do numerProgramu wynik sumy numerProgramu + 1
  numerProgramu = numerProgramu + 1;

  // ustaw zm. iterator na 0
  iterator = 0;
}

// jezeli numerProgramu osiagnal 6
if (numerProgramu == 6) {
  // przypisz do numerProgramu wartosc 0
  numerProgramu = 0;
// lub jesli numerProgramu osiagnal -1
} else if (numerProgramu == -1) {
  // przypisz do numerProgramu wartosc 5
  numerProgramu = 5;
}

// ****************************************** //

// jezeli wartosc zm. numerProgramu jest rowna 0
// tryb1
if (numerProgramu == 0) {
  // przypisz do zmiennej display 170
  display = 170;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);
```

```
  // przypisz do zmiennej display 85
  display = 85;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 255
  display = 255;
}

// jezeli wartosc zm. numerProgramu jest rowna 1
// tryb2
else if (numerProgramu == 1) {
  // przypisz do zmiennej display 255
  display = 255;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 85
  display = 85;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 170
  display = 170;
}

// jezeli wartosc zm. numerProgramu jest rowna 2
// tryb3
else if (numerProgramu == 2) {
  // przypisz do zmiennej display 219
```

```
  display = 219;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 231
  display = 231;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 126
  display = 126;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 231
  display = 231;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 219
  display = 219;
}

// jezeli wartosc zm. numerProgramu jest rowna 3
// tryb4
else if (numerProgramu == 3) {
```

```
// przypisz do zmiennej display 24
display = 24;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 36
display = 36;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 24
display = 24;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 231
display = 231;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 129
display = 129;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
```

```
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 126
  display = 126;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 36
  display = 36;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 36
  display = 36;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 24
  display = 24;
}

// jezeli wartosc zm. numerProgramu jest rowna 4
// tryb5
else if (numerProgramu == 4) {
  // przypisz do zmiennej display 24
  display = 24;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;
```

```
// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 36
display = 36;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 66
display = 66;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 129
display = 129;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 66
display = 66;
// przypisz wartosc zm. display do PORTA
PORTA = display;

// wykonaj petle for aby zrobic efekt opoznienia
// opoznienie w zaleznosci od przechowywanej
// wartosci przez SCALEprogram
for (i = SCALEprogram * SCALE; i > 0; i--);

// przypisz do zmiennej display 36
```

```
  display = 36;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 24
  display = 24;
}

// jezeli wartosc zm. numerProgramu jest rowna 5
// tryb6
else if (numerProgramu == 5) {
  // przypisz do zmiennej display 255
  display = 255;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 195
  display = 195;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);

  // przypisz do zmiennej display 102
  display = 102;
  // przypisz wartosc zm. display do PORTA
  PORTA = display;

  // wykonaj petle for aby zrobic efekt opoznienia
  // opoznienie w zaleznosci od przechowywanej
  // wartosci przez SCALEprogram
  for (i = SCALEprogram * SCALE; i > 0; i--);
```

```
      // przypisz do zmiennej display 60
      display = 60;
      // przypisz wartosc zm. display do PORTA
      PORTA = display;

      // wykonaj petle for aby zrobic efekt opoznienia
      // opoznienie w zaleznosci od przechowywanej
      // wartosci przez SCALEprogram
      for (i = SCALEprogram * SCALE; i > 0; i--);

      // przypisz do zmiennej display 102
      display = 102;
      // przypisz wartosc zm. display do PORTA
      PORTA = display;

      // wykonaj petle for aby zrobic efekt opoznienia
      // opoznienie w zaleznosci od przechowywanej
      // wartosci przez SCALEprogram
      for (i = SCALEprogram * SCALE; i > 0; i--);

      // przypisz do zmiennej display 195
      display = 195;
      // przypisz wartosc zm. display do PORTA
      PORTA = display;

      // wykonaj petle for aby zrobic efekt opoznienia
      // opoznienie w zaleznosci od przechowywanej
      // wartosci przez SCALEprogram
      for (i = SCALEprogram * SCALE; i > 0; i--);

      // przypisz do zmiennej display 255
      display = 255;
    }

    // przypisz pod zmienna iterator wynik sumy iterator + 1
    iterator = iterator + 1;
  }

  return 0;
}
```

5 Zadanie 5

Treść zadania:
Zegar szachowy, 2 graczy, 3 nastawy (1min, 3min, 5 min). RD6 - przełącznik

gracza 1, RD13 - przełącznik gracza 2.

Zadanie 5 wydawało się na pierwszy rzut oka poza naszym zasięgiem. Zabraliśmy się za nie po połowie zajęć 23 maja 2019 roku (czyli dość późno). Ku mojemu zaskoczeniu udało nam się je jednak wykonać do końca regulaminowego czasu zajęć. Nie jest to jednak pełny sukces ponieważ wyświetlamy liczbę sekund graczy i o ile dobrze to wygląda dla nastaw 1 minuta i 3 minuty to dla 5 minut niestety słabo.. Dopóki licznik danego gracza nie zejdzie poniżej 255 "sekund" to na diodach nie zobaczymy żadnych zmian. Gdybyśmy mieli do dyspozycji 9 bitów do rozdysponowania na diodach to byśmy użyli kodu Graya do przedstawienia 300 sekund ale nie mamy. Implementacja wraz z zakomentowanymi liniami kodu znajduje się poniżej.

```
#include <p24fj128ga010.h>

_CONFIG1(JTAGEN_OFF & GCP_OFF & GWRP_OFF & BKBUG_OFF & COE_OFF & FWDTEN_OFF)
_CONFIG2(FCKSM_CSDCMD & OSCIOFNC_ON & POSCMOD_HS & FNOSC_PRI)

/* numer ktÃ?rym dostajemy siÄ? do potencjometru */
#define TVOLT 5
#define AINPUTS 0xffcf

void ADCinit(int amask)
{
  AD1PCFG = amask;
  AD1CON1 = 0x00e0;
  AD1CSSL = 0;
  AD1CON2 = 0;
  AD1CON3 = 0x1f02;
  AD1CON1bits.ADON = 1;
}

int readADC(int ch)
{
  AD1CHS = ch;
  AD1CON1bits.SAMP = 1;
  while (!AD1CON1bits.DONE);
  return ADC1BUF0;
}


int main(void)
{
  unsigned long SCALE = 308L;
  unsigned long i;
```

```
unsigned char display=0;

// przechowuje wartosc 1, 2 lub 3 w zaleznosci od nastawy
// jaka aktualnie jest wskazywana
unsigned int wybrana = 0;

// 'kontrolka' przechowujaca informacje czy wybrana zostala
// nastawa 0 -> NIE, 1 -> TAK
unsigned int wybranoNastawe = 0;

// 'kontrolka' przechowujaca informacje czy czas graczy
// zostal zainicjalizowany 0 - NIE, 1 - TAK
unsigned int zainicjalizowanoCzas = 0;

unsigned int czasGracza1 = 0; // przechowuje czas gracza 1
unsigned int czasGracza2 = 0; // przechowuje czas gracza 2

// pilnuje kogo obecnie jest tura
// ustalmy, ze 0 => Gracz 1, 1 => Gracz 2
unsigned int ktoTeraz = 0;

// nastawa przechowuje zatwierdzona nastawe
unsigned int nastawa = 0;

int volt;

//maksymalna wartoĹ?Ä? jakÄ? moĹźna odczytaÄ? to ok 1123. na jej
//podstawie przeskalujemy wynik
int maxOdczyt = 1123;

//inicjalizacja
PORTA = 0x0000;
TRISA=0xFF00;
TRISD=0xFFFF;
ADCinit(AINPUTS); //inicjalizacja konwertera AD


while (1) {
  Nop();
  PORTA=(unsigned int) display;

  for (i = SCALE * SCALE; i > 0; i--) Nop();

  volt = readADC(TVOLT); //Odczytanie potencjometru

  // ****************************************** //
```

```
// sprawdzenie jak ustawiony jest potencjometr

// jezeli zmienna volt ma wartosc mniejsza/rowna 374
if (volt <= 374)
{
  // przypisz do zm. wybrana wartość 1
  wybrana = 1;
}
// lub jesli zmienna volt ma wartosc wieksza od 374
// i jednoczesnie mniejsza/rowna 748
else if (volt > 374 && volt <= 748)
{
  // przypisz do zm. wybrana wartosc 3
  wybrana = 3;
}
// lub jesli zmienna volt ma wartosc wieksza od 748
else if (volt > 748)
{
  // przypisz do zm. wybrana wartosc 5
  wybrana = 5;
}

// jezeli wcisniesz RD7 to zatwierdzasz aktualna nastawe
if (PORTDbits.RD7 == 0)
{
  // przypisz do nastawa biezaca wartosc spod zm. wybrana
  nastawa = wybrana;

  // przypisz do zm. wybranoNastawe wartosc 1
  wybranoNastawe = 1;
}

// jezeli nie zatwierdzono nastawy pokaz aktualnie wskazywana
// na diodach
if(wybranoNastawe == 0)
{
  // przypisz do zm. display biezaca wartosc zm. wybrana
  display = wybrana;
}

// ****************************************** //

// jezeli wybranoNastawe
// (czyli jezeli wartosc zm. wybranoNastawe jest rozna od 0)
if(wybranoNastawe)
```

```
{
  // jezeli nastawa wynosi 1
  if (nastawa == 1)
  {
    // jezeli zainicjalizowanoCzas wynosi 0
    // proces inicjalizacji czasu
    if (zainicjalizowanoCzas == 0)
    {
      // przypisz do czasGracza1 60 (co odpowiada 1min)
      czasGracza1 = 60;
      // przypisz do czasGracza2 60 (co odpowiada 1min)
      czasGracza2 = 60;
      // przypisz 1 do zm. zainicjalizowanoCzas
      zainicjalizowanoCzas = 1;
    }
  }
  // lub jezeli nastawa wynosi 3
  else if (nastawa == 3)
  {
    // jezeli zainicjalizowanoCzas wynosi 0
    // proces inicjalizacji czasu
    if (zainicjalizowanoCzas == 0)
    {
      // przypisz do czasGracza1 180 (co odpowiada 3min)
      czasGracza1 = 180;
      // przypisz do czasGracza2 180 (co odpowiada 3min)
      czasGracza2 = 180;
      // przypisz 1 do zm. zainicjalizowanoCzas
      zainicjalizowanoCzas = 1;
    }
  }
  // lub jezeli nastawa wynosi 5
  else if (nastawa == 5)
  {
    // jezeli zainicjalizowanoCzas wynosi 0
    // proces inicjalizacji czasu
    if (zainicjalizowanoCzas == 0)
    {
      // przypisz do czasGracza1 300 (co odpowiada 5min)
      czasGracza1 = 300;
      // przypisz do czasGracza2 300 (co odpowiada 5min)
      czasGracza2 = 300;
      // przypisz 1 do zm. zainicjalizowanoCzas
      zainicjalizowanoCzas = 1;
    }
  }
```

```
      // opoznienie
      for (i = 1000L * SCALE; i > 0; i--);

      // proces odliczania (zaczyna gracz 1)
      // gdy ktoTeraz jest rowne zeru to odliczany jest
      // czasGracza1, gdy ktoTeraz jest rowne 1 - odliczamy
      // czasGracza2

      // jezeli ktoKteraz jest rowne 0
      if(ktoTeraz == 0)
      {
        // przypisz do zm. display biezaca wartosc zm. czasGracza1
        display = czasGracza1;
        // zmniejsz czasGracza1 o 1
        czasGracza1 = czasGracza1 - 1;
      }
      // lub jesli ktoTeraz jest rowne 1
      else if(ktoTeraz == 1)
      {
        // przypisz do zm. display biezaca wartosc zm. czasGracza2
        display = czasGracza2;
        // zmniejsz czasGracza2 o 1
        czasGracza2 = czasGracza2 - 1;
      }

      // sprawdzenie czy gracz nacisnal swoj przycisk

      // jezeli wcisnieto RD6 i ktoTeraz wynosi 0
      // (czyli gdy ture ma teraz gracz 1 i nacisnal przycisk)
      if (PORTDbits.RD6 == 0 && ktoTeraz == 0)
      {
        // ustaw zmienna ktoTeraz na 1
        ktoTeraz = 1;
      }
      // lub jezeli wcisnieto RD13 i ktoTeraz wynosi 1
      // (czyli gdy tura gracza 2 i nacisniety zostal przycisk RD13)
      else if (PORTDbits.RD13 == 0 && ktoTeraz == 1)
      {
        // ustaw zmienna ktoTeraz na 0
        ktoTeraz = 0;
      }
    }
  }
  return 0;
}
```