**Raport do zadania 5**

Imię i nazwisko: Jakub Dmochowski

Numer Albumu: 169236

**Konfiguracja sprzętowa**

**A table with black text

AI-generated content may be incorrect.**

POSCMOD = XT - wybór zewnętrznego kwarcu jako źródła zegara

OSCIOFNC = ON - funkcja wyjścia oscylatora włączona

FCKSM = CSDCMD - wyłączenie monitorowania zegara i przełączania

FNOSC = PRI - Podstawowe źródło zegara jako domyślne

IESO = ON - włączenie wewnętrznego/zewnętrznego przełączania oscylatora

**Watchdog timer, debugowanie**

**A table of text with different colored letters

AI-generated content may be incorrect.**

WDTPS = PS32768 - Prescaler watchdog timera ustawiony na 1:32768

FWPSA = PR128 - Prescaler A watchdog timera ustawiony na 1:128

WINDIS = ON - Windowed watchdog timer wyłączony

FWDTEN = OFF - Watchdog timer całkowicie wyłączony

ICS = PGx2 - Komunikacja z debuggerem przez piny PGC2/PGD2

GWRP = OFF - Ochrona zapisu do pamięci programu wyłączona

GCP = OFF - Ochrona odczytu kodu wyłączona

JTAGEN = OFF - Interfejs JTAG wyłączony

**Biblioteki**

A computer code with text

AI-generated content may be incorrect.

#include <xc.h> - główna biblioteka dla kompilatorów XC

#include <libpic30.h> - biblioteka dla PIC30, zawiera funkcje opóźnień

#include <stdbool.h> - biblioteka standardowa C dla typu bool

#include <stdio.h> - biblioteka standardowa C dla funkcji I/O

#include <string.h> - biblioteka standardowa C dla operacji na stringach

#include "lcd.h" - biblioteka do obsługi LCD

#include "adc.h" - biblioteka do obsługi ADC

**Definicje i zmienne globalne**

**A screenshot of a computer program

AI-generated content may be incorrect.**

#define FCY 4000000UL - częstotliwość procesora 4 MHz

#define DEBOUNCE\_DELAY\_MS 2000 - opóźnienie dla eliminacji drgań styków

#define PRZYCISK\_GRACZ1 PORTDbits.RD6 - przycisk gracza 1

#define PRZYCISK\_GRACZ2 PORTDbits.RD13 - przycisk gracza 2

unsigned int czas\_g1, czas\_g2 - czas pozostały dla każdego gracza

unsigned char aktywny\_gracz = 0; - Numer gracza, którego czas aktualnie jest odliczany

unsigned int wybrany\_czas - czas wybrany przez potencjometr

volatile bool przycisk\_g1\_flag = false; - Flaga ktora sygnalizuje wcisniecie przez gracza 1

volatile bool przycisk\_g2\_flag = false; - Flaga ktora sygnalizuje wcisniecie przez gracza 2

volatile bool przycisk\_flag = false; - Flaga która sygnalizuje wciśnięcie jakiegokolwiek przycisku

**Przerwania**

**A computer code with black text

AI-generated content may be incorrect.**

void attribute((interrupt, auto\_psv)) \_CNInterrupt(void) - Funkcja obsługi przerwania od zmiany stanu pinów

if (!PRZYCISK\_GRACZ1) - Sprawdzenie czy przycisk gracza 1 został naciśnięty

przycisk\_g1\_flag = true; - Ustawienie flagi sygnalizującej naciśnięcie przycisku gracza 1

if (!PRZYCISK\_GRACZ2) - Sprawdzenie czy przycisk gracza 2 został naciśnięty

przycisk\_g2\_flag = true; - Ustawienie flagi sygnalizującej naciśnięcie przycisku gracza 2

przycisk\_flag = true; - Ustawienie ogólnej flagi informującej o naciśnięciu przycisku

IFS1bits.CNIF = 0; - Wyzerowanie flagi przerwania Change Notification

**Opóźnienie**

A white background with black text

AI-generated content may be incorrect.

while (ms--) - Pętla wykonywana określoną liczbę razy

\_\_delay32(FCY / 1000); - Opóźnienie na 1 milisekundę bazujące na częstotliwości procesora

**Wyświetlanie czasu gry**

**A white background with black and orange text

AI-generated content may be incorrect.**

char bufor[33] - bufor dla sformatowanego tekstu

char wskaznik1 = (aktywny\_gracz == 1) ? '>' : ' ' - wskaźnik dla gracza 1

char wskaznik2 = (aktywny\_gracz == 2) ? '>' : ' ' - wskaźnik dla gracza 2

sprintf(bufor, "%cG1: %02u:%02u\n%cG2: %02u:%02u", ...) - formatowanie tekstu:

* %c - znak wskaźnika
* %02u - liczba bez znaku, minimum 2 cyfry z zerami
* czas\_g1 / 60 - minuty
* czas\_g1 % 60 - sekundy
* \n - znak nowej linii między graczami

LCD\_ClearScreen() - wyczyszczenie wyświetlacza LCD

LCD\_PutString(bufor, strlen(bufor)) - wyświetlenie tekstu z obliczeniem długości

**Wyświetlanie komunikatu**

**A close-up of a computer code

AI-generated content may be incorrect.**

LCD\_ClearScreen() - wyczyszczenie wyświetlacza

LCD\_PutString(msg, strlen(msg)) - wyświetlenie komunikatu z automatycznym obliczeniem długości

**Wyświetlanie czasu**

**A close up of a code

AI-generated content may be incorrect.**

bufor[33] - bufor dla sformatowanego tekstu

sprintf(bufor, "Nastaw czas:\n%u:%02u min", wybrany\_czas / 60, wybrany\_czas % 60)

* "Nastaw czas:" - tekst instrukcji w pierwszej linii
* %u - liczba bez znaku bez wypełnienia zerami (minuty)
* %02u - sekundy z wypełnieniem zerami do 2 cyfr
* "min" - jednostka czasu

Wyświetla aktualnie wybrany czas z potencjometru

**Odczyt z potencjometru**

**A screenshot of a computer program

AI-generated content may be incorrect.**

uint16\_t pot = ADC\_Read10bit(ADC\_CHANNEL\_5) - odczyt 10-bitowej wartości ADC

Mapowanie zakresu na 3 opcje czasowe:

* if (pot < 341) return 60 - 1 minuta
* else if (pot < 682) return 180 - 3 minuty
* else return 300 - 5 minut

Zwraca czas w sekundach

**Konfiguracja przerwań**

**A screenshot of a computer code

AI-generated content may be incorrect.**

CNEN1bits.CN15IE = 1; - Włączenie przerwania dla pinu CN15 (RD6)

CNEN2bits.CN19IE = 1; - Włączenie przerwania dla pinu CN19 (RD13)

IPC4bits.CNIP = 2; - Ustawienie priorytetu przerwania na poziom 2

IFS1bits.CNIF = 0; - Wyczyszczenie flagi przerwania przed włączeniem

IEC1bits.CNIE = 1; - Włączenie obsługi przerwań

**Pętla ustawienia czasu**

A screenshot of a computer program

AI-generated content may be incorrect.

**void petla\_ustawiania\_czasu() - Główna funkcja**

unsigned int poprzedni\_czas = 0; - Zmienna do śledzenia zmian potencjometru

komunikat("Ustaw czas"); - Wyświetlenie komunikatu

opoznienie\_ms(1500); - Pauza 1.5 sekundy na przeczytanie

**while(1)**

wybrany\_czas = odczytaj\_czas\_z\_potencjometru(); - Ciągły odczyt pozycji potencjometru

**if (wybrany\_czas != poprzedni\_czas) - Sprawdzenie czy pozycja potencjometru się zmieniła**

* pokaz\_ustawienia\_czasu(); - Aktualizacja wyświetlacza z nową wartością czasu
* poprzedni\_czas = wybrany\_czas; - Zapamiętanie nowej wartości

**if (przycisk\_flag) - Sprawdzenie czy któryś przycisk został naciśnięty**

* \_\_delay32(DEBOUNCE\_DELAY); - Opóźnienie eliminujące drgania styków

**if (przycisk\_g1\_flag && !PRZYCISK\_GRACZ1) - Potwierdzenie naciśnięcia przycisku gracza 1**

* czas\_g1 = czas\_g2 = wybrany\_czas; - Ustawienie czasu dla obu graczy na wybraną wartość
* while (!PRZYCISK\_GRACZ1) - Oczekiwanie na zwolnienie przycisku
* return; - Wyjście z funkcji ustawiania czasu

**Dla gracza 2 następuję identyczne działanie ustawiania gry**

**Inicjalizacja**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**void inicjalizacja() - Funkcja konfigurująca wszystkie peryferia mikrokontrolera**

TRISDbits.TRISD6 = 1; - Konfiguracja pinu RD6 jako wejście dla przycisku gracza 1

TRISDbits.TRISD13 = 1; - Konfiguracja pinu RD13 jako wejście dla przycisku gracza 2

TRISA = 0xFFFF; - Cały port A skonfigurowany jako wejścia analogowe dla ADC

ADC\_SetConfiguration(ADC\_CONFIGURATION\_DEFAULT); - Ustawienie domyślnej konfiguracji ADC

ADC\_ChannelEnable(ADC\_CHANNEL\_5); - Włączenie kanału 5 ADC dla potencjometru

LCD\_Initialize(); - Inicjalizacja wyświetlacza LCD

**Główna pętla gry**

**A screenshot of a computer program

AI-generated content may be incorrect.**

A screenshot of a computer code

AI-generated content may be incorrect.

**void petla\_zegaru() – Główna funkcja zegaru szachowego**

unsigned int licznik\_sekund = 0; - Licznik do odmierzania sekund opiera sie na iteracji

aktywny\_gracz = 0; - Na początku żaden gracz nie ma aktywnego zegara

komunikat("Nacisnij aby\nzaczac"); - Kominukat aby zacząć

opoznienie\_ms(2000); - Pauza na przeczytanie instrukcji

**while (1)**

if (przycisk\_flag) - Sprawdzenie czy został naciśnięty któryś przycisk

if (przycisk\_g1\_flag && !PRZYCISK\_GRACZ1) - Naciśnięcie przycisku gracza 1

aktywny\_gracz = 2; - Przełączenie zegara na gracza 2 kiedy gracz 1 skończył ruch

pokaz\_czas(); - Aktualizacja wyświetlacza

licznik\_sekund++; - Inkrementacja licznika dla odmierzania czasu

if (licznik\_sekund >= 100) - Po 100 iteracjach pętli

licznik\_sekund = 0; - Zerowanie licznika

if (aktywny\_gracz == 1 && czas\_g1 > 0) - Jeśli aktywny jest gracz 1 i ma jeszcze czas

czas\_g1--; - Odejmowanie jednej sekundy z czasu gracza 1

pokaz\_czas(); - Aktualizacja wyświetlacza

if (czas\_g1 == 0 && aktywny\_gracz == 1) - Sprawdzenie czy graczowi 1 skończył się czas

komunikat("Gracz 1 \nPRZEGRAL"); - Wyświetlenie komunikatu o przegranej

break; - Wyjście z pętli gry

opoznienie\_ms(10); - Krótkie opóźnienie między iteracjami pętli

**Dla gracza 2 działanie jest identyczne, lecz potem z gracza 2 przełącza na gracza 1**

**Petla main**

**A screenshot of a computer code

AI-generated content may be incorrect.**

**int main(void) - Punkt startowy programu**

inicjalizacja(); - Skonfigurowanie wszystkich peryferiów mikrokontrolera

konfiguruj\_przerwania(); - Włączenie obsługi przycisków przez przerwania

petla\_ustawiania\_czasu(); - Przejście do fazy ustawiania początkowego czasu gry

komunikat("Czas ustawiony"); - Potwierdzenie zakończenia ustawiania

opoznienie\_ms(2000); - Pauza przed rozpoczęciem gry

petla\_zegaru(); - Rozpoczęcie właściwej gry z zegarem szachowym

**while (1) - Nieskończona pętla po zakończeniu gry**

* opoznienie\_ms(1000); - Program pozostaje aktywny, ale nic nie robi

return 0; - Koniec programu