PTM - Końcowy program

Kierunek *Automatyka i Robotyka*	**Termin** *Wtorek 13:15 TP*
Temat *Końcowy program*	**Problem**
Skład grupy *Adam Jankowiak 252919 Kacper Klupś 252870*	**Nr grupy**
Prowadzący *Mgr inż. Wojciech Tarnawski*	**data** *8 czerwca 2021*

1 Wstęp

1.1 Czego się nauczyliśmy?

Podczas kursu nauczyliśmy się obsługi wyświetlacza LED, wyświetlacza 7 segmentowego, klawiatury, przerwań oraz odpowiedniej inicjalizacji danych funkcji.

1.2 Oczekiwania

Zaczynając ten kurs nie mieliśmy jakiegokolwiek pojęcia o programowaniu mikrokontrolerów, dlatego nasze oczekiwania nie były zbyt wysokie.

1.3 Podsumowanie

Uważam, że podczas kursu nauczyłem się wielu przydatnych rzeczy, które można wykorzystać w codziennym życiu. Materiał podczas zajęć był bardzo dobrze tłumaczony. Po każdym spotkaniu wiedziałem dokładnie o czym była mowa i byłem w stanie samodzielnie napisać program bez większych problemów.

2 Menu

Funkcja Menu opiera się na czterech przyciskach. Przyciski Up / Down odpowiadają za poruszanie się po menu w górę / dół. Przycisk X służy do powrotu z wybranej przez nas funkcji do Menu. Natomiast przycisk Ok (W naszym przypadku to znak pierwiastka) odpowiada za uruchomianie wybranej funkcji. Funkcje można rozpoznać poprzez 5 tytułów:

Info - wyświetla nam ponownie ekran powitalny na 4 sekundy.

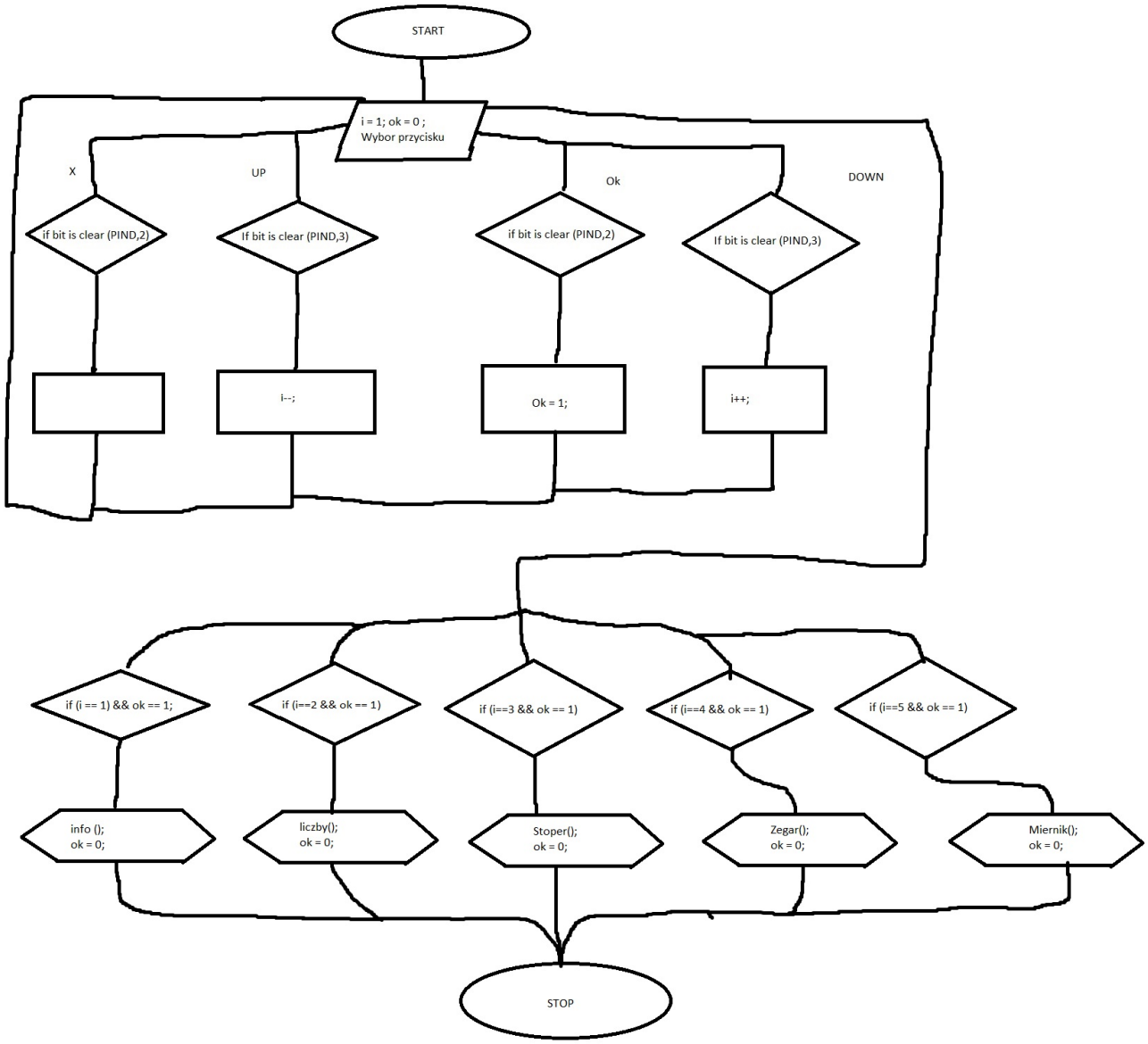
Liczby - Z zakresu 1-50 zostaję wybierane liczby przy pomocy UP/DOWN. Za pomocą pierwszej diody widać czy liczba jest parzysta , za pomocą trzeciej czy jest pierwsza , dioda druga świeci sie gdy liczba nie jest parzysta.

Stoper - Funkcja uruchamia stoper. Na wyświetlaczu pokazane są milisekundy oraz sekundy. Dokładność 0,1s.

Zegar - Funkcja uruchamia zegar. Dokładność 1s

Miernik - Wykorzystuje przetwornik ADC do do pomiaru napięcia z potencjometru.

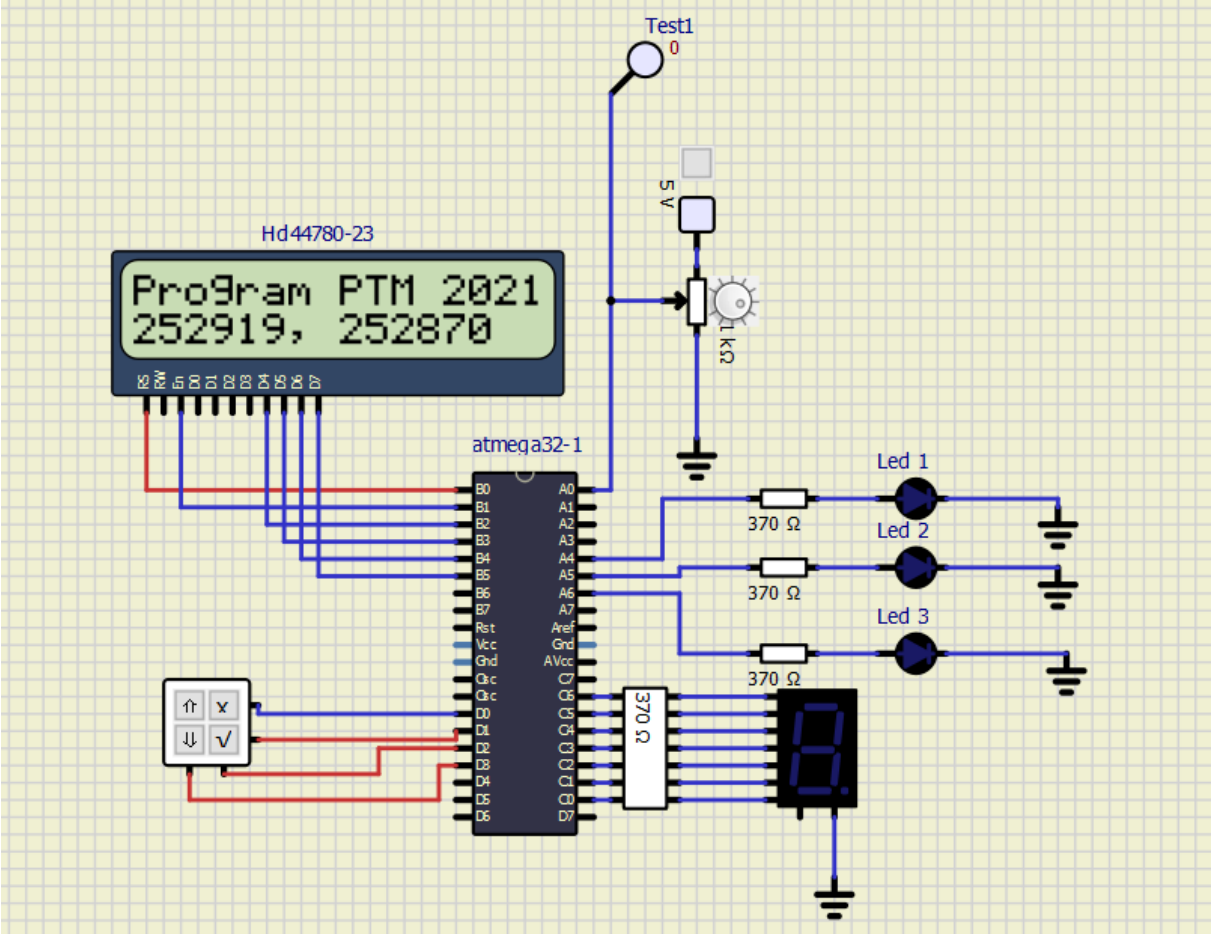
2.1 Schemat blokowy



Rysunek 1: Schemat blokowy Menu

3 Info

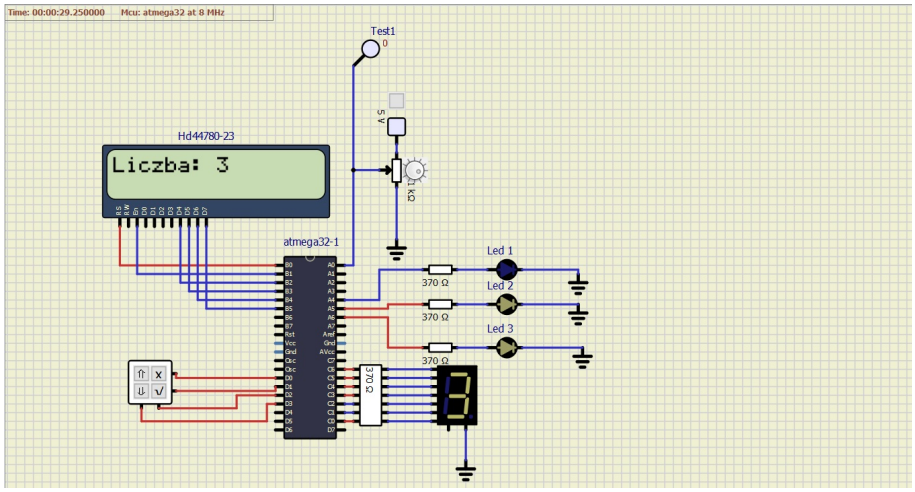
Funkcja Info opiera się na obsłudze wyświetlacza. Za pomocą LCD GoTo zostaje wybrana kolumna oraz wiersz w którym ma zostać zapisany nasz tekst. Sprintf zapisuje nasz tekst do tablicy , która za pomocą komendy LCD Write-Text zostaje wypisana na wyświetlaczu. Zostaje ona wyświetlona przy odpaleniu symulacji na czas 4 sekund. Oraz potem bo wybraniu Opcji Info z menu.



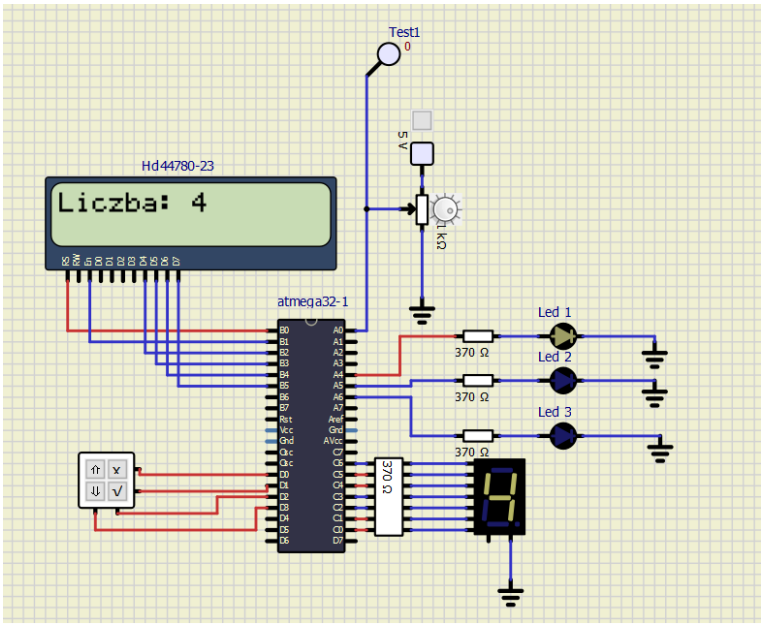
Rysunek 2: Przestawia wyświetlane informacje za pomocą funkcji info.

4 Liczby

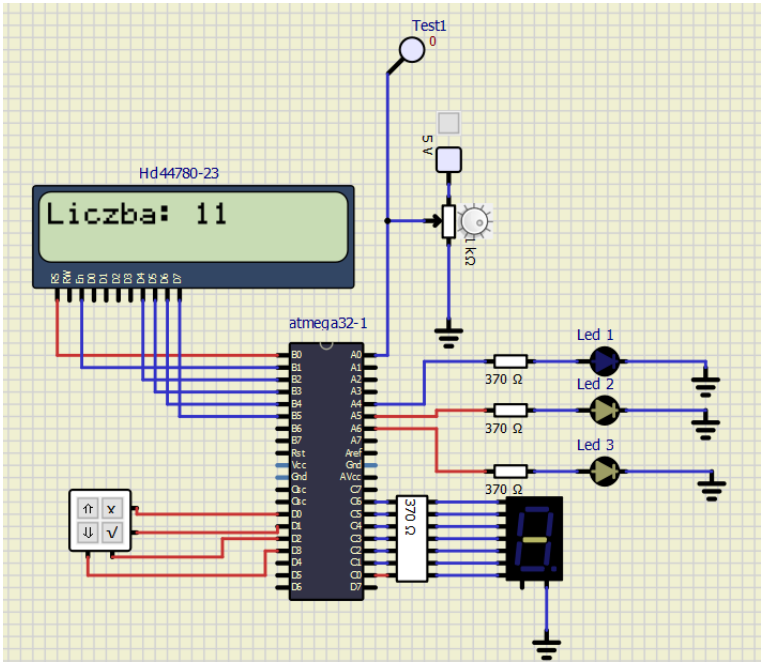
Funkcja liczby - po wciśmięciu przycisku UP zwiększa się wartość liczby o 1, natomiast po wciśnięciu DOWN zmniejszany jest o 1. Na wyświetlaczu LCD wyświetlane są liczby z zakresu od 0 do 50. Wyświetlacz 7 segmentowy wyświetla liczby z zakresu 0d 0 do 9 natomiast dla większych wartości wyświetla -". Po wciśnięciu przycisku X zmieniana jest flaga "wyjdźńa 0 co powoduje zakończenie dziabania funkcji. Dodatkowo po wybraniu liczby parzystaje pali się LED 1, dla pozpstałych LED 2 i dla liczb pierwszych zapala się LED 3.



Rysunek 3: Zdjęcie z symulacji - Zapalenie się LED 2 liczba nieparzysta, zapalenie się LED 3 liczba pierwsza.



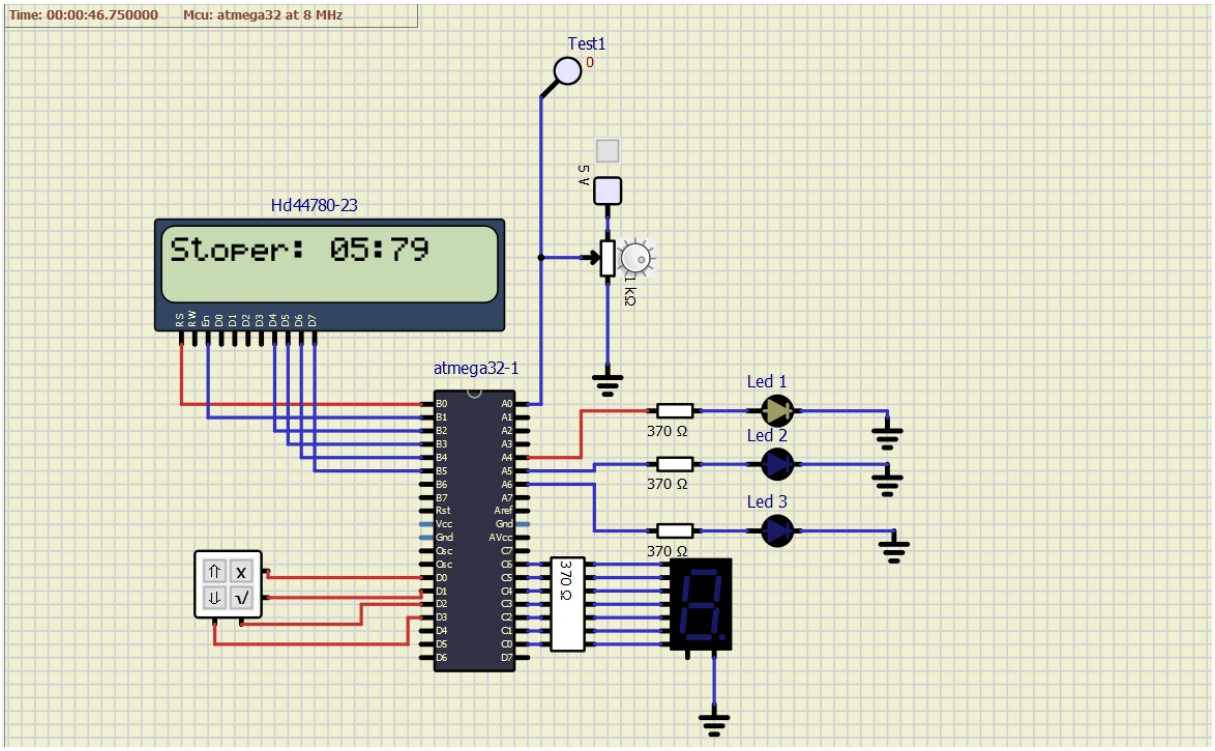
Rysunek 4: Przedstawia zapalenie się diody LED 1 dla liczby parzystej.



Rysunek 5: Przedstawia wyświetlenie - na wyświetlaczu 7 segmentowym dla liczby większej od 10.

5 Stoper

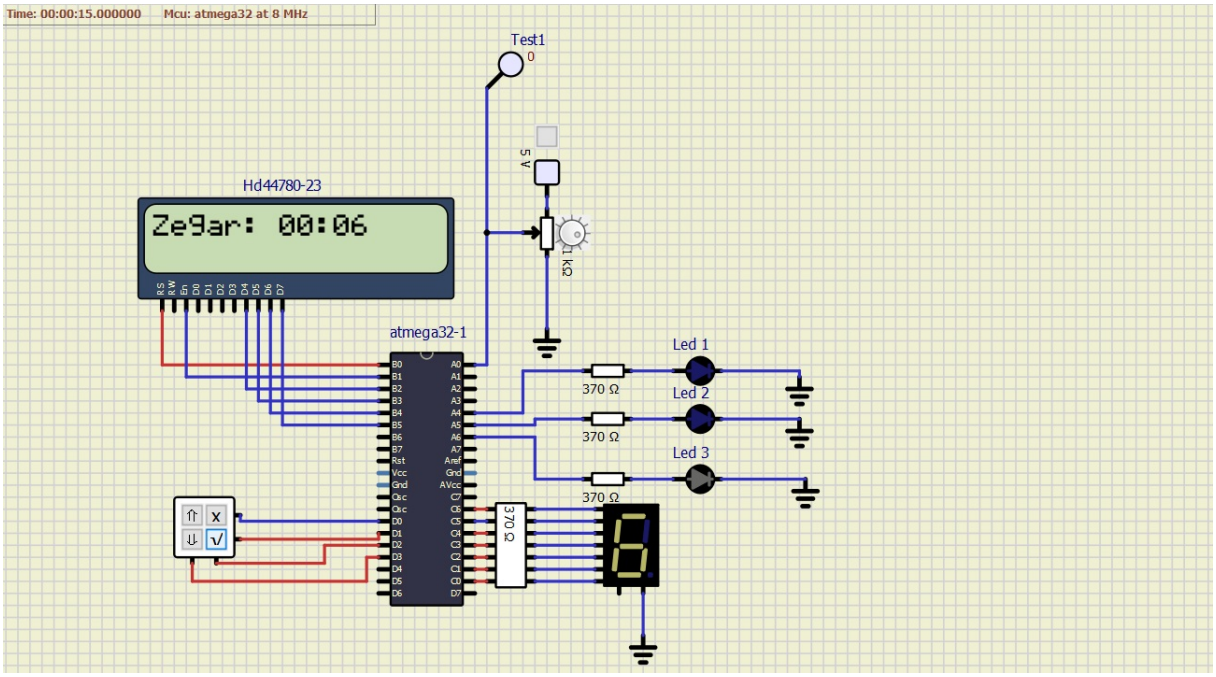
Funkcję rozpoczęto poprzez zainicjowanie Timera. Wybrano tryb pracy CTC z TOP OCR1A , dzielnik częstotliwości , przypisano odpowiedni parametr OCR1A w tym przypadku to 31.25 , oraz uruchomiono przerwania OCIE1A.
Główną jednostką w programie są milisekundy. W programie znajdują się flagi : kasuj , zatrzymaj wyjdź dzięki którym funkcja rozpoznaje kiedy zatrzymać stoper , zresetować go oraz wyjść do menu. Przycisk OK (W naszym programie jest to znak pierwiastka) odpowiada za start oraz reset stopera. Jego podwójne naciśnięcie powoduje wyzerowanie. Za pomocą przycisku X wychodzimy do menu programu.



Rysunek 6: Przedstawia poprawne wyświetlanie sekund i setnych części na wyświetlaczu LCD oraz zapalenie się diody LED 1 co 1 sek.

6 Zegar

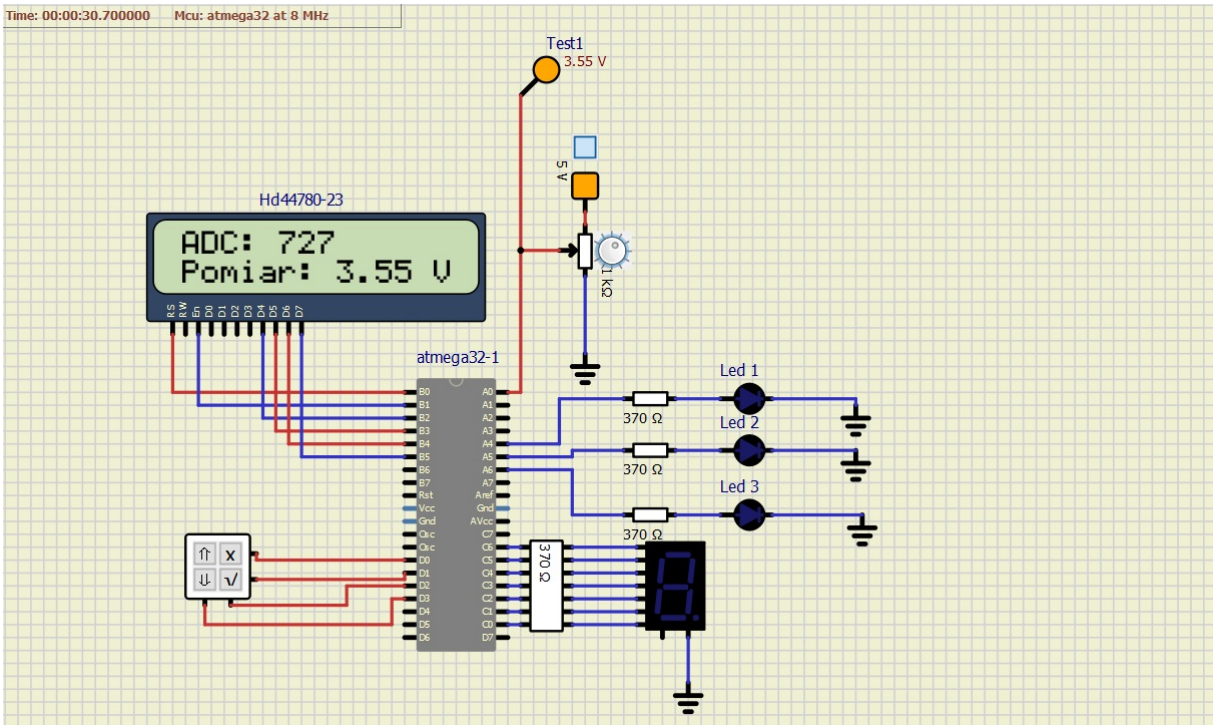
Zegar opera się na przerwaniach wewnętrznych , w których są zliczane kolejne sekundy. Funkcja inicjująca Timer różni się od stopera wartością OCR1A. Dodatkowo został także zainicjowany wyświetlacz 7-segmentowy ma którym wyświetlane są wartości od 0-9 w zależności którą mamy sekundę. Główną jednostką są sekundy. Na ich podstawie zliczane są odbywa się świecenie wyświetlacza 7-segmentowego. W momencie gdy nalicza się jedna sekunda wzrasta także zmienna pomocnicza dzięki której zostają wyświetlone kolejno wartości na wyświetlaczu. Liczby które mają zostać wyświetlane znajdują się w tablicy cyfry.
W programie znajdują się funkcję if które odpowiadają za wyświetlanie zegara. Gdy liczba sekund/minut jest mniejsza niż 10 wtedy na wyświetlaczu widzimy np.: 05:06. Liczby zostają zapisane w tablicach z następnie wyświetlane na wyświetlaczu. Na końcu można zauważyć obsługę diody która zapala się na masz 200ms co każdą rozpoczętą sekundę.



Rysunek 7: Przedstawia poprawne wyświetlanie zegara na LCD oraz liczb od 0-9 na 7-seg. Zapalana jest także dioda LED 3 na 0,2 sek.

7 Miernik

Na początek skonfigurowano przetwornik A/D. Opis Rejestrów znajduję się w kodzie programu w komentarzach. Następnie napisano funkcję ADC 10bit która odczytuje wartość z rejestru pomiarowego A/D. Ustawiono na stan wysoki port ADSC rejestr ADSCRA. Następnie mamy pętle while która czeka na wykonanie pomiaru i zostaje nam zwrócona wartość. Funkcja ADC measure zawiera w sobie wzór dzięki któremu zostaje wyliczona wartość napięcia. Przy użyciu komend wyświetlacza obie wartości zostają wyświetlone na wyświetlaczu.



Rysunek 8: W pierwszej lini LCD wyświetla dokładny pomiar z rejestru ADC, w dugiej lini jest wyświetlany pomiar w V.

8 Kod programu

```
1  #include <avr/io.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <stdio.h>
5  #include <util/delay.h>
6  #include <avr/sfr_defs.h>
7  #include <math.h>
8  #include <avr/interrupt.h>
9  #include <avr/eeprom.h>
10 #include <inttypes.h>
11 #include "HD44780.h"
12 #include <stdbool.h>
13
14
15
16 #ifndef _BV
17 #define _BV(bit)                (1<<(bit))
18 #endif
19
20 #ifndef inb
21 #define inb(addr)               (addr)
22 #endif
23
24 #ifndef outb
25 #define outb(addr, data)        addr = (data)
26 #endif
27
28 #ifndef sbi
29 #define sbi(reg,bit)            reg |= (_BV(bit))
30 #endif
31
32 #ifndef cbi
33 #define cbi(reg,bit)            reg &= ~(_BV(bit))
34 #endif
35
36 #ifndef tbi
37 #define tbi(reg,bit)            reg ^= (_BV(bit))
38 #endif
39
40 //  Gotowe zaimplementowane:
41 #define     bit_is_set(sfr, bit)    (_SFR_BYTE(sfr) & _BV(bit))
42 #define     bit_is_clear(sfr, bit)    (!(_SFR_BYTE(sfr) & _BV(bit)))
43 #define     loop_until_bit_is_set(sfr, bit)    do { } while (bit_is_clear(
       sfr, bit))
44 #define     loop_until_bit_is_clear(sfr, bit)    do { } while (bit_is_set(
       sfr, bit))
45
46
```

```
47
48  // MIN/MAX/ABS macros
49  #define MIN(a,b)            ((a<b)?(a):(b))
50  #define MAX(a,b)            ((a>b)?(a):(b))
51  #define ABS(x)              ((x>0)?(x):(-x))
52
53  //Napi cie referencyjne 5V
54
55  volatile uint32_t i;
56
57  //Funkcja inicjuje miernik sygna u analogowego
58  void ADC_init()
59  {
60      //Rejestr Admux
61      sbi(ADMUX,REFS0);
62      cbi(ADMUX,REFS1);
63      //Rejestr ADCSRA Mniejsze od 100kHz
64      sbi(ADCSRA,ADPS0);
65      sbi(ADCSRA,ADPS1);
66      sbi(ADCSRA,ADPS2);
67      //Rejestr ADCSRA Aden
68      sbi(ADCSRA,ADEN);
69      //ADMUX bity Mux - ustawienie kana u
70      cbi(ADMUX,MUX0);
71      cbi(ADMUX,MUX1);
72      cbi(ADMUX,MUX2);
73      cbi(ADMUX,MUX3);
74      cbi(ADMUX,MUX4);
75  }
76
77  //Funkcja zwraca wartosc odczytanego sygnalu analogowego
78  uint16_t ADC_10bit()
79  {
80
81      sbi(ADCSRA,ADSC);
82
83      while(bit_is_clear(ADCSRA,ADSC));
84
85      return(ADC);
86  }
87
88  //Funkcja przelicza wartosc odczytana na volty
89  uint32_t ADC_measure()
90  {
91
92      if(ADC_10bit() != 0){
93
94      return 1 + 500*((uint32_t)ADC_10bit())/1024;
95
96      }
```

```
97      return  500*((uint32_t)ADC_10bit())/1024;
98
99  }
100
101 //Funcja wyswietla poprawna wartosc napiecia w V
102 void wyswietl(){
103
104     char text[20];          //Tablica znakow
105     uint32_t a = 0;         //Zmienna pomocnicza
106     uint32_t b = 0;         //Zmienna pomocnicza
107     uint32_t x = 0;         //Zmienna pomocnicza
108
109     x = ADC_measure();      //Przypisanie wartosci odczytanej do x
110
111     a = x/100;              //Obliczenie wartosci setek
112     b = x - 100 * a;        //Obliczenia wartosci po przecinku
113
114     LCD_GoTo(0,1);
115     sprintf(text , " Pomiar: %"PRIu32".%"PRIu32" V" , a, b );       //
            Wyswietlenie pomiaru
116     LCD_WriteText(text);
117 }
118
119 //Tablica zankow przechowujaca liczby do wyswietlacza
120 char cyfra[11] = { 0b1111110, 0b0110000, 0b1101101, 0b1111001, 0b0110011,
121         0b1011011, 0b1011111, 0b1110000, 0b1111111, 0b1111011, 0b0000001};
122
123 //Inicjalizacja portow do obs ugi wyswietlacza 7 segmentowego
124 void seg7Init() {
125     //Inicjalizacja segmentu
126     DDRC = 0xff;
127     PORTC = 0x00;
128 }
129
130 //Wyswietla na wyswietlaczu 7 segmentowym cyfre z argumentu
131 void seg7ShowCyfra(uint8_t cyfraDoWyswietlenia) {
132     PORTC = cyfra[cyfraDoWyswietlenia];
133 }
134
135 //Inicjalizacja Timer1 do wywolywania przerwania z cz stotliwo ci  100Hz
136 void TimerInit() {
137     //Wybranie trybu pracy CTC z TOP OCR1A
138     sbi(TCCR1B,WGM12);
139     //Wybranie dzielnika czestotliwosci
140     sbi(TCCR1B,CS12);
141     //Zapisanie do OCR1A wartosci odpowiadajacej 0,01s
142     OCR1A=312.5;
143     //Uruchomienie przerwania OCIE1A
144     sbi(TIMSK,OCIE1A);
145 }
```

```
146
147 //Inicjalizacja Zegara do wywolywania przerwania z cz stotliwo ci  1Hz
148 void ZegarInit() {
149     //Wybranie trybu pracy CTC z TOP OCR1A
150     sbi(TCCR1B,WGM12);
151     //Wybranie dzielnika czestotliwosci
152     sbi(TCCR1B,CS12);
153     //Zapisanie do OCR1A wartosci odpowiadajacej 1s
154     OCR1A=31250;
155     //Uruchomienie przerwania OCIE1A
156     sbi(TIMSK,OCIE1A);
157 }
158
159 //Funcja wyswietla informacje o programie
160 void info(){
161
162     char text[20];          //Tablica znakow
163     int wyjdz = 1;          //Zmienna pomocnicza
164
165     LCD_Clear();
166     sprintf(text , "Program PTM 2021" );     //Wyswietlenie napisu
167     LCD_GoTo(0,0);
168     LCD_WriteText(text);
169
170     sprintf(text , "252919, 252870" );       //Wyswietlenie numerow indeksow
171     LCD_GoTo(0,1);
172     LCD_WriteText(text);
173     for(int i = 1; (i <= 800)&&(wyjdz); i++){       //Petla umozliwia nagle
            wyjscie z funkcji
174
175
176         cbi(PORTD, 0);                              //Ustawia wyj cie stan
                niski
177         _delay_ms(5);
178                                                     //Sprawdza ktory
                                                        przycisk jest
                                                        wlaczony
179         if(bit_is_clear(PIND,2)){
180             wyjdz = 0;
181         }
182         sbi(PORTD,0);
183     }
184 }
185
186 //Funkcja liczby
187 void liczby(){
188
189     int wyjdz = 1;                  //Zmienna pomocnicza
190     int pierwsza = 1;               //Zmienna pomocnicza
191     uint8_t a = 0;                  //Zmienna pomocnicza
```

```
192     char text[20];                  //Tablica znakow
193
194     seg7Init();                     //Uruchamia funkcje seg7Init()
195
196     while(1 && wyjdz){                          //Petla sprawdzajaca czy
            zostal wcisniety X
197         _delay_ms(100);
198
199         cbi(PORTD, 0);                          //Ustawia wyj cie stan
                niski
200         _delay_ms(5);
201                                                 //Sprawdza ktory przycisk
                                                        jest wlaczony
202         if(bit_is_clear(PIND,2)){
203             wyjdz = 0;
204         }
205         if(bit_is_clear(PIND,3)){               //Przycisk up
206             if(a == 50){
207                 a = 50;
208             }
209             else{
210                 a++;
211             }
212         }
213         sbi(PORTD, 0);                          //Ustawia wyj cie stan
                wysoki
214
215         cbi(PORTD, 1);                          //Ustawia wyj cie stan
                niski
216         _delay_ms(5);
217                                                 //Sprawdza ktory przycisk
                                                        jest wlaczony
218         if(bit_is_clear(PIND,3)){
219             if(a == 0){
220                 a = 0;
221             }
222             else{
223                 a--;                            //Przycisk down
224             }
225         }
226         sbi(PORTD, 1);                          //Ustawia wyj cie stan
                wysoki
227
228         LCD_Clear();
229         LCD_GoTo(0,0);
230         sprintf(text , "Liczba: %"PRIu8" ", a);     //Wypisanie wartosci
                danej liczby
231         LCD_WriteText(text);
232
233         if(a > 9){
```

```
234             seg7ShowCyfra(10);                          //Wyswietlenie "-"
235         }
236         else{
237             seg7ShowCyfra(a);                           //Wyswietla liczby od
                    1-9 na wyswietlaczu 7 segmentowym
238         }
239
240         if(a % 2){                                      //Sprawdza czy liczba
                jest nieparzysta
241             cbi(PORTA,4);                               //Gasi LED 1
242             sbi(PORTA,5);                               //Zapala LED 2
243         }
244         else{                                           //Liczba jest parzysta
245             sbi(PORTA,4);                               //Zapala LED 1
246             cbi(PORTA,5);                               //Gasi LED 2
247         }
248
249         if(a < 2){                                      //Jezeli liczba jest
                mniejsza od 1 to nie jest liczba pierwsza
250             pierwsza = 0;
251         }
252
253         if(a >= 2){
254             for(uint8_t i = 2; i*i <= a; i++)
255                 if(a % i == 0){
256                     pierwsza = 0;          //gdy znajdziemy dzielnik, to dana
                            liczba nie jest pierwsza
257                 }
258         }
259         if(pierwsza == 0){                  //Jezeli liczba nie jest licza
                pierwsza
260             cbi(PORTA,6);                   //Gasi LED 3
261             pierwsza = 1;
262         }
263         else{
264             sbi(PORTA,6);                   //Zapala LED 3
265         }
266     }
267
268     cbi(PORTA,4);                           //Ustawia stan niski na A4
269     cbi(PORTA,5);                           //Ustawia stan niski na A5
270     cbi(PORTA,6);                           //Ustawia stan niski na A6
271     PORTC = 0x00;                           //Ustawia stan niski na portach C
272 }
273
274 //Funkcja liczy czas z dokladnoscia do 0,01s
275 void stoper () {
276
277     TimerInit();                            //Uruchamia funkcje TimerInit()
278
```

```
279     char text[20];                          //Tablica znakow
280     uint32_t ms = 0;                        //Zmienna pomocnicza
281     uint32_t s = 0;                         //Zmienna pomocnicza
282     uint32_t k = 1;                         //Zmienna pomocnicza
283     int licznik = 0;                        //Zmienna pomocnicza
284     int kasuj = 1;                          //Zmienna pomocnicza
285     int zatrzymaj = 0;                      //Zmienna pomocnicza
286     int wyjdz = 1;                          //Zmienna pomocnicza
287
288
289
290     sprintf(text , "Stoper: 00:00");            //Wypisanie na wyswietlaczu
            00:00
291     LCD_GoTo(0,0);
292     LCD_Clear();
293     LCD_WriteText(text);
294     _delay_ms(80);
295
296     while(1 && wyjdz){                                  //Petla sprawdza
            czy zostal wcisniety przycisk X
297         while((k != i) && zatrzymaj && wyjdz){          //Petla sprawdza
                czy zostal wcisniety przycisk X lub OK
298             k = i;                                      //Przypisanie
                    wartosci z przerwania
299             ms = i - licznik * 100;                     //Obliczanie
                    dokladnej wartosci setnych sekund
300
301             if((i - licznik * 100) >= 100){
302                 tbi(PORTA,4);                           //Ustrawia stan
                        niski i wysoki na porcie A4
303                 licznik++;                              //Zwiekszenie
                        zakresu licznik
304                 s++;                                    //Zwiekszenie
                        zakresu s
305                 ms = i-licznik*100;                     //Obliczenie
                        wartosci milisekund
306             }
307
308             if(s == 60){                                //Jezli dojdzie do
                    60 sek
309                 s = 0;                                  //Zeruje wartosc s
310             }
311
312             if(s < 10){                                 //Umozliwia
                    poprawne wyswietlanie stopera dla wartosi mniejszych od 10
313                 sprintf(text , "Stoper: 0%"PRIu32":%"PRIu32" ",s ,ms );
314                 LCD_GoTo(0,0);
315                 LCD_WriteText(text);
316                 _delay_ms(80);
317             }
```

```
318             else{                                           //Umozliwia
                    poprawne wyswietlanie stopera dla wartosi wiekszych od 10
319                 sprintf(text , "Stoper: %"PRIu32":%"PRIu32" ",s ,ms );
320                 LCD_GoTo(0,0);
321                 LCD_WriteText(text);
322                 _delay_ms(80);
323             }
324             cbi(PORTD, 0);                                  //Ustawia wyj cie
                    stan niski
325             _delay_ms(5);
326                                                             //Sprawdza ktory
                                                                przycisk jest
                                                                wlaczony
327             if(bit_is_clear(PIND,2)){
328                 wyjdz = 0;
329             }
330             sbi(PORTD,0);
331
332             cbi(PORTD, 1);                                  //Ustawia wyj cie
                    stan niski
333             _delay_ms(5);
334                                                             //Sprawdza ktory
                                                                przycisk jest
                                                                wlaczony
335             if(bit_is_clear(PIND,2)){
336                 zatrzymaj = 0;
337                 kasuj = 0;
338                 _delay_ms(100);
339             }
340             sbi(PORTD,1);
341         }
342
343         cli();                                          //Wylacza przerywanie
344
345         cbi(PORTD, 0);                                  //Ustawia wyj cie stan
                niski
346         _delay_ms(5);
347                                                         //Sprawdza ktory
                                                            przycisk jest
                                                            wlaczony
348         if(bit_is_clear(PIND,2)){
349             wyjdz = 0;
350         }
351         sbi(PORTD,0);
352
353         if(kasuj == 1){                                 //Sprawdzenie
                warunkow
354             cbi(PORTD, 1);                              //Ustawia wyj cie
                    stan niski
```

```
355                 _delay_ms(5);                                   //Sprawdza ktory
                        przycisk jest wlaczony
356                 if(bit_is_clear(PIND,2)){
357                     zatrzymaj = 1;
358                     _delay_ms(50);
359                     sei();                                      //funkcja uruchamia
                            globalne przerwania
360                 }
361                 sbi(PORTD,1);
362             }
363             else{
364                 cbi(PORTD, 1);                                  //Ustawia wyj cie
                        stan niski
365                 _delay_ms(5);                                   //Sprawdza ktory
                        przycisk jest wlaczony
366                 if(bit_is_clear(PIND,2)){
367                     kasuj = 1;                                  //Wyzerowanie
                            wartosci
368                     i = 0;
369                     ms = 0;
370                     s = 0;
371                     licznik = 0;
372
373                     sprintf(text , "Stoper: 00:00");            //Wyswietlenie
                            zresetowanego stopera
374                     LCD_GoTo(0,0);
375                     LCD_Clear();
376                     LCD_WriteText(text);
377                     _delay_ms(100);
378                 }
379                 sbi(PORTD,1);
380
381             }
382
383         }
384
385     }
386
387     //Funcja pelni role zegara zliczajacego czas z dokladnoscia do 1sek za
            pomoca przerwan
388     void zegar(){
389
390         ZegarInit();                    //Uruchamia funkcje ZegarInit()
391         seg7Init();                     //Uruchamia funkcje seg7Init()
392         i = 0;                          //Wyzerowanie zmiennej globalnej
393         int wyjdz = 1;                  //Zmienna pomocnicza
394         int k = 0;                      //Zmienna pomocnicza
395         int j = 0;                      //Zmienna pomocnicza
396         char text[20];                  //Zmienna pomocnicza
397         uint32_t m = 0;                 //Zmienna pomocnicza
```

```
398
399         LCD_GoTo(0,0);
400         LCD_Clear();                        //Czyszczenie wyswietlacza
401
402         sei();                              //Uruchamienie przewania
403         while(1 && wyjdz){
404
405             cbi(PORTD, 0);                          //Ustawia wyj cie stan niski
406             _delay_ms(5);
407                                                         //Sprawdza ktory
                                                               przycisk jest
                                                               wlaczony
408             if(bit_is_clear(PIND,2)){
409                 wyjdz = 0;
410             }
411             sbi(PORTD, 0);                          //Ustawia wyj cie stan wysoki
412
413             if(wyjdz == 0){                         //Jezeli zostal wcisniety
                    przycisk X
414                 PORTC = 0x00;                       //Gasi wyswietlacz 7 segmentowy
415             }
416
417
418             while(1 && (k != i)){                   //Petla jest uruchamiana
                    dokladnie co 1sek
419                 j++;                                //Zwiekszenie zakresu o 1
420                 k = i;                              //Podstawienie wartosci i pod
                        zmienna k
421                 if(i > 59){                         //Warunek umozliwia poprawne
                        wyswietlanie sekund
422                     m++;                            //Zwiekszenie ilosci minut o 1
423                     i = 0;                          //Wyzerowanie zmiennej
424                 }
425                 if(m > 59){                         //Warunek umozliwia poprawne
                        wyswietlanie minut
426                     m = 0;                          //Wyzerwoanie zmiennej
427                 }
428
429                 if(j < 10){                         //Sprawdza czy j jest mniejsze
                        od 10
430                     seg7ShowCyfra(j);               //Wyswietla liczbe od 0 do 9
431                     if(j == 9){                     //Warunek sprawdza czy j jest
                            rowne 9
432                         j = -1;                     //Podtsawinie -1 pod zmienna j
433                     }
434                 }
435
436                 if(i < 10 && m < 10){               //Umozliwia poprawen
                        wyswietlanie zegara
```

```
437                 sprintf(text ,"Zegar: 0%"PRIu32":0%"PRIu32" ",m ,i);
                        //Wyswietlanie godziny
438                 LCD_GoTo(0,0);
439                 LCD_WriteText(text);
440
441             }
442             else if(i < 10){                //Umozliwia poprawen
                    wyswietlanie zegara
443                 sprintf(text ,"Zegar: %"PRIu32":0%"PRIu32" ",m ,i);
                        //Wyswietlanie godziny
444                 LCD_GoTo(0,0);
445                 LCD_WriteText(text);
446
447             }
448             else if(m < 10){                //Umozliwia poprawen
                    wyswietlanie zegara
449                 sprintf(text ,"Zegar: 0%"PRIu32":%"PRIu32" ",m ,i);
                        //Wyswietlanie godziny
450                 LCD_GoTo(0,0);
451                 LCD_WriteText(text);
452
453             }
454             sbi(PORTA,6);                       //Ustawia stan wysoki na
                    porcie A6
455             _delay_ms(200);
456             cbi(PORTA,6);                       //Ustawia stan niski na
                    porcie A6
457         }
458     }
459 }
460
461 //Funcja mierzy napi cie
462 void miernik(){
463
464     ADC_init();                                 //Inicjalizacja
            pomiaru
465
466     int wyjdz = 1;                              //Podtsawienie 1
            pod zmienna wyjdz
467     uint32_t c = 0;                             //Zmienna
            pomocnicza
468     char text[20];                              //Tablica znakow
469
470     while(1 && wyjdz) {                         //Petla dziala tak
            dlugo az zostanie wcisniety przycisk X
471         c = ADC_10bit();                        //Zwraca wartosc
                pomiaru
472         LCD_Clear();
473         LCD_GoTo(0,0);
```

```
474                 sprintf(text , " ADC: %"PRIu32" " , c );     //Wystwielenie
                        pomiaru na wyswietlaczu
475                 LCD_WriteText(text);
476
477                 LCD_GoTo(0,1);
478                 wyswietl();
479
480                 cbi(PORTD, 0);                                          //Ustawia
                        wyj cie stan niski
481                 _delay_ms(5);
482                                                                         //Sprawdza
                                                                            ktory
                                                                            przycisk
                                                                             jest
                                                                            wlaczony
483                 if(bit_is_clear(PIND,2)){
484                     wyjdz = 0;                                          //
                            Podstawienie 0 pod zmienna wyjdz
485                 }
486                 sbi(PORTD,0);                                           //Ustawia
                        stan wysoki na porcie D0
487
488
489                 _delay_ms(100);
490             }
491 }
492
493 //Funkcja umozliwia wybranie odpowiedniego programu
494 void menu(){
495
496     char text[20];              //Tablica znakow
497     int i = 1;
498     uint16_t ok = 0;
499
500     sbi(DDRD, 0);               //Wyj cie stan wysoki porty PD0-PD1
501     sbi(PORTD, 0);
502     sbi(DDRD, 1);
503     sbi(PORTD, 1);
504
505                                 //Ustawienie PD2-PD3 jako PULL-UP
506     cbi(DDRD, 2);
507     sbi(PORTD, 2);
508     cbi(DDRD, 3);
509     sbi(PORTD, 3);
510
511     while(1){
512         _delay_ms(100);
513
514         cbi(PORTD, 0);                      //Ustawia wyj cie stan niski
515         _delay_ms(5);
```

```
516                                             //Sprawdza ktory przycisk jest
                                                    wlaczony
517         if(bit_is_clear(PIND,2)){
518
519         }
520         if(bit_is_clear(PIND,3)){           //Przycisk up
521             i--;
522             if(i < 0){
523                 i = 0;
524             }
525         }
526         sbi(PORTD, 0);                      //Ustawia wyj cie stan wysoki
527
528
529         cbi(PORTD, 1);                      //Ustawia wyj cie stan niski
530         _delay_ms(5);
531                                             //Sprawdza ktory przycisk jest
                                                    wlaczony
532         if(bit_is_clear(PIND,2)){
533             ok = 1;
534         }
535         if(bit_is_clear(PIND,3)){
536             i++;                            //Przycisk down
537             if(i > 5){
538                 i = 5;
539             }
540         }
541         sbi(PORTD, 1);                          //Ustawia wyj cie stan
                wysoki
542
543         if(i == 1){                             //Sprawdza wybrana opcje
544
545             LCD_Clear();
546             sprintf(text , "1. Info" );         //Wyswietla na wyswietlaczu
                    napis "1. Info"
547             LCD_GoTo(0,0);
548             LCD_WriteText(text);
549             _delay_ms(10);
550
551             if(ok == 1){                        //Uruchamia dana funkcje po
                        klknieciu przycisku ok
552                 info();                         //Uruchamia funkcje info()
553                 ok = 0;                         //Wyzerowanie zmiennej ok
554             }
555         }
556         if(i == 2){                             //Sprawdza wybrana opcje
557
558             LCD_Clear();
559             sprintf(text , "2. Liczby" );       //Wyswietla na wyswietlaczu
                    napis "2. Liczby"
```

```
560             LCD_GoTo(0,0);
561             LCD_WriteText(text);
562             _delay_ms(10);
563 
564             if(ok == 1){                        //Uruchamia dana funkcje po
                    klikniecie przycisku ok
565                 liczby();                       //Uruchamia funkcje liczby
                        ()
566                 ok = 0;                         //Wyzerowanie zmiennej ok
567             }
568         }
569         if(i == 3){                             //Sprawdza wybrana opcje
570 
571             LCD_Clear();
572             sprintf(text , "3. Stoper" );       //Wyswietla na wyswietlaczu
                    napis "3. Stoper"
573             LCD_GoTo(0,0);
574             LCD_WriteText(text);
575             _delay_ms(10);
576 
577             if(ok == 1){                        //Uruchamia dana funkcje po
                    klikniecie przycisku ok
578                 stoper();                       //Uruchamia funkcje stoper
                        ()
579                 ok = 0;                         //Wyzerowanie zmiennej ok
580             }
581         }
582         if(i == 4){                             //Sprawdza wybrana opcje
583 
584             LCD_Clear();
585             sprintf(text , "4. Zegar" );        //Wyswietla na wyswietlaczu
                    napis  "4. Zegar"
586             LCD_GoTo(0,0);
587             LCD_WriteText(text);
588             _delay_ms(10);
589 
590             if(ok == 1){                        //Uruchamia dana funkcje po
                    klikniecie przycisku ok
591                 zegar();                        //Uruchamia funkcje zegar()
592                 ok = 0;                         //Wyzerowanie zmiennej ok
593             }
594         }
595         if(i == 5){
596 
597             LCD_Clear();
598             sprintf(text , "5. Miernik" );      //Wyswietla na wyswietlaczu
                    napis "5. Miernik"
599             LCD_GoTo(0,0);
600             LCD_WriteText(text);
601             _delay_ms(10);
```

```
602
603             if(ok == 1){                        //Uruchamia dana funkcje po
                    klikniecie przycisku ok
604                 miernik();                      //Uruchamia funkcje miernik
                        ()
605                 ok = 0;                         //Wyzerowanie zmiennej ok
606             }
607         }
608     }
609 }
610
611
612 int main(){
613
614     LCD_Initalize();        //Inicjalizacja LCD
615     LCD_Home();
616     LCD_Clear();            //Czyszczenie wyswietlacza
617
618
619
620
621
622     sbi(DDRA,4);            //Ustawia stanu wysokiego na DDRA 4
623     sbi(DDRA,5);            //Ustawia stanu wysokiego na DDRA 5
624     sbi(DDRA,6);            //Ustawia stanu wysokiego na DDRA 6
625
626     info();                 //Uruchamia funkcje INFO
627
628     menu();                 //Uruchamia funkcje MENU
629
630     return 0;               //Konczy program
631
632 }
633
634 //Funkcja uruchamiana z przerwaniem po przepelnieniu licznika w timer1
635 ISR(TIMER1_COMPA_vect) {
636     i++;                    //Zwiekszenie zakresu o 1 co dana ilosc sekund
637 }
```