5. Projekt części elektronicznej urządzenia

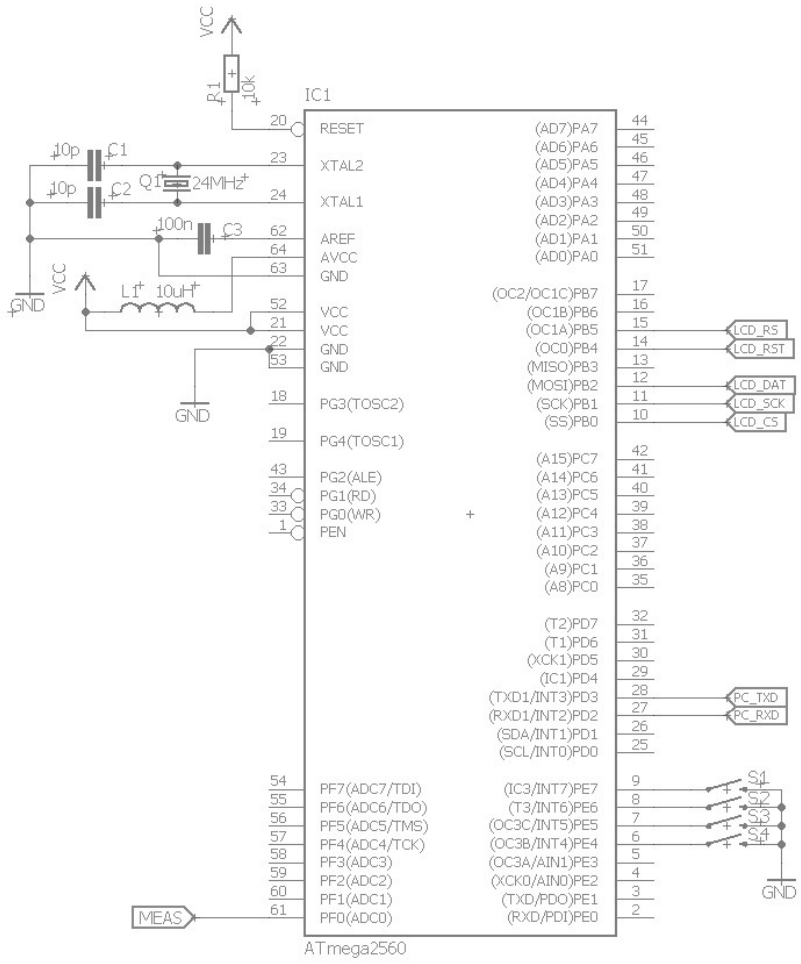
W niniejszym rozdziale przedstawiona została budowa części elektronicznej zaprojektowanego analizatora ograniczona do kluczowych elementów. Omówione zostały sposoby ich połączenia oraz rola w układzie.

5.1 Mikrokontroler do DSP

Jako główny mikrokontroler liczący FFT wybrany został układ scalony firmy Atmel o oznaczeniu ATmega2560[16]. Jest to jeden z najlepiej wyposażonych układów z tej rodziny mikrokontrolerów. Został wybrany z powodu posiadania 8KB pamięci SRAM, w której przechowywane będą próbki analizowanego sygnału. Posiada on w swojej strukturze wbudowany przetwornik A/C o dziesięciobitowej rozdzielczości pomiaru i działa na zasadzie sukcesywnej aproksymacji, która została opisana w rozdziale drugim. Czas pojedynczego pomiaru wynosi 12 taktów zegarowych, a maksymalna częstotliwość jego taktowania to 1MHZ (przy ośmiobitowej rozdzielczości pomiaru), co pozwala osiągnąć próbkowanie na poziomie 75kSPS. Mikrokontroler według dokumentacji może być taktowany z maksymalną częstotliwością 20 [MHz] jednak w przypadku, gdy nie jest używana wbudowana pamięć EEPROM można go przetaktować do 25 [MHz] bez widocznych wpływów na jego pracę. Sam rdzeń posiada zestaw instrukcji typu RISC i oparty jest o architekturę harwardzką, czyli każda z pamięci adresowana jest w oddzielnej przestrzeni adresowej. Posiada 32 rejestry ogólnego przeznaczenia współpracujące z ośmiobitową jednostką ALU, która nie wspiera sprzętowych operacji na liczbach zmiennoprzecinkowych. Dodatkowo układ wyposażony jest w szereg sprzętowych interfejsów w tym USART oraz SPI, które posłużą do komunikacji z komputerem oraz wysyłania danych do wyświetlacza LCD.

Schemat połączeń mikrokontrolera został przedstawiony na rysunku 5.1. Port B służy do komunikacji mikrokontrolera z wyświetlaczem przez sprzętową magistralę szeregową SPI. Na czterech pinach portu E dołączone zostają przyciski, którymi użytkownik będzie zmieniał parametry pracy programu. Wyjścia PD2 oraz PD3 wewnętrznie połączone są ze sprzętowym interfejsem USART i służą do komunikacji z komputerem poprzez magistralę RS232. Do wejść oscylatora XTAL dołączony zostaje kwarc o częstotliwości 24MHz, co pozwoli osiągnąć teoretyczną moc obliczeniową 24MIPS. Do wejścia PF0 doprowadzony zostaje mierzony sygnał, który będzie przetwarzany przez wewnętrzny przetwornik A/C. Reszta elementów są to

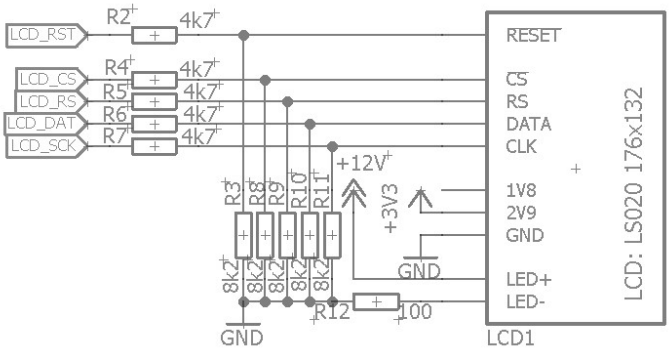
komponenty niezbędne do poprawnej pracy według dokumentacji mikrokontrolera, skąd zostały też zaczerpnięte ich wartości.



Rysunek 5.1 Układ połączeń mikrokontrolera[opracowanie własne]

5.2 Wyświetlacz LCD

Jako komponent przedstawiający wyniki analizy dobrany został kolorowy wyświetlacz LCD ze sterownikiem LS020B8UD06 posiadający rozdzielczość 176 na 132 piksele z szesnastobitową głębią kolorów. Schemat jego połączeń przedstawia rysunek 5.2.

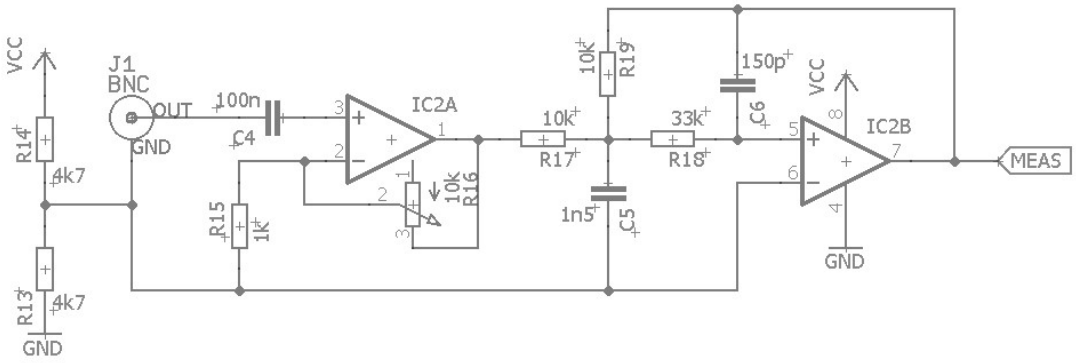


Rysunek 5.2 Układ połączeń mikrokontrolera[opracowanie własne]

Komunikacja z mikrokontrolerem odbywa się na różnych poziomach napięć, dlatego konieczne jest zastosowanie układów dopasowujących w postaci dzielników rezystorowych. Pozwalają one na zmniejszenie napięcia 5 [V] wystawianego przez porty mikrokontrolera do wartości 2,9 [V], które jest akceptowalne przez sterownik wyświetlacza.

5.3 Tor pomiarowy

Tor pomiarowy został oparty o dwa wzmacniacze operacyjne pozwalające na wykonanie dolnoprzepustowego filtru antyaliasingowego oraz wzmacniacza sygnału wejściowego co przedstawia poniższy schemat.



Rysunek 5.3 Tor pomiarowy[opracowanie własne]

Sygnał wejściowy podawany jest na złącze J1. Rezystory R14 oraz R13 tworzą dla niego układ sztucznej masy na poziomie połowy napięcia zasilania. Konieczność stosowania takiego rozwiązania spowodowana jest tym, że używany przetwornik A/C nie potrafi mierzyć napięć ujemnych. W dalszej kolejności poprzez kondensator C4 odcinana jest składowa stała sygnału i trafia on na pierwszy stopień wzmacniacz IC2A. Jego wzmocnienie regulowane jest potencjometrem R16 i zawiera się w przedziale od 1 do 11. Drugi stopień to dolnoprzepustowy

filtr antyaliasingowy z wielokrotnym sprzężeniem zwrotnym (typu LP-MFB) o charakterystyce Czebyszewa i spadku charakterystyki 18dB na oktawę. Wartości elementów z których został zbudowany zostały wyznaczone tak, aby częstotliwość graniczna wynosiła 25 [kHz].

5.4 Podsumowanie

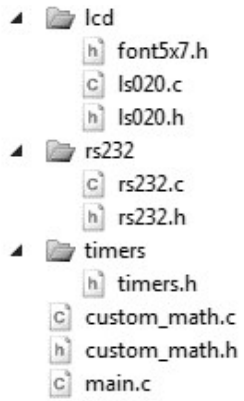
Przedstawiona część sprzętowa urządzenia zawiera jedynie elementy niezbędne do jego uruchomienia. Pominięte zostały łącznik pomiędzy komputerem PC, a analizatorem w postaci konwertera poziomów napięć magistrali RS232 oraz źródła zasilania wytwarzające napięcia zasilania 3,3 [V] oraz 5 [V].

6. Oprogramowanie analizatora

W niniejszym rozdziale opisana została struktura przygotowanego oprogramowania dla mikrokontrolera. Przedstawiono kluczowe dla działania programu funkcje i algorytmy oraz opisano ich parametry.

6.1 Środowisko programistyczne i struktura programu

Oprogramowanie zostało napisane w języku C przy użyciu środowiska programistycznego Atmel Studio w wersji 7.0[7][8]. Kompilacja odbywa się przy pomocy kompilatora avr-gcc w wersji 5.4.0. Struktura projektu przedstawiona została na poniższym rysunku.



Rysunek 6.1 Struktura plików projektu

Program został podzielony na pojedyncze pliki zawierające funkcje realizujące określone zadania. W pliku *main.c* znajduje się główna pętla sterująca całą pracą programu oraz funkcje obsługujące inicjalizację i wyświetlanie obrazu na ekranie. Plik *ls020.c* zawiera niskopoziomowe funkcje do komunikacji z wyświetlaczem LCD przez interfejs szeregowy oraz prostą bibliotekę graficzną do rysowania kształtów i wypisywania liter. W pliku *r232.c* znajdują się funkcje do komunikacja urządzenia z komputerem PC przez interfejs COM. Plik *custom_math.c* zawiera zoptymalizowane algorytmy do wyznaczania funkcji trygonometrycznych, tablice funkcji okien oraz funkcje obliczające dyskretną szybką transformatę Fouriera.

6.2 Komunikacja z wyświetlaczem LCD

Przesyłania danych poprzez magistralę SPI do wyświetlacza odbywa się przy użyciu funkcji przedstawionych na poniższym listingu.

```
13  void spi_init(void);
14  void spi_send_byte(uint8_t byte);
15  void spi_send_word(uint16_t word);
16  void lcd_command_byte(uint8_t command_byte);
17  void lcd_command_word(uint16_t command_word);
18  void lcd_data_byte(uint8_t data_byte);
19  void lcd_data_word(uint16_t data_word);
```

Listing 6.2 Niskopoziomowe funkcje do komunikacji z wyświetlaczem.

Funkcja *spi_init()* inicjalizuje magistralę do pracy w trybie master i ustawia jej częstotliwość taktowania na 12MHz. Kolejne dwie funkcje służą do wysyłania pojedynczego bajtu oraz dwóch bajtów przez magistralę. Funkcje nazwane jako *command* przesyłają instrukcje sterujące do rejestrów wyświetlacza, natomiast oznaczone przez data informacje odnośnie kolorów kolejnych pikseli.

Do czyszczenia zawartości wyświetlacza, rysowania prostych kształtów oraz wypisywania tekstu napisane zostały funkcje przedstawione na listingu 6.3.

```
23  void lcd_init(void);
24  void lcd_clear(uint16_t colour);
25  void lcd_pixel(uint8_t x_pos, uint8_t y_pos, uint8_t colour);
26  void lcd_set_pos(uint8_t x_pos, uint8_t y_pos);
27  void lcd_set_area(uint8_t x1, uint8_t x2, uint8_t y1, uint8_t y2);
28  void lcd_rectangle(uint8_t x1, uint8_t y1, uint8_t x2, uint8_t y2,
29                     uint8_t colour);
30  void lcd_frame(uint8_t x_pos, uint8_t y_pos, uint8_t width, uint8_t height,
31                 char colour);
32  void lcd_char_5x7(uint8_t x_pos, uint8_t y_pos, char character, uint8_t colour,
33                    uint8_t background);
34  void lcd_char_10x14(uint8_t x_pos, uint8_t y_pos, char character,
35                      uint8_t colour, uint8_t background);
36  void lcd_string(uint8_t x_pos, uint8_t y_pos, char *string, uint8_t colour,
37                  uint8_t background, uint8_t size);
38  void lcd_line(uint8_t x1, uint8_t y1, uint8_t x2, uint8_t y2, uint16_t colour);
```

Listing 6.3 Funkcje graficzne LCD.

Większość z nich jako parametry przyjmuje współrzędne położenia rysowanego elementu na ekranie, jego kolor, tło oraz rozmiar. Powstały w ten sposób zestaw funkcji pozwala na utworzenie interfejsu użytkownika oraz kreślenia widma badanych sygnałów.

6.3 Komunikacja z komputerem PC

Jako interfejs komunikacyjny urządzenia z komputerem wybrana została magistrala RS232. Obsługę wysyłania danych realizują funkcje przedstawione na listingu 6.4.

```
44  #define BAUD 9600
45  void rs232_init(void);
46  void rs232_send_byte(uint8_t byte);
47  void rs232_send_string(const char* s);
48  void rs232_send_int(int i);
49  void send_mag_via_uart();
```

Listing 6.4 Funkcje graficzne LCD.

Definicja BAUD określa prędkość przesyłania danych w bitach na sekundę. Jej maksymalna wartość dla tego typu mikrokontrolerów może wynosić 2Mb/s[12]. Pierwsza funkcja służy do inicjalizacji magistrali z podaną wcześniej prędkością w trybie 8N1, czyli ramka składa się z ośmiu bitów, brak jest bitu parzystości i kończy się ona jednym bitem stopu. Jest to najczęściej stosowany tryb komunikacji przez magistralę RS232. Kolejne trzy funkcje służą do wysyłania danych w postaci pojedynczego bajtu, łańcucha znaków oraz zmiennej typu całkowitego. Ostatnia obsługuje wysłanie całego bufora wynikowego z widmem sygnału w postaci ramki. Ramka składa się z flagi początku w postaci łańcucha znaków AT+START_SEND, następnie z ciągu wartości do wysłania oddzielonych ze sobą spacją. Zakończona jest natomiast łańcuchem AT+STOP_SEND. Podejście takie pozwoli na oddzielenie kolejnych ramek po stronie komputera i dzięki temu odpowiednie ich kreślenie na wykresie.

6.4 Funkcje matematyczne

Z racji tego, że wykorzystany mikrokontroler nie posiada instrukcji realizujących obliczenia na liczbach zmiennoprzecinkowych konieczne było stworzenie algorytmów bazujących na obliczeniach stałoprzecinkowych. W pliku *custom_math.c* zaimplementowane zostały funkcje przedstawione w listingu 6.5.

Funkcja sinus() zwraca wartość funkcji sinus przemnożoną przez wartość 32767, co pozwala wyeliminować konieczność korzystania ze zmiennych typu float. Dodatkowa wartości zwracane przez funkcje zostały wyznaczone poza programem i umieszczone w tablicy znajdującej się w pamięci programu *sin_tab[].* Tablica ta zawiera jedynie 256 wartości funkcji sinus z przedziału od 0 do Π/2, a reszta wartości wyznaczana jest z własności, że *sin(x+pi)=-sin(x)* oraz *sin(k)=sin(pi-k),* gdy *k* należy do przedziału od 0 do Π/2. Dzięki takiej operacji tablica zawierająca wartości funkcji sinus mogła zostać zmniejszona czterokrotnie.

```
59  const int16_t sin_tab[256] PROGMEM;
60  const int16_t hanning_table[1024] PROGMEM;
61  const int16_t hamming_table[1024] PROGMEM;
62  const int16_t triangle_table[1024] PROGMEM;
63  const int16_t flat_top_table[1024] PROGMEM;
64
65  int16_t sinus(int k);
66  int16_t cosinus(int k);
67  void fix_fft(int16_t fr[], int16_t fi[], int16_t m);
68  void fix_fftr(int16_t f[], int m);
69  void fix_fft_mag(int16_t f[], int16_t k);
70  int16_t window(int16_t val,int8_t w, int16_t idx);
71  uint16_t square_root(uint32_t n);
```

Listing 6.5 Funkcje matematyczne z pliku custom_math.c.

Kolejna funkcja zwraca wartość funkcji cosinus korzystając z własności, że funkcje te są przesunięte między sobą o Π/2. Następne trzy funkcje służą do obliczania wartości szybkiej transformaty Fouriera kolejno dla liczb zespolonych, rzeczywistych oraz wyznaczenia wartości skutecznej poszczególnych składowych. Algorytm wyznaczania widma został specjalnie zoptymalizowany pod względem obliczeń na dwubajtowych zmiennych całkowitych, przez co z każdym kolejnym wyznaczaniem motylka jego wartość jest skalowana przy użyciu funkcji *fix_mpy()*. Sam algorytm to implementacja transformaty radix 2 z decymacją w dziedzinie czasu[17]. Do mnożenia sygnału przez okna czasowe służy funkcja *window()*. Jej wywołanie musi zachodzić podczas zbierania próbek sygnału poddawanego transformacie. Podejście to pozwala na zaoszczędzenie czasu jednego przejścia pętli po wszystkich próbkach i ich przemnożeniu przez funkcje okna, gdyż dzieje się to w momencie, gdy program i tak oczekuje na pobranie kolejnej wartości próbki. Wartości mnożników dla próbek zostały obliczone przy pomocy wzorów przedstawionych w tabeli 3.1 i umieszone w tablicach od dwa do pięć z listingu 6.5. Ostatnia funkcja jest implementacją szybkiego obliczania pierwiastka kwadratowego z czterobajtowej liczby całkowitej[6]. Jako, że dzielenie oraz mnożenie liczb większych niż jednobajtowe wymagają kilkaset taktów procesora napisany został algorytm wyznaczający pierwiastek jedynie w oparciu o instrukcje wykonywane w jednym cyklu zegarowym, czyli przesunięcia bitowe, dodawanie, odejmowanie oraz sumę logiczną.

6.5 Zbieranie próbek

Analizowany sygnał mierzony jest przez wbudowany w mikrokontroler przetwornik AC. Funkcja *adc_init()* konfiguruje go w tryb pracy ciągłej. Oznacza to, że pod odczytaniu wartości wyniku konwersji automatycznie wykonywana jest kolejna konwersja. Napięcie referencyjne ustawiane jest na wartość napięcia zasilania co pozwoli mierzyć sygnały z przedziału od 0 do 5 [V]. Ustawiona zostaje również wartość taktowania przetwornika i wynosi ona 750 [kHz]

i przy 12 taktach zegarowych potrzebnych na dokonanie jednej konwersji pozwala to na zbieranie próbek z częstotliwością 62,5kSPS. Samo zbieranie próbek do tablicy *samples[]* wykonuje szesnastobitowy timer1. Jego taktowanie oraz sposób zliczania zostały dobrane tak, aby pobierał on wartość wyniku konwersji 1024 razy w ciągu 20 [ms]. Przerwanie timera1 przedstawiające sposób zbierania próbek przedstawia listing 6.6.

Pierwszą instrukcją w przerwaniu musi być pobranie wartości z rejestru *ADCW*, czyli wyniku konwersji przetwornika AC. Dzieje się tak dlatego, że timer1 pobiera wartość 1024 razy w ciągu sekundy, a przetwornik pracując na granicy swojej wydajności jest w stanie dokonać maksymalnie 1250 konwersji w ciągu zapełniania całej tablicy z próbkami. Następnie sprawdzane jest, czy właśnie została włączona procedura zapełniania bufora i jeżeli jest to pierwsza konwersja, to wynik zostaje odrzucony. W dalszej kolejności następuje przemnożenie wartości przez funkcję okna czasowego w zależności od wybranego typu okna. Po dokonaniu powyższych operacji na samym końcu sprawdzane jest, czy dokonana konwersja nie była ostatnią i jeżeli tak się dzieje, to procedura zbierania próbek zostaje przerwana.

```
78  ISR( TIMER1_COMPA_vect ) {
79      static uint16_t cnt=0;              // local variable
80      int16_t tmp=ADCW;
81      if(cnt>0){                          // discard first measure
82          samples[cnt-1] = window(tmp,flags.window_type, (cnt-1));
83      }
84      cnt++;                              // inc cnt
85      if(cnt==1025){                      // end of buffer
86          timer1_stop();                  // stop filling
87          cnt=0;                          // reset probe cnt
88          flags.meas_flag=0;              // clear meas flag
89      }
90  }
```

Listing 6.6 Przerwanie od timer1

Sama procedura przerwania również pracuje na granicy dostępnych zasobów sprzętowych. Przerwanie wykonywane jest co około 19,5 [us] przez co w tym czasie upływa 468 taktów zegarowych, gdzie samo przemnożenie przez funkcję okna zajmuje ponad 200 cykli[18].

6.6 Główna pętla programu

Główna pętla programu analizatora przedstawiona została na listingu 6.7.

```
93      // main loop
94      while (1){
95          if(flags.demo_mode) demo_mode();
96          if(flags.redraw_flag) draw_status();
97          fill_sample_buffer();
98          fix_fftr(samples,10);
99          fix_fft_mag(samples,1024);
100         draw_bars();
101         if(flags.rs232_flag){
102             send_mag_via_uart();
103         }
104     }
```

Listing 6.7 Główna pętla programu

Do sterowania programem zostało przyjęte podejście stworzenia maszyny skończonej maszyny stanów opartej na flagach. Znajdują się one w głównej strukturze programu nazwanej *flags*. Pierwszym warunkiem sprawdzanym w głównej pętli programu jest to, czy urządzenie znajduje się w trybie demonstracyjnym. Jeżeli warunek jest spełniony wywoływana jest funkcja demonstracyjna, która generuje sygnały testowe i wyświetla ich widma na wyświetlaczu LCD. W dalszej kolejności sprawdzana jest flaga, określająca czy konfiguracja urządzenia uległa zmianie ze strony użytkownika. Jeżeli tak, to wymagane jest ponowne narysowanie interfejsu na wyświetlaczu LCD z uwzględnieniem dokonanych zmian. Następnie dokonywany jest właściwy program, czyli zebranie próbek, wyznaczenie transformaty oraz wartości skutecznych sygnału wejściowego i narysowanie słupków odzwierciedlających widmo na wyświetlaczu. Ostatnim sprawdzanym warunkiem jest to, czy flaga transmisji danych do komputera jest ustawiona i dane są wysyłane, bądź nie w zależności od jej stanu.

6.7 Podsumowanie

Przedstawiony w rozdziale zestaw funkcji pozwala na poprawną pracę urządzenia. Umożliwia wyświetlanie grafik na wyświetlaczu LCD, przesyłanie danych do komputera PC, realizuje funkcję wyzwalania pomiaru, umieszczania próbek w pamięci RAM oraz wyznaczania widma wejściowego sygnału. Dodatkowo możliwa jest zmiana parametrów pracy programu za pomocą klawiszy dołączonych do mikrokontrolera.

7. Działanie programu

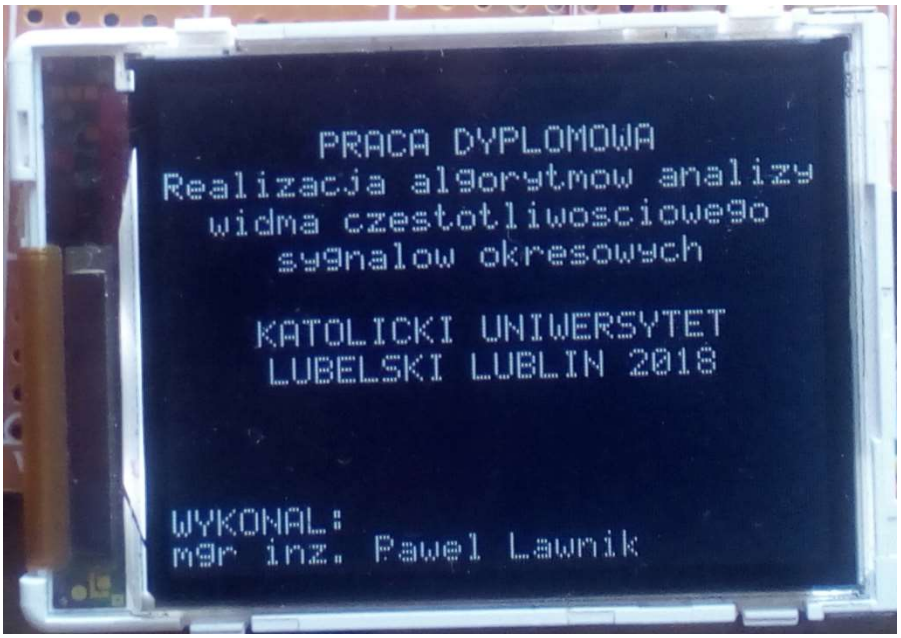
W niniejszym omówiono sposób obsługi zaprojektowanego urządzenia oraz przedstawiono efekty jego pracy dla przykładowych wygenerowanych sygnałów wejściowych.

7.1 Obsługa analizatora

Sterowanie pracą urządzenia odbywa się poprzez klawisze S1-S4 dołączonych według rysunku 5.1. Klawisz pierwszy służy do uruchamiania trybu demonstracyjnego. Kolejny do zmiany typu funkcji okna. Dostępne opcje to kolejno: okno prostokątne, okna Hanninga, okno Hamminga, okno trójkątne oraz okno typu flat-top. Trzeci klawisz służy do uruchamiania i wyłączania transmisji danych do komputera PC. Ostatniemu nie została przyporządkowana żadna funkcja i jest opcją do dalszej rozbudowy urządzenia.

7.2 Praca urządzenia

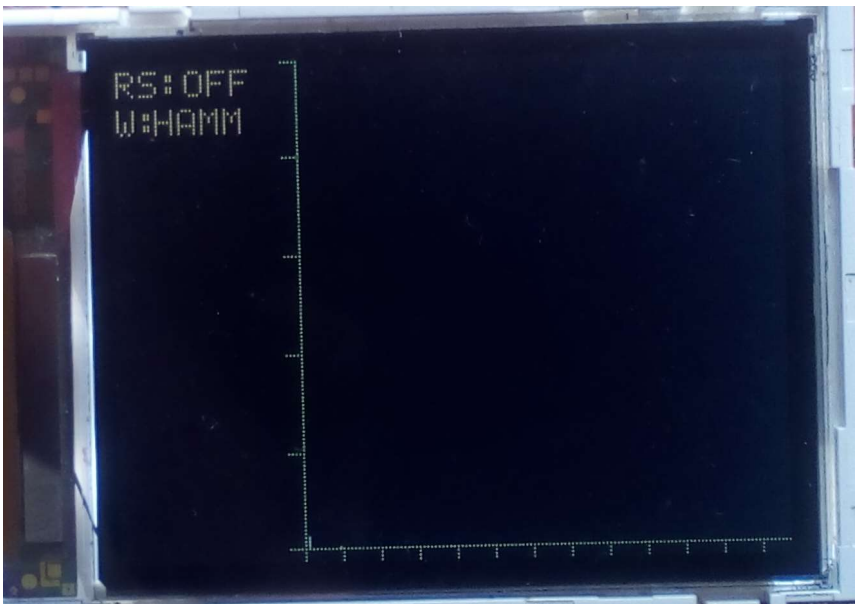
Po uruchomieniu analizatora wyświetlany jest ekran powitalny zawierający informacje o urządzeniu oraz autorze, który przedstawiony został na fotografii 7.1.



Fotografia 7.1. Ekran powitalny

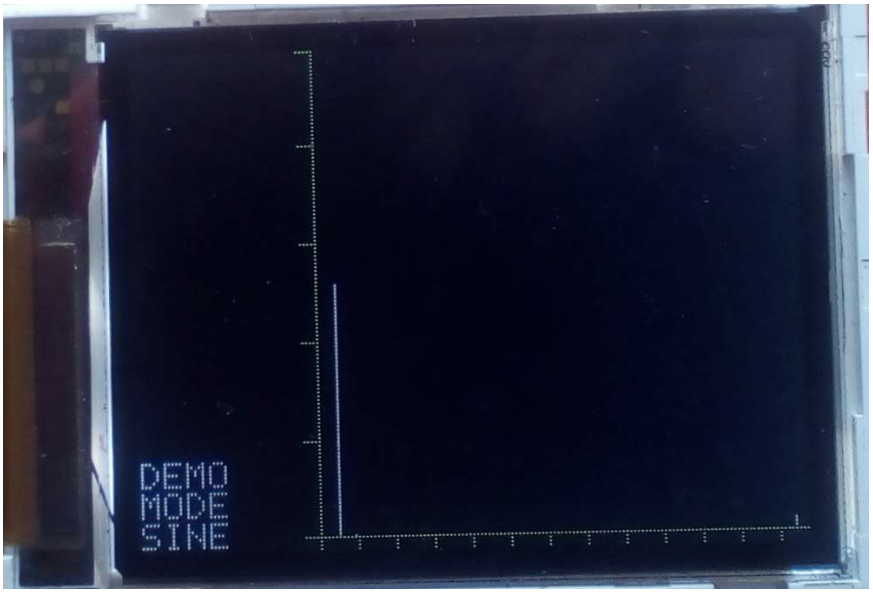
Urządzenie wyświetla ekran powitalny przez 5 sekund po czym wyświetlany jest ekran główny z fotografii 7.2. W lewym górnym rogu umieszczone są parametry pracy urządzenia. W pierwszym rzędzie znajduje się informacja o tym, czy aktywny jest tryb transmisji danych do PC, natomiast w drugim jaki rodzaj okna czasowego został wybrany. Zielonym kolorem narysowane zostały osie w których kreślone jest widmo sygnału. Na osi pionowej zastosowana

jest podziałka, gdzie każdy odcinek odzwierciedla napięcie 0,5V dla harmonicznych. W ten sposób mogą one wynosić o 0 do 2,5V. W osi częstotliwości pierwsza harmoniczna wynosi 200 [Hz]. Podziałka została zrobiona co 2 [kHz] dzięki czemu ostatni wyświetlany prążek ma częstotliwość 25400 [Hz]. Ze względu na strukturą toru pomiarowego składowa stała nie jest wyświetlana.

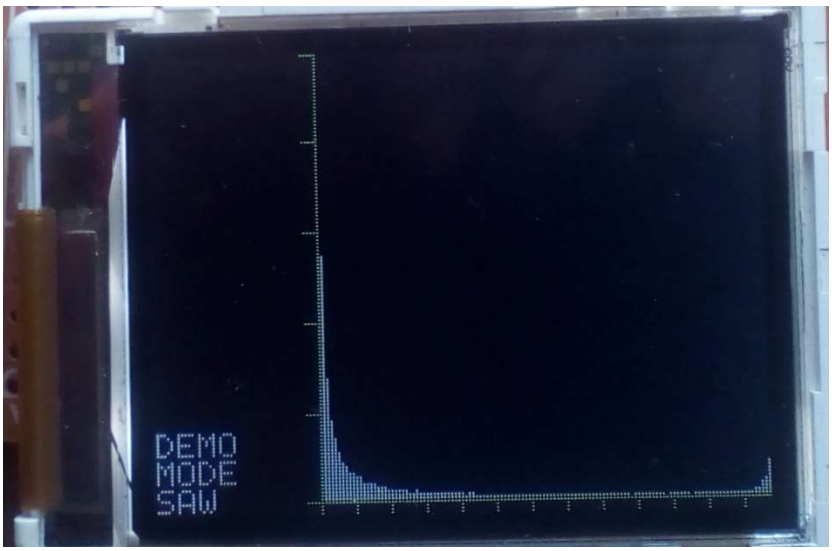


Fotografia 7.2. Ekran główny analizatora

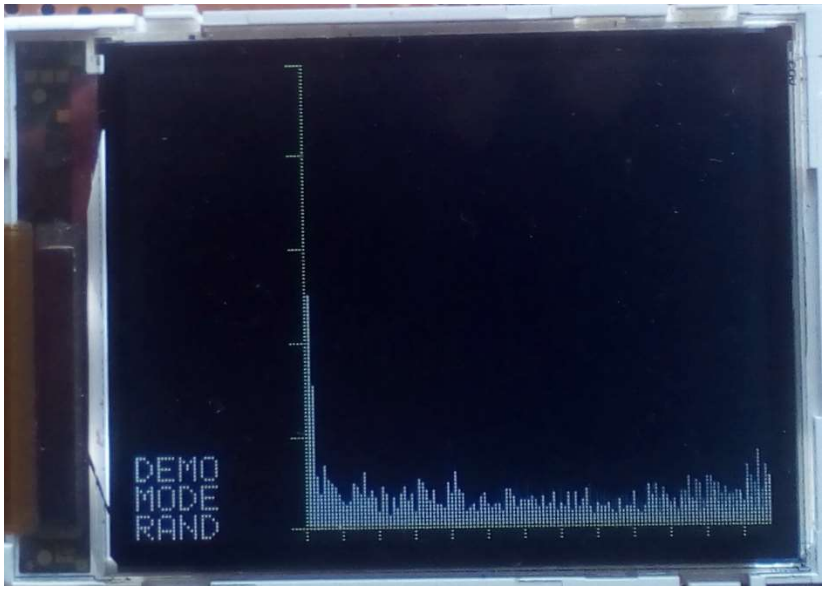
W trybie demonstracyjnym wykreślane są widma wygenerowanych programowo sygnałów przedstawionych na fotografiach od 7.3 do 7.5.



Fotografia 7.3. Sygnał sinusoidalny o częstotliwości 1kHz



Fotografia 7.4. Sygnał piłokształtny o częstotliwości 200kHz



Fotografia 7.5. Sygnał losowy w postaci szumu białego

7.3 Podsumowanie

Wykonany analizator zdaje się poprawnie wyświetlać widma wygenerowanych sygnałów wejściowych. Ze względu na niską rozdzielczość wyświetlacza ilość informacji na nim wyświetlanym została ograniczona do minimum. Brakuje miejsca na wyświetlenia skali oraz jednostek dla obu osi. Pomimo tego, że wyznaczone widmo zawiera 256 prążków sygnału wejściowego możliwe jest narysowanie jedynie 128 jego wartości.

8. Oprogramowanie na PC

W niniejszym rozdziale opisany został program komunikujący się pomiędzy urządzeniem a komputerem PC. Omówiono sposób jego użycia oraz przedstawiono efekt działania.

8.1 Kod programu

Do wizualizacji wyników analizy w czasie rzeczywistym przygotowany został program w postaci skryptu w języku Python. Poniższy listing przedstawia jego strukturę.

```
14  # searching for all avilable com ports
15  def search_serial_ports():
26
27  # receive data
28  def receive_until(port,txt):
44
45  # screen animation
46  def animate(i):
74
75  # main program
76  ...
```

Listing 8.1 Struktura programu odbierającego dane po stronie PC

Pierwsza funkcja korzysta z biblioteki *pyserial* i wyszukuje wszystkie dostępne w komputerze porty COM. Kolejna służy do odbierania danych z portów aż do napotkania podanego w parametrze ciągu znaków. Pozwala to na wykrycie rozpoczęcia i zakończenie ramki z danymi wysyłanymi przez analizator. Z funkcji *animate()* korzysta biblioteka *matplotlib*, która służy do rysowania wykresów w czasie rzeczywistym. W głównym programie początkowo wywoływana jest funkcja wyszukująca wszystkie porty w systemie, a następnie ich lista wyświetlana jest użytkownikowi. Po wybraniu odpowiedniego portu przez użytkownika następuje połączenie po czym otwierane jest okno z widmem sygnału powstałym z odebranych danych. Zamknięcie tego okna powoduje zakończenie pracy programu.

8.2 Obsługa programu

Do poprawnego działania programu potrzebny jest Python w wersji 3.6 z zainstalowanymi bibliotekami *pyserial* oraz *matplotlib*. Program działa poprawnie zarówno z systemami Windows jak i Linux. Różnica w działaniu będzie jedynie w nazewnictwie portów, gdzie dla windowsa porty oznaczone będą jako COM, a dla linuxa tty Uruchomienie odbywa się z konsoli/terminala poprzez wywołanie nazwy skryptu poprzedzonego frazą *python*. Sposób

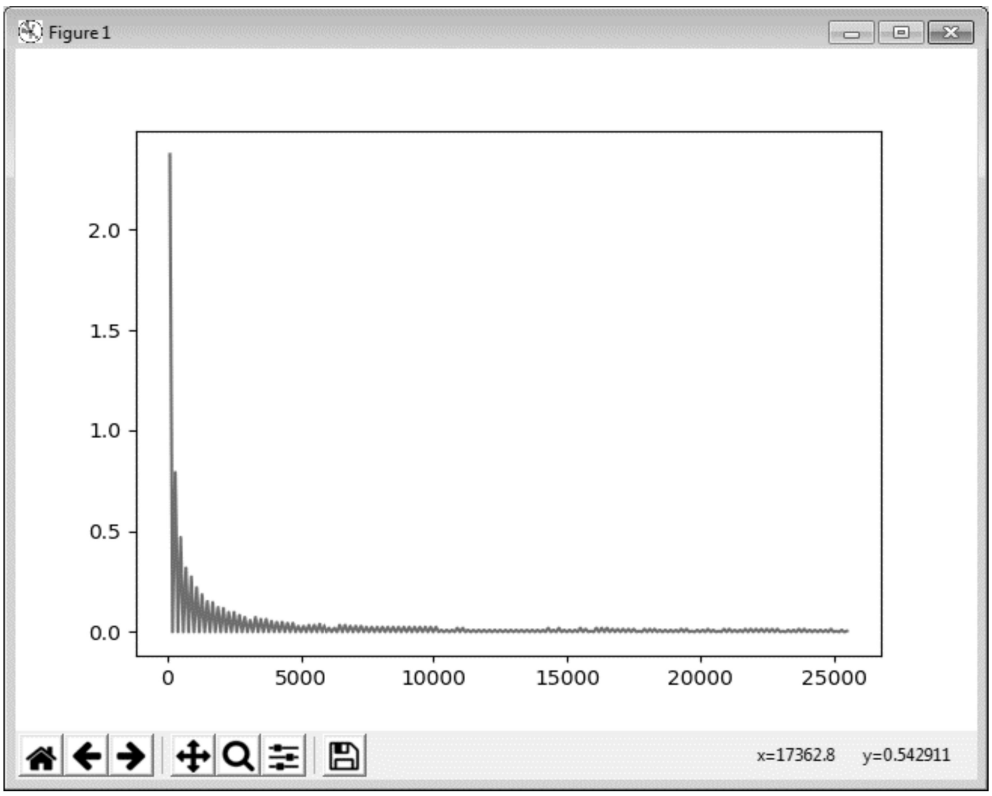
uruchomienia pod systemem windows przedstawia poniższy rysunek. Program po starcie wypisuje nazwy dostępnych w systemie portów COM i oczekuje od użytkownika wpisania nazwy tego, do którego połączony jest analizator. Po wybraniu portu program łączy się z urządzeniem i zaczyna nasłuch danych otwierając jedocześnie okno z wykresem.

```
Pawel@PC C:\Users\Pawel\Desktop
$ python DDA.py
Found 3 com ports!
List:
COM1
COM2
COM6
Choose one of them(COM1,COM2,etc)
COM6
Listening on COM6
```

Rysunek 8.1 Działanie programu PC

8.3 Działanie programu

Po uruchomieniu programu pojawia się poniższe okno przedstawiające dane odebrane z analizatora.



Rysunek 8.1 Działanie programu PC, widmo sygnału prostokątnego

Jednostką osi rzędnych są [Hz], a jej zakres ustawiony jest na sztywno i zawiera się w przedziale od 100 do 25,6 [KHz]. Zakres osi odciętych jest automatycznie skalowany i zależy od amplitudy największej harmonicznej. Jednostka tej osi to [V]. W dolnej części okna znajdują się przyciski nawigacyjne. Pozwalają na powiększenie dowolnej części wykresu, zmianę zakresu osi, rozmiaru wykresu oraz zapisanie wykresu do pliku. Po najechaniu kursorem w dowolny obszar pola wykresu w prawym dolnym rogu wyświetlane są wartości liczbowe położenia kursora co pozwala na dokładny ich odczyt.

8.4 Podsumowanie

Przygotowany program pozwala na dokładniejszy odczyt wartości z widma sygnału niż bezpośrednio z wyświetlacza LCD urządzenia. Dodatkowo umieszczona jest na nim dwa razy większa ilość prążków niż na wyświetlaczu, a ich liczba wynosi 256.