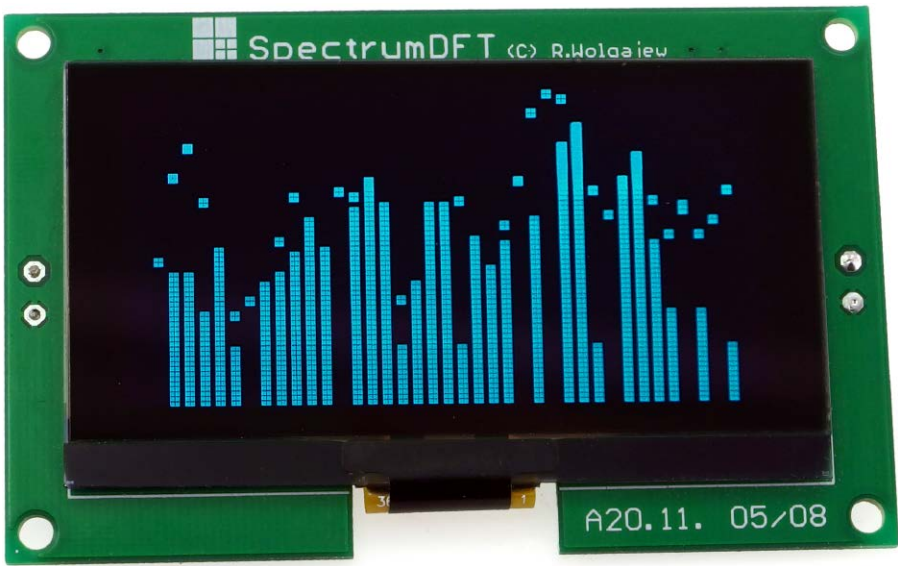
SpectrumDFT

Analizator widma sygnału akustycznego

Pomysł na analizator widma sygnału akustycznego z programową realizacją analizy częstotliwościowej „chodził za mną” od dłuższego czasu. Jak wiadomo potrzebny jest odpowiednio wydajny sprzęt (mikrokontroler) oraz wiedza z zakresu techniki DSP. Cóż tu ukrywać, na samym początku brakowało jednego, jak i drugiego. Jednak natknąłem się na bardzo ciekawy artykuł autorstwa Łukasza Podkalickiego (EP 12/2019) prezentujący arcyciekawe zagadnienia z zakresu DSP w odniesieniu do transformacji Fourier'a w ujęciu realizacji na prostych, 8-bitowych mikrokontrolerach o ograniczonej mocy obliczeniowej i niewielkiej ilości pamięci RAM. Autor pokazuje jak w prosty sposób jesteśmy w stanie wykonać dyskretną transformatę Fourier'a (DFT) sygnału audio.



Dodatkowe materiały do pobrania ze strony www.media.avt.pl

W ofercie AVT* AVT-5748

Podstawowe parametry:
- 40 pasm częstotliwościowych,
- wyświetlacz OLED o rozdzielczości 128×64,
- 4 tryby wyświetlania,
- sygnał wejściowy o maksymalnej amplitudzie 2 Vp-p,
- zasilanie napięciem z zakresu 4,5…6 V.

Projekty pokrewne na www.media.avt.pl:

AVT-5712	Spectrum – prosty analizator widma sygnału akustycznego (EP 9/2019)
AVT-5678	Stereofoniczny wskaźnik wysterowania (EP 6/2019)
AVT-5585	Sterownik wskaźnika wychyłowego do wzmacniacza (EP 1/2018)
AVT-1716	Wskaźnik wysterowania z pamięcią (EP 12/2012)
AVT-1517	Wskaźnik nie tylko wysterowania (EP 9/2012)
AVT-5219	Wizualizator do Winampa na USB (EP 1/2010)
AVT-5210	Analizator widma sygnału audio (EP 11/2009)

* Uwaga! Elektroniczne zestawy do samodzielnego montażu. Wymagana umiejętność lutowania!
Podstawową wersją zestawu jest wersja [B] nazywana potocznie KIT-em (z ang. zestaw). Zestaw w wersji [B] zawiera elementy elektroniczne (w tym [UK] – jeśli występuje w projekcie), które należy samodzielnie wlutować w dołączoną płytkę drukowaną (PCB). Wykaz elementów znajduje się w dokumentacji, która jest podlinkowana w opisie kitu.
Mając na uwadze różne potrzeby naszych klientów, oferujemy dodatkowe wersje:
- wersja [C] – zmontowany, uruchomiony i przetestowany zestaw [B] (elementy wlutowane w płytkę PCB)
- wersja [A] – płytka drukowana bez elementów i dokumentacji

Kity w których występuje układ scalony wymagający zaprogramowania, mają następujące dodatkowe wersje:
- wersja [A+] – płytka drukowana [A] + zaprogramowany układ [UK] i dokumentacja
- wersja [UK] – zaprogramowany układ

Nie każdy zestaw AVT występuje we wszystkich wersjach! Każda wersja ma załączony ten sam plik pdf! Podczas składania zamówienia upewnij się, którą wersję zamawiasz! http://sklep.avt.pl. W przypadku braku dostępności na http://sklep.avt.pl, osoby zainteresowane zakupem płytek drukowanych (PCB) prosimy o kontakt via e-mail: kity@avt.pl.

Dyskretną transformatę Fourier'a (DFT) sygnału audio można wykonać przy użyciu arytmetyki stałoprzecinkowej i współczynników wektora rotującego (tzw. *twiddle factors*). Jakby tego było mało, autor prezentuje również praktyczną realizację 6-punktowego „kolorofonu” bazującego na niewielkim mikrokontrolerze firmy Atmel typu ATtiny13 i programowej realizacji DFT. Posiłkując się tym unikalnym materiałem postanowiłem wykonać prosty i efektowny analizator widma sygnału akustycznego z wykorzystaniem wspomnianej wcześniej techniki DSP.

Mój projekt wymagał jednak rzeczywistej analizy kilkudziesięciu punktów DFT co pociągało za sobą spore wymagania dotyczące mocy obliczeniowej mikrokontrolera wykraczające, zdawałoby się, poza możliwości typowego AVR-a, którymi zwykłem się zajmować. Mając już jednak spore doświadczenie w realizacji projektów z użyciem wspomnianej wcześniej rodziny mikrokontrolerów w naturalny sposób skierowałem się w stronę nowej rodziny AVR a mianowicie Xmega.

Mikrokontrolery rodziny Xmega, mimo że 8-bitowe, wyposażone zostały w wiele mechanizmów i peryferiów, które wydatnie zwiększają ich moce obliczeniowe. Wystarczy wymienić zegar pracujący z częstotliwością 32 MHz, system zdarzeń, wielopoziomowy kontroler przerwań czy układ DMA. Te cechy funkcjonalne, jak i cena niejednokrotnie niższa, niż w przypadku starszych „członków” rodziny AVR, spowodowały, iż zdecydowałem się na budowę prezentowanego urządzenia.

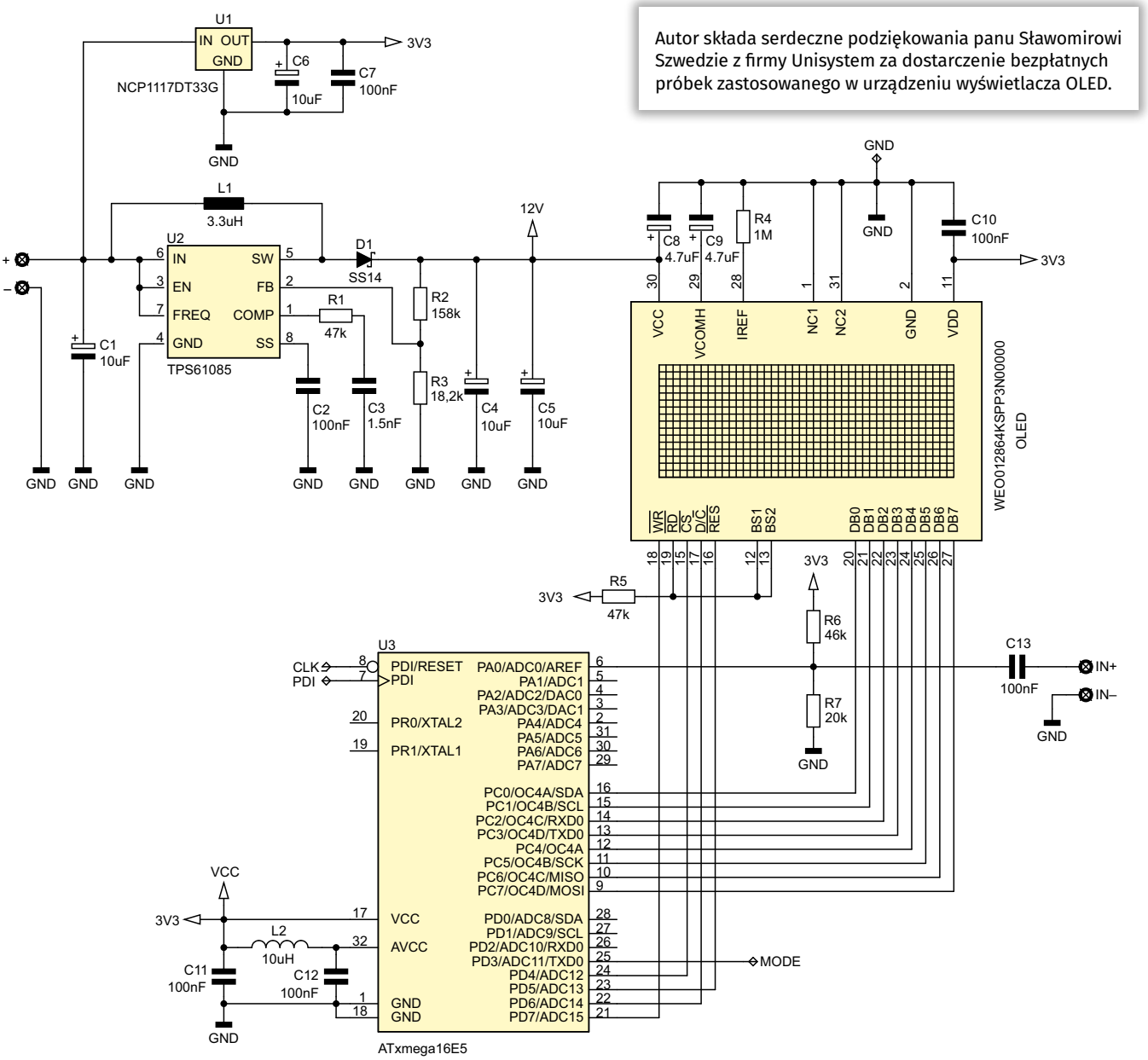
Budowa i działanie

Schemat urządzenia SpectrumDFT pokazano na **rysunku 1**. Zaprojektowano bardzo prosty system mikroprocesorowy, którego „sercem” jest niewielki mikrokontroler firmy Microchip (dawniej Atmel) typu ATXmega16E5 odpowiedzialny za realizację całej założonej funkcjonalności. Taktowany jest wewnętrznym wysokostabilnym generatorem RC o częstotliwości 32 MHz i realizuje programową implementację 80-punktowej transformaty Fourier'a DFT, a wyniki obliczeń wyświetla w postaci wykresu widma na efektownym, graficznym wyświetlaczu OLED o rozdzielczości 128×64 piksele.

Sygnał audio podawany jest na wejście ADC0 wbudowanego w mikrokontroler 12-bitowego przetwornika ADC. Przetwornik próbkuje sygnał z częstotliwością 32 kHz, a wyniki trafiaja do odpowiedniej tablicy, która na drodze programowej poddawana jest obróbce DSP. Wynik działania tego procesu jest pokazywany na wyświetlaczu OLED.

Wyświetlacz do działania potrzebuje napięcia 12 V, w tym celu zastosowano nowoczesną przetwornicę typu step-up o oznaczeniu TPS61085 produkcji firmy Texas Instruments w podstawowej aplikacji. Przetwornica odznacza się doskonałymi parametrami elektrycznymi, z których najważniejsze to wysoka sprawność dochodząca do 93%, szeroki zakres napięcia wejściowego 2,3...6 V, opcja miękkiego startu oraz szereg wbudowanych zabezpieczeń (termiczne, od niskiego napięcia wejściowego). Duża częstotliwość przełączania 1,2 MHz pozwala stosować elementy o mniejszych

Autor składa serdeczne podziękowania panu Sławomirowi Szwedzie z firmy Unisystem za dostarczenie bezpłatnych próbek zastosowanego w urządzeniu wyświetlacza OLED.



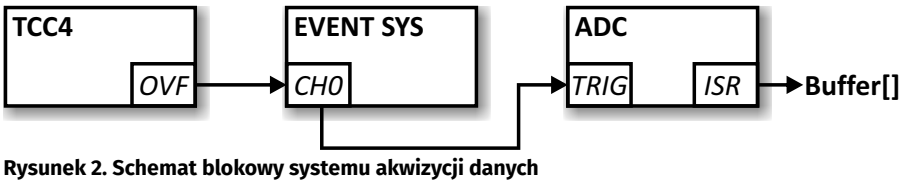
Rysunek 1. Schemat ideowy urządzenia

wymiarach (mniejsza indukcyjność i pojemność). Dodatkowo, układ TPS61085 wyposażono w wejście EN (Enable), przy pomocy którego możemy wyłączyć przetwornicę, jeśli zajdzie taka potrzeba (np. wygaszenie wyświetlacza) i tym samym ograniczyć pobór prądu ze źródła napięcia zasilającego.

Nie bez powodu wybrano ten rodzaj wyświetlacza. Jego właściwości użytkowe w porównaniu z typowym elementem LCD są nie do przecenienia, a cena jest zbliżona do ceny analogicznego elementu wykonanego w technologii LCD.

Główne zadanie układu, jakim jest akwizycja i przetwarzanie danych próbek sygnału audio, jest możliwe dzięki sprzęgnięciu ze sobą kilku podsystemów mikrokontrolera AVR, co pokazano na **rysunku 2**. Podstawowym podsystemem mikrokontrolera ATxmega16E5, inicjującym pomiar przetwornika ADC, jest układ czasowo-licznikowy TCC4. Jest on taktowany wysokostabilnym przebiegiem zegarowym o częstotliwości 32 MHz (tym samym, co rdzeń mikrokontrolera) i generuje zdarzenie przepełnienia licznika (OVF) co 1000 taktów zegara (PER=999), czyli 32000 razy na sekundę.

Zdarzenie przepełnienia licznika połączone jest z kanałem CH0 systemu zdarzeń (EVENT SYSTEM), dla którego stanowi źródło zdarzeń. Kanał CH0 systemu zdarzeń jest z kolei wyzwalaczem (TRIG) dla wbudowanego w strukturę mikrokontrolera przetwornika ADC, dzięki czemu możliwe jest próbkowanie wejściowego sygnału audio w równych odstępach czasu (32 kHz).



Rysunek 2. Schemat blokowy systemu akwizycji danych

Listing 1. Kod odpowiedzialny za inicjalizację układu czasowo-licznikowego TCC4

```
#define SAMPLING_FREQ 32000

//Uruchomienie timera TCC4 by
//przepełniał się 32000 razy/sek
TCC4.CTRLB = TC_WGMODE_NORMAL_gc;
TCC4.PER = F_CPU/SAMPLING_FREQ-1;
```

Listing 2. Kod odpowiedzialny za powiązanie zdarzenia przepełnienia licznika TCC4 z kanałem CH0 systemu zdarzeń

```
//Kanał 0 systemu zdarzeń będzie
//przekazywał zdarzenie przepełnienia TCC4
EVSYS.CH0MUX = EVSYS_CHMUX_TCC4_OVF_gc;
```

Przerwanie od zakończenia konwersji przetwornika ADC (ADCA_CH0_vect) odpowiedzialne jest za właściwą akwizycję danych oraz ustawianie stosownych flag dla programu obsługi aplikacji.

Warto wspomnieć o innej możliwości automatycznej akwizycji danych, którą to początkowo wykorzystywałem w programie obsługi aplikacji. Mowa o podsystemie DMA, dzięki któremu możliwa staje się akwizycja danych i ich automatyczne umieszczanie w buforze programowym bez udziału rdzenia mikrokontrolera, dzięki specjalnym wyzwalaczom podsystemu DMA, którym może być np. zdarzenie zakończenia konwersji przetwornika ADC. Początkowo wykorzystywałem tą zaawansowaną możliwość mikrokontrolera rodziny Xmega. Jednak program obsługi aplikacji nie wykonuje absolutnie żadnych zadań poza zbieraniem i obróbką danych, więc w końcowym programie zrezygnowałem z usług podsystemu DMA, gdyż nie wnosił on żadnych usprawnień.

Program sterujący

Kod odpowiedzialny za inicjalizację układu czasowo-licznikowego TCC4 pokazano na **listingu 1**, zaś kod odpowiedzialny za powiązanie zdarzenia przepełnienia licznika TCC4 z kanałem CH0 systemu zdarzeń pokazano na **listingu 2**. Dalej, na **listingu 3** pokazano kod odpowiedzialny za inicjalizację przetwornika ADC mikrokontrolera. Wybrano 8-bitowy tryb SINGLE ENDED przetwornika co podyktowane było potrzebą uproszczenia mechanizmów odpowiedzialnych za późniejsze obliczenie transformaty DFT badanego sygnału (ograniczenie zakresu zmiennych). Jako źródło referencji wybrano napięcie VCC/1,6, zaś jako wyzwalacz konwersji kanał CH0 systemu zdarzeń. Ponadto uruchomiono przerwanie po konwersji przetwornika ADC odpowiedzialne za akwizycję danych, którego ciało pokazano na **listingu 4**.

Zebraniu kompletnej porcji danych towarzyszy zatrzymanie akwizycji danych (wyłączenie timera TCC4) oraz ustawienie flagi *ADCdataReady*, dzięki czemu możliwe jest przetworzenie danych przez program

Listing 3. Kod odpowiedzialny za inicjalizację przetwornika ADC mikrokontrolera

```
//ADC config: 8bit single ended, no curr limit,
//ref = VCC/1.6, gain = x1, trigger - event ch0
//ADC Enabled
ADCA.CTRLA = ADC_ENABLE_bm;
//300ksps max sampling rate
ADCA.CTRLB = ADC_CURRLIMIT_NO_gc | ADC_RESOLUTION_8BIT_gc;
//Internal VCC / 1.6
ADCA.REFCTRL = ADC_REFSEL_INTVCC_gc;
//First event triggers channel conversion
ADCA.EVCTRL = ADC_EVSEL_0_gc | ADC_EVACT_CH0_gc;
//1MHz @ 32MHz
ADCA.PRESCALER = ADC_PRESCALER_DIV32_gc;
ADCA.CH0.CTRL = ADC_CH_GAIN_1X_gc | ADC_CH_INPUTMODE_SINGLEENDED_gc;
//pin PA0
ADCA.CH0.MUXCTRL = ADC_CH_MUXPOS_PIN0_gc;
ADCA.CH0.INTCTRL =
        //Interrupt on conversion complete
        ADC_CH_INTMODE_COMPLETE_gc |
        //High level
        ADC_CH_INTLVL_HI_gc;
```

Listing 4. Kod funkcji obsługi przerwania przetwornika ADC odpowiedzialnej za akwizycję danych

```
//Liczba próbek sygnału
#define N 80
#define DATA_ACQUISITION_ON TCC4_CTRLA = TC_CLKSEL_DIV1_gc
#define DATA_ACQUISITION_OFF TCC4_CTRLA = TC_CLKSEL_OFF_gc

ISR(ADCA_CH0_vect){
    static uint8_t Idx;

    Buffer[Idx++] = ADCA.CH0RESL;

    if(Idx == N){
        Idx = 0;
        ADCdataReady = 1;
        //Wyłączenie timera TCC4
        DATA_ACQUISITION_OFF;
    }
}
```

Listing 5. Kod funkcji odpowiedzialnej za obliczenie dyskretnej transformaty Fourier'a

```
//Multiplication factor, max 128
#define MULF 64

void DFT(void){
    uint16_t a, b;
    int32_t Re, Im;
    //Obliczamy moc sygnału dla poszczególnych
    //prążków częstotliwościowych
    for (uint8_t i=0; i<(N/2)+1; ++i){
        Re = Im = a = 0;
        b = 3*N/4;

        for (uint8_t j=0; j<N; ++j){
            Re += (Buffer[j] * Twiddle[a % N])/MULF;
            Im -= (Buffer[j] * Twiddle[b % N])/MULF;
            a += i;
            b += i;
        }
        Power[i] = (Re*Re + Im*Im)/(N*N);
    }
}
```

Listing 6. Kod funkcji odpowiedzialnej za wyznaczenie współczynników wektora rotującego

```
#define PI2 6.2832  //2*Pi

void calculateTwiddleFactors(void){
    for(uint8_t i=0; i<N; ++i) Twiddle[i] = (int16_t) (MULF*cos(i*PI2/N));
}
```

Listing 7. Kod funkcji odpowiedzialnej za wyznaczenie współczynników okna Hanna (Hanninga)

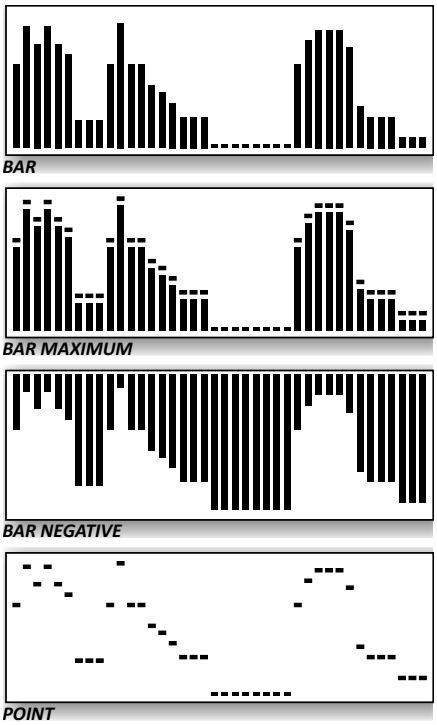
```
void calculateWindowFactors(void){
    for(uint8_t i=0; i<N; ++i) Window[i] = (int16_t) (MULF*(0.5-0.5*cos(i*PI2/(N-1))));
}
```

Listing 8. Kod odpowiedzialny za konfigurację zegara mikrokontrolera Xmega16E5

```
//Uruchamiamy wewnętrzny oscylator 32MHz
// i czekamy na ustabilizowanie się jego przebiegu
OSC.CTRL |= OSC_RC32MEN_bm;
while(!(OSC.STATUS & OSC_RC32MRDY_bm));
//Włączenie oscylatora 32M, jako źródła taktowania mikrokontrolera
CPU_CCP = CCP_IOREG_gc;
CLK.CTRL = CLK_SCLKSEL_RC32M_gc;
//Wyłączamy wewnętrzny oscylator 2MHz, gdyż nie jest już potrzebny
OSC.CTRL &= ~ OSC_RC2MEN_bm;
```

główny aplikacji. Za przetworzenie danych, czyli obliczenie dyskretnej transformaty Fourier'a z próbek sygnału zebranych w tablicy *Buffer[]*, odpowiedzialna jest funkcja pokazana na **listingu 5**. Wynikiem działania funkcji *DFT()* jest obliczenie mocy poszczególnych prążków częstotliwości (co 400 Hz) i zapisanie ich w tablicy *Power[]*.

Właśnie to zadanie stanowi główny problem obliczeniowy, o którym wspomniano na wstępie artykułu. Wynika to z liczby obliczeń stałoprzecinkowych wykonywanych w ramach dwóch pętli, z których składa się wspomniana funkcja. Jak widać, liczba tych obliczeń zależy bezpośrednio od liczby punktów transformaty Fourier'a (N), która w naszym przypadku wynosi 80. Z kolei liczba punktów transformaty determinuje



Rysunek 3. Wizualizacje przedstawiające 4 tryby wyświetlania informacji o widmie sygnału audio

Wykaz elementów:
Rezystory: (obudowy SMD 0805)
R1, R5: 47 kΩ
R2: 158 kΩ 1%
R3: 18,2 kΩ 1%
R4: 1 MΩ
R6: 46 kΩ 1%
R7: 20 kΩ 1%

Kondensatory:
C1, C4…C6: 10 µF/16 V tantalowy (SMD A)
C2, C7, C10…C12: 100 nF (SMD 0805)
C3: 1,5 nF (SMD 0805)
C8, C9: 4,7 µF/16 V tantalowy (SMD A)

Półprzewodniki:
U1: NCP1117DT33G (TO252)
U2: TPS61085 (TSSOP8)
U3: ATxmega16E5 (TQFP32)
D1: SS14 (SMA)
OLED: OLED Winstar WEO012864KSPP3N00000 (128×64)

Inne:
L1: dławik mocy 3,3 µH typu DLG-0504-3R3
L2: dławik SMD 10 µH (SMD 0805)
ZIF: złącze typu ZIF (raster 0,5 mm, 31-pin, górny kontakt)

odległość kolejnych prążków mocy (tzw. BIN) a wynika z zależności BIN = częstotliwość próbkowania sygnału/liczba punktów transformaty (N). Dla naszego przypadku BIN=400 Hz (32 kHz/80), co oznacza że kolejne wartości częstotliwości, dla których liczona jest moc sygnału są wielokrotnością 400 Hz. Wartość ta jest kompromisem pomiędzy rozdzielczością mocy (BIN) a czasem niezbędnym na wykonanie funkcji DFT() przy przyjętej liczbie punktów transformaty (N). W naszym przypadku czas ten wynosi około 50 ms (zmierzony empirycznie), co determinuje częstotliwość odświeżania wykresu widma (tzw. framerate), która w tym przypadku wynosi 20 Hz.

Dalsze zwiększanie liczby punktów transformaty (N), mimo iż pożądane, zmniejszyłoby częstotliwość odświeżania wykresu widma do wartości nieakceptowalnych i praktycznie nieużytecznych. To jest główne ograniczenie software'owe naszej implementacji i wynika w głównej mierze z 8-bitowej architektury mikrokontrolera Xmega, jak i maksymalnej, dostępnej częstotliwości taktowania rdzenia. Trzeba mieć na uwadze, że w ciągu wspomnianych 50 ms mikrokontroler Xmega moze wykonać niemal 1,6 mln instrukcji (jednotaktowych), co daje nam zarys sytuacji w zakresie złożoności obliczeniowej funkcji *DFT()*.

Funkcja *DFT()* korzysta z tablicy *Twiddle[]* współczynników wektora rotującego (tzw. *twiddle factors*), której wyznaczeniem zajmuje się funkcja *calculateTwiddleFactors()* pokazana na **listingu 6**. Oczywiście funkcja z listingu 6 jest niejako nadmiarowa, gdyż współczynniki takie moglibyśmy wyznaczyć sobie w arkuszu kalkulacyjnym i zapisać na stałe w pamięci programu (Flash) co zmniejszyłoby zajętość pamięci RAM mikrokontrolera oraz kod obsługi aplikacji. Ja zdecydowałem się na wyznaczenie ich w trakcie działania programu, gdyż po pierwsze, jest to szybsze, zaś po drugie i niemniej ważne, mikrokontroler wyposażono w dużą ilość pamięci RAM.

Warto podkreślić, że przed wykonaniem funkcji *DFT()* zebrana tablica danych wejściowych *Buffer[]* poddawana jest okienkowaniu, które ma na celu ograniczenie tak zwanych wycieków widma sygnału. Zastosowana funkcja okna jest typu Hanna (Hanninga), zaś wyznaczeniem stosownych współczynników okna (*Window[]*) zajmuje się funkcja *calculateWindowFactors()*, której to ciało pokazano na **listingu 7**. Spróbkowany przebieg wejściowy zebrany w tablicy *Buffer[]*, jest przemnażany przez funkcję okna *Window[]* przed wykonaniem transformaty Fourier'a w ramach funkcji *DFT()*.

Po wykonaniu funkcji *DFT()*, a przed wyświetleniem widma mocy sygnału, stosowne moce przeliczane są do skali logarytmicznej (dB). Wynika to głównie z dużej dynamiki sygnału *Power[]* i konieczności pokazania go na ograniczonej rozdzielczością pionową wyświetlacza OLED (64 piksele) skali sygnału. Przeliczenie, o którym mowa, wykonywane jest według następującej zależności: $Power=10\cdot\log_{10}(Power)$.

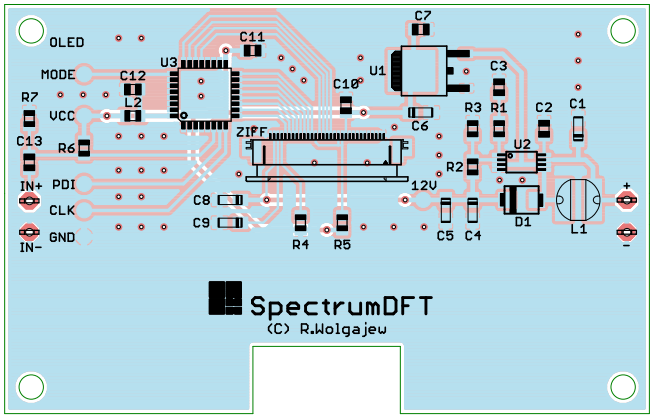
Na koniec przedstawię kod odpowiedzialny za uruchomienie wewnętrznego oscylatora 32 MHz mikrokontrolera Xmega, jako że podstawowo mikrokontroler ten uruchamiany jest z aktywnym oscylatorem o częstotliwości 2 MHz, zaś nasze urządzenie stawia dość wysokie wymagania obliczeniowe. Kod odpowiedzialny za konfigurację zegara mikrokontrolera Xmega16E5 pokazano na **listingu 8**.

Program obsługi aplikacji urządzenia SpectrumDFT przewiduje 4 tryby wyświetlania informacji o widmie sygnału audio, których przykładowe wizualizacje przedstawiono na **rysunku 3**. Warto podkreślić, że tryb trzeci (Bar maximum) integruje dodatkową funkcjonalność w postaci pokazywania wartości szczytowej w każdym z 40 pasm częstotliwościowych.

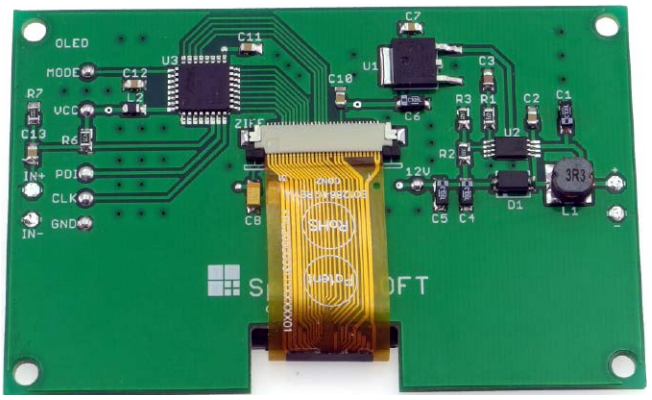
Montaż i uruchomienie

Schemat montażowy urządzenia SpectrumDFT pokazano na **rysunku 4**. Zaprojektowano bardzo zwarty obwód drukowany o wielkości zbliżonej do zastosowanego wyświetlacza OLED zbudowany wyłącznie z elementów SMD umieszczonych tylko po stronie BOTTOM.

Moduł wyświetlacza OLED podłączony jest do płytki naszego urządzenia z użyciem gniazda ZIF o bardzo gęstym rastrze (0,5 mm), dlatego montaż naszego urządzenia rozpoczynamy właśnie od przylutowania tegoż gniazda. Najprostszym sposobem montażu elementów o tak dużym zagęszczeniu wyprowadzeń, niewymagającym jednocześnie posiadania specjalistycznego sprzętu, jest użycie typowej stacji lutowniczej, wraz z dobrej jakości cyną oraz odpowiednią ilością topnika. Niezbędna będzie, także dość cienka plecionka rozlutownicza, która umożliwi usunięcie nadmiaru cyny spomiędzy wyprowadzeń złącza. Należy



Rysunek 4. Sc hemat montażowy



Fotografia 1. Zmontowany obwód drukowany widziany od strony BOTTOM

przy tym uważać by nie uszkodzić termicznie tego elementu. Jakość wykonanego połączenia sprawdzamy pod lupą korzystając z najprostszego miernika ze sprawdzaniem ciągłości połączeń.

Następnie przechodzimy do przylutowania mikrokontrolera i przetwornicy napięcia TPS61085. Potem lutujemy pozostałe półprzewodniki a na samym końcu elementy bierne. Opcjonalny przycisk MODE sterujący trybem wyświetlania widma przylutowujemy pomiędzy wyprowadzenia pola lutowniczego MODE oraz masę zasilania. Na samym końcu podłączamy wyświetlacz OLED do złącza ZIFF po stronie BOTTOM, zaś sam element przyklejamy po stronie TOP (w miejscu wyznaczonym obrysem) korzystając z dwustronnej taśmy klejącej. Zmontowany obwód drukowany urządzenia widziany od strony BOTTOM pokazano na **fotografia 1**.

Źródło zasilania o napięciu z zakresu 4,5...6 V podłączamy przewodami do pól lutowniczych oznaczonych + i –. Z kolei sygnał wejściowy o maksymalnej amplitudzie 2 Vp-p podłączamy do pól lutowniczych oznaczonych IN+ i IN–.

Poprawnie zmontowany układ powinien działać po włączeniu zasilania. Dla pewności warto sprawdzić wartość napięcia przetwornicy step-up (punkt lutowniczy 12 V), które powinno oscylować w okolicach 12 V. Warto również podkreślić, iż domyślnym trybem wyświetlania informacji o widmie jest tryb „BAR MAXIMUM”.

Robert Wołgajew, EP