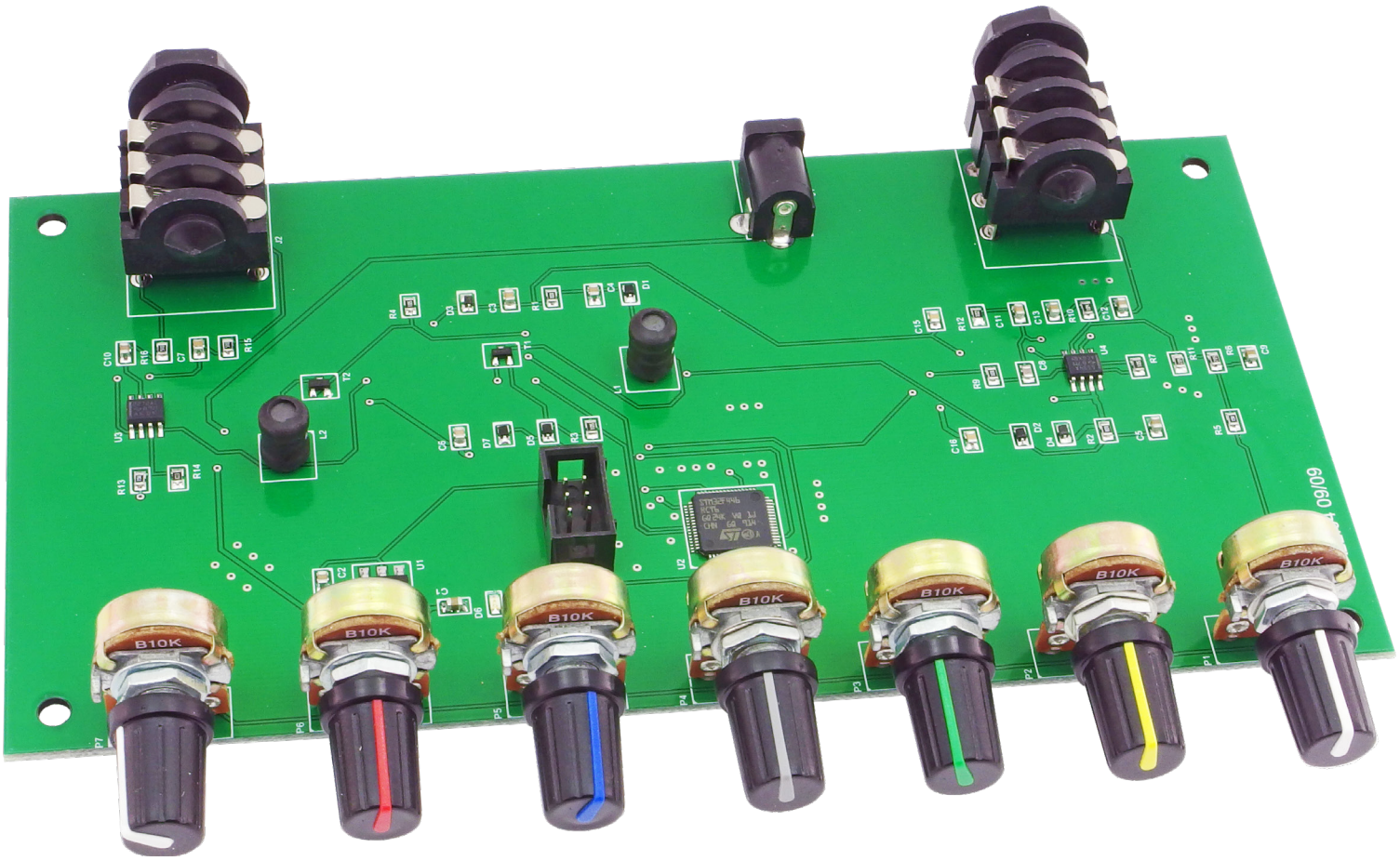


Efekt do gitary

z zastosowaniem cyfrowego przetwarzania sygnałów

Dzięki nowoczesnym mikrokontrolerom staje się możliwe realizowanie skomplikowanych zadań stosunkowo niewielkim kosztem. Najnowsze układy są na tyle wydajne, że w niektórych zadaniach mogą zastępować skomplikowane procesory DSP i umożliwiają budowę układów analogowych z cyfrowym przetwarzaniem sygnału. Prezentowane urządzenie to efekt gitarowy wykorzystujący takie możliwości.

Proponowane przeze mnie rozwiązanie bazuje na mikrokontrolerze STM32F466, który ma rozbudowane układy peryferyjne, a wśród nich trzy 12-bitowe przetworniki A/C z możliwością multipleksowania wejść, oraz dwa 12-bitowe przetworniki C/A. Dzięki takim komponentom możliwe było znaczne obniżenie stopnia skomplikowania układu. Realizowany efekt to fuzz (distortion) graficzny o siedmiu kanałach częstotliwościowych wyposażony dodatkowo w efekt pogłosu i efekt tremolo.

Sposób realizacji efektów

Schemat ideowy sposobu realizacji efektu fuzz pokazuje **rysunek 1**. Tor sygnału jest podzielony na wiele kanałów częstotliwościowych i w każdym z nich sygnał podlega zniekształceniu (*distortion*). Sygnał wyjściowy powstaje z zsumowania wyjść tych kanałów.

Efekt tremolo działa na zasadzie modulowania sygnału audio sygnałem z generatora wolnych przebiegów LFO (*low frequency oscylator*) (**rysunek 2**). W naszym układzie

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

W ofercie AVT* AVT-5798

Podstawowe parametry:
- realizowane efekty: fuzz (distortion) graficzny o siedmiu kanałach częstotliwościowych wyposażony dodatkowo w efekt pogłosu i efekt tremolo,
- prosta konstrukcja dzięki zastosowaniu rozbudowanego mikrokontrolera,
- zasilanie napięciem 5 V.

Projekty pokrewne na www.media.avt.pl:

Projekt 239	Multiefekt dla wokalistów VEP10 (EP 1/2019)
AVT-5641	Procesor wokalny z efektami echa DRP-10 (EP 10/2018)
AVT-5576	Moduł „delay/reverb” (EP 4/2017)
----	Pogłos analogowy (EP 3/2017)
AVT-5569	Mikser Dry/Wet (EP 2/2017)
AVT-5544	Stereofoniczna, cyfrowa linia opóźniająca (EP 7/2016)
----	DSPfactory – profesjonalny efekt dźwiękowy dla muzyków (EP 3-5/2016)
----	Efekt „Reverb” do gitary lub instrumentu klawiszowego (EP 3/2015)
AVT-5484	Delay – efekt do instrumentu muzycznego (EP 1/2015)
AVT-3049	AVRSYN2 – syntezator muzyczny na ośmiobitowym mikrokontrolerze (EdW 1/2013)
AVT-1466	Echo cyfrowe (EP 6/2008)

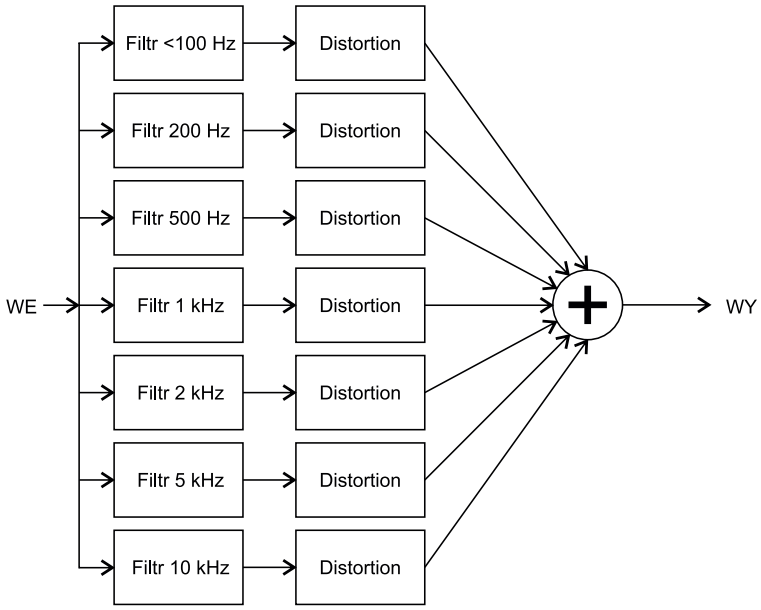
*** Uwaga! Elektroniczne zestawy do samodzielnego montażu. Wymagana umiejętność lutowania!**

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.

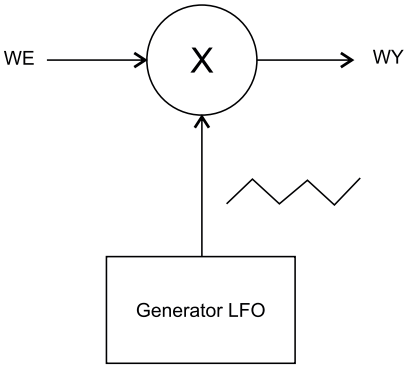
Mając na uwadze różne potrzeby naszych klientów, oferujemy dodatkowe wersje:
- wersja [**C**] – zmontowany, uruchomiony i przetestowany zestaw [**B**] (elementy wlutowane w płytkę PCB)
- wersja [**A**] – płytka drukowana bez elementów i dokumentacji

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.



Rysunek 1. Schemat ideowy sposobu realizacji efektu fuzz



Rysunek 2. Schemat ideowy sposobu realizacji efektu tremolo

generator LFO wytwarza przebieg o regulowanej częstotliwości 1...100 Hz i amplitudzie 0...1. Do przebiegu dodawana jest składowa stała tak, że sygnał z generatora LFO nie przyjmuje wartości ujemnych.

Schemat blokowy

Na **rysunku 3** został pokazany schemat blokowy urządzenia. Pierwszym blokiem w torze audio jest przedwzmacniacz zbudowany na niskoszumowym wzmacniaczu operacyjnym. Przetworniki A/C i C/A to komponenty zintegrowane w mikrokontrolerze, pracujące z rozdzielczością 12 bitów. Bufor wyjściowy zrealizowany został na wzmacniaczu operacyjnym TLC2272. Układy regulacji to potencjometry podłączone do wejść multipleksowanego, drugiego przetwornika A/C.

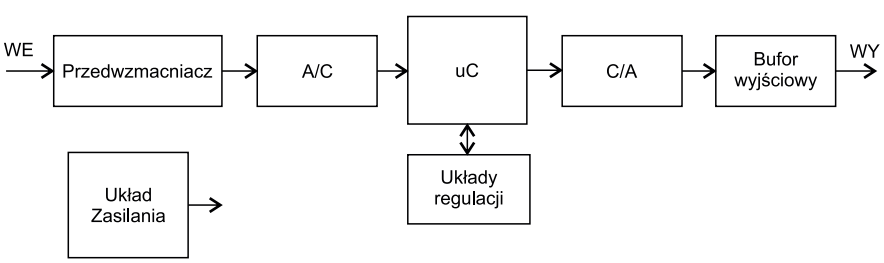
Układ zasilany jest napięciem 5 V. Zadaniem układu zasilania jest wytworzenie napięcia 3,3 V, niezbędnego do zasilania mikrokontrolera, oraz napięć dodatniego i ujemnego do zasilania wzmacniaczy operacyjnych.

Budowa układu

Schemat elektryczny urządzenia został pokazany na **rysunku 4**. Sygnał wejściowy podawany jest na gniazdo J3, typu MONO JACK 6,3 mm. Na wejściu toru audio znajduje się przedwzmacniacz zbudowany na układzie U4 i elementach sąsiadujących z nim. Zastosowano niskoszumowym wzmacniacz operacyjny, przeznaczony do stosowania w torach audio, typu NE5534. Spośród innych podobnych układów wyróżnia się bardzo niskim współczynnikiem szumów, wynoszącym poniżej 4 nV/√Hz, oraz szerokim pasmem sięgającym 10 MHz.

Przedwzmacniacz ma wzmocnienie ok. 1000 i jest tak skonstruowany, aby dopasował sygnał do wejścia analogowego mikrokontrolera. Na jego wyjściu znajduje się układ zabezpieczający przed wystąpieniem napięcia zbyt wysokiego dla wejścia mikrokontrolera.

W bloku wyjściowym pracuje bufor zrealizowany na podwójnym wzmacniaczu operacyjnym U3, typu TLC2271. Wybrano ten wzmacniacz ze względu na niski pobór mocy i stosunkowo duży prąd wyjściowy.



Rysunek 3. Schemat blokowy urządzenia

Wykaz elementów:

Rezystory: (SMD0805)
R1, R2, R9: 100 Ω
R3, R4, R7: 1 MΩ
R5, R14: 10 kΩ
R6, R11: 2 MΩ
R8: 330 Ω
R10: 100 kΩ
R12: 1 kΩ
R13: 150 kΩ
R15, R16: 51 Ω
P1…P7: potencjometr 10 kΩ liniowy

Kondensatory: (SMD0805)
C1, C2, C4, C5, C9, C12, C13: 1 µF
C3, C6, C7, C10, C11: 10 µF
C8: 33 pF
C14…C16: 100 nF

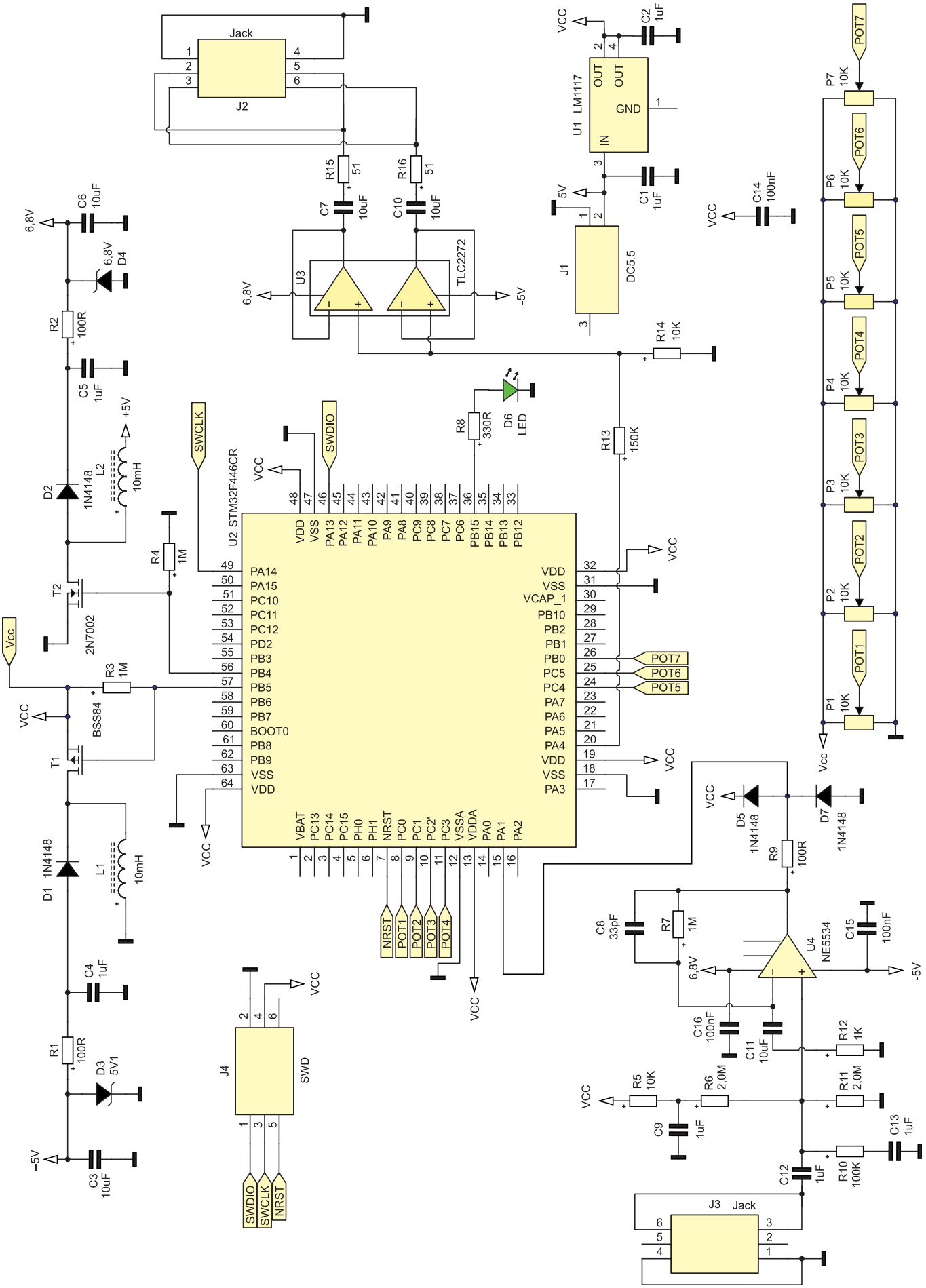
Półprzewodniki:
D1, D2, D5, D7: 1N4148 SMD
D3: dioda Zenera 5,1 V SMD
D4: dioda Zenera 6,8 V SMD
D6: LED SMD0805
T1: BSS84
T2: 2N7002
U1: LM1117 SOT223
U4: NE5534 SO8
U2: STM32F446CR SMD
U3: TLC2272 SO8

Pozostałe:
L1, L2: 10 mH THT
J1: gniazdo zasilania DC5,5
J2, J3: gniazdo jack 6,3 do druku
J4: złącze IDC6

Wzmacniacze pracują jako wtórniki, po jednym na każdy kanał, a sygnały wyprowadzone są na gniazdo J2, typu stereo jack 6,3 mm. Wyjście z przetwornika C/A jest podawane na wejście wzmacniaczy operacyjnych przez dzielnik rezystorowy (R13 i R14) dopasowujący amplitudę sygnału do standardu sygnału wyjściowego.

Do multipleksowanych wejść drugiego przetwornika A/C mikrokontrolera podłączone są potencjometry służące do regulacji parametrów efektów.

Ostatnim elementem układu jest blok zasilania, który składa się ze scalonego stabilizatora liniowego U1, typu LM1117, dającego napięcie 3,3 V oraz dwóch przetwornic napięcia. Obie pracują w klasycznym układzie z indukcyjnością, jedna wytwarza napięcie ujemne –5 V (T1, D1, L1), a druga napięcie dodatnie +6,8 V (T2, D2, L2). Elementy kluczujące (tranzystory MOSFET) są sterowane z wyjść mikrokontrolera, które kontroluje



Rysunek 4. Schemat elektryczny urządzenia

Listing 1. Kod głównych procedur programu

```
#include "main.h"
#include "efekt10.h"

int main(){
    InitPeryferia();

    HAL_DAC_Start(&hdac, DAC_CHANNEL_1);

    __HAL_ADC_ENABLE_IT(&hadc1, ADC_IT_EOC);
    __HAL_ADC_ENABLE_IT(&hadc2, ADC_IT_EOC);

    HAL_TIM_Base_Start(&htim3);
    HAL_TIM_Base_Start_IT(&htim2);
    HAL_TIM_Base_Start(&htim1);
    HAL_TIM_OC_Start(&htim1, TIM_CHANNEL_1);
    HAL_TIM_OC_Start(&htim3, TIM_CHANNEL_1);
    HAL_TIM_OC_Start(&htim3, TIM_CHANNEL_2);


    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_SET);

    for(;;);                // petla nieskonczona

    return 0;
}

/****** Barwa tonu*****************************************/

#define K_D (int)((double)0x10000*2*3.14*200/CZ_PROBKOWANIA)
#define K_G (int)((double)0x10000*2*3.14*5000/CZ_PROBKOWANIA)

int Filtr(int w){
    static int C_D = 0, C_G = 0;

    C_D += (w-C_D)/0x10000*K_D;
    C_G += (w-C_G)/0x10000*K_G;

    volatile register int wy_d, wy_g, wy_s;

    wy_d = C_D/0x1000*Potencjometry[POT_DOLNA];
    wy_g = (w-C_G)/0x1000*Potencjometry[POT_GORNA];

    #define f 1000
    #define Q 2
    #define C_DT (int)((double)0x10000/(1.0 / f / Q / 2.0 / 3.14 * CZ_PROBKOWANIA))
    #define L_DT (int)((double)0x10000/(Q / 2.0 / 3.14 / f * CZ_PROBKOWANIA))

    static int U_1 = 0;
    static int I_1 = 0;

    volatile register int uc = U_1 + ((I_1 /0x10000) * C_DT);
    volatile register int i = ((w - uc - I_1)/0x10000) * L_DT + I_1;

    U_1 = uc;
    I_1 = i;

    wy_s = i/0x1000*Potencjometry[POT_SRODEK];

    return wy_d+wy_g +wy_s;
}

/****************** LFO *********************************/
#define F_LFO2  40

double LFO(){
    static int kierunek = 1;
    static int aLFO = 0;

    if(kierunek)
    {
        aLFO += Potencjometry[POT_F_LFO]*0x10*F_LFO2/CZ_PROBKOWANIA;
        if(aLFO>=0xFFFF)
            kierunek = 0;
    } else {
        aLFO -= Potencjometry[POT_F_LFO]*0x10*F_LFO2/CZ_PROBKOWANIA;
        if(aLFO<=0)
            kierunek = 1;
    }

    return aLFO*Potencjometry[POT_A_LFO]/0x1000+(0xFFF-Potencjometry[POT_A_LFO])*0x10;
}

/**************** Sprzęganie ****************************/

volatile float S_Stala;
#define S_Omega 2000;

float Sprzeganie(float w_we) {
    S_Stala += (w_we-S_Stala)/S_Omega;
    return w_we-S_Stala;
}

/********* Główna procedura ***************************/

int Przetwarzanie(int w_we) {
    int w_wy;

    w_we = (w_we-0x7FF)*0x10000;

    w_wy = Sprzeganie(w_we);

    if(Potencjometry[POT_FUZZ]<0xF00)
    {
        w_wy = Fuzz(w_wy);
    } else {
        w_wy = w_wy/0x10000*LFO();
    }

    w_wy = Filtr(w_wy);
```

wewnętrzny układ licznika/timera. Na wyjścia te podawany jest sygnał prostokątny o odpowiednim wypełnieniu i częstotliwości około 100 kHz.

Opis programu sterującego

Działanie programu jest synchronizowane przerwaniami generowanymi przez timery TIM1 i TIM2. Timer 1 wyzwala pomiar przetwornika A/C w torze audio.

Listing 1. cd.

```
    w_wy = w_wy/0x1000*Potencjometry[POT_AMP];

    if(w_wy>0x7FFFFF)
        w_wy = 0x7FFFFF;
    if(w_wy<-0x7FFFFF)
        w_wy = -0x7FFFFF;

    w_wy = (w_wy/0x10000)+0x7FF;
    return w_wy;
}
```

Natomiast timer 2 wyzwala sekwencję pomiarów napięć na potencjometrach regulacyjnych.

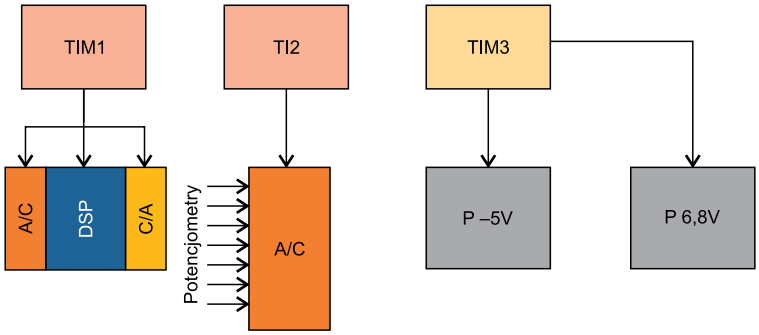
Wyzwolenie pomiaru przez timer TIM1 uruchamia cykl przetwarzania przetwornika A/C. Po jego zakończeniu zostaje wygenerowane przerwanie, które rozpoczyna procedurę przetwarzania sygnału. Po wykonaniu wszystkich operacji związanych z modyfikacją sygnału wynikowa wartość jest podawana na wejście przetwornika C/A.

Dodatkowym zadaniem programu jest generowanie przebiegów sterujących przetwornicami napięć. Całe zadanie realizuje, odpowiednio skonfigurowany, timer TIM3.

Schemat obrazujący strukturę programu został pokazany na **rysunku 5**. W programie użyto arytmetyki stało-przecinkowej, ponieważ operacje, ta tym typie danych są znacznie szybsze niż na wartościach zmiennoprzecinkowych. Na **listingu 1** zostały pokazane kody najistotniejszych procedur programu głównego, natomiast na **listingu 2** pokazano kod realizujący zadanie jednego kanału efektu fuzz. Pełne źródło programu jest dołączone do materiałów dodatkowych do projektu. Doświadczeni programiści mogą zmodyfikować kod, tworząc swoje własne efekty.

Tabela 1. Sposób podłączenia programatora do płytki urządzenia

ST – link	PCB
1	4
4	2
7	1
9	3
15	5



Rysunek 5. Schemat obrazujący strukturę programu

Montaż i uruchomienie

Schemat płytki PCB, wraz z rozmieszczeniem elementów, pokazano na **rysunku 6**.

Układ należy zmontować bardzo starannie, ponieważ zawiera małe elementy SMD. Przed uruchomieniem należy umyć płytkę i dokładnie sprawdzić, czy elementy zostały przylutowane na właściwe miejsca oraz czy nie powstały przypadkowe zwarcia na wyprowadzeniach mikrokontrolera.

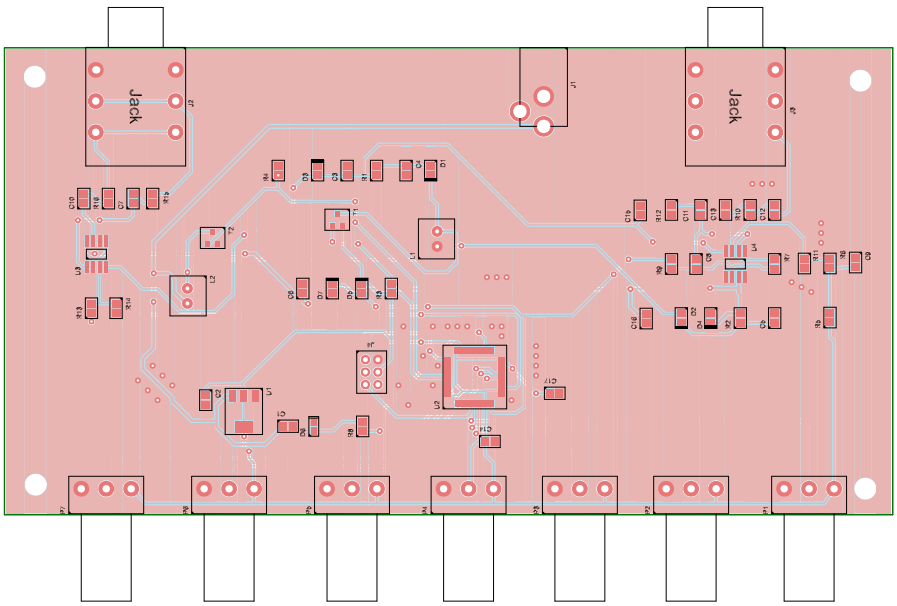
Po zmontowaniu płytki musimy zaprogramować mikrokontroler. Można do tego użyć specjalnej aplikacji dostępnej na stronie firmy ST – STB32Cubeprog. Potrzebny będzie także najpopularniejszy programator – ST-Link. W przypadku gdy nie dysponujemy odpowiednim kablem, którym można połączyć programator z płytką, możemy go wykonać samodzielnie. Kabel składa się z dwóch złączy typu IDC 6-pinowego i 20-pinowego. Złącza należy połączyć zgodnie z opisem w **tabeli 1**.

Programowanie za pomocą programu STM32Cubeprog należy przeprowadzić w następujący sposób:

1. podłączamy programator ST-link do komputera i łączymy go ze złączem na płytce,
2. podłączamy zasilanie do układu,
3. uruchamiany program STM32CubeProg,
4. wczytujemy plik *Efekt10.hex*,
5. ustawiamy typ programatora jako ST-link,
6. ustawiamy tryb programowania jako SWD,
7. wczytujemy program do pamięci flash mikrokontrolera.

Listing 2. Kod jednego kanału efektu fuzz

```
Listing jednego kanału efektu fuzz
float Fuzz1000(int w_we) {
/////////////////////////// filtrowanie
    #define f 1000
    #define C_DT (int)((double)0x10000/(1.0 / f / DOBROC / 2.0 / 3.14 * CZ_PROBKOWANIA))
    #define L_DT (int)((double)0x10000/(DOBROC / 2.0 / 3.14 / f * CZ_PROBKOWANIA))

    static int U_1 = 0;
    static int I_1 = 0;

    volatile register int uc = U_1 + ((I_1 /0x10000) * C_DT);
    volatile register int i = ((w_we - uc - I_1)/0x10000) * L_DT + I_1;

    U_1 = uc;
    I_1 = i;

/////////////////////////// wyliczanie amplitudy
    static int suma = 0;
    static int ba = 0;
    static int licznik = 1;

    if (licznik >= (int)(CZAS_MIER * CZ_PROBKOWANIA))
    {
        ba = suma;
        suma = (i<0 ? -i : i)/ (int)(CZAS_MIER * CZ_PROBKOWANIA / 3.14 * 2.0);
        licznik = 1;
    } else {
        suma += (i<0 ? -i : i)/ (int)(CZAS_MIER * CZ_PROBKOWANIA / 3.14 * 2.0);
        ++licznik;
    }

/////////////////////////// zniekształcenia
    volatile register int odc = (ba/WSP_ODC)*Potencjometry[POT_FUZZ];

    if(i>odc)
    {
        return ba;
    } else if(i<-odc) {
        return -ba;
    } else {
        return (i/Potencjometry[POT_FUZZ])*0x1000;
    }
}
```



Rysunek 6. Schemat płytki PCB wraz z rozmieszczeniem elementów (pomniejszony 35%)

Obsługa urządzenia

Parametry urządzenia reguluje się siedmioma potencjometrami. Pierwszy z nich (od lewej) to głębokość efektu „distortion”, jeśli ustawimy jego wartość na zero, wyłącza się efekt „fuzz graficzny” i włącza się efekt „tremolo”. Dwa następne potencjometry służą do ustawiania częstotliwości modulującej i współczynnika głębokości modulacji. Trzy kolejne potencjometry to trójpunktowy equalizer. Ostatni potencjometr służy do regulacji poziomu sygnału.

Układ zasilany jest napięciem 5 V. Sygnał z gitary podłącza się do wejścia „duży jack” znajdującego się z lewej strony, a sygnał wyjściowy można pobrać ze złącza znajdującego się po stronie prawej. Wyjście ma niewielką rezystancję wyjściową, więc można do niego podłączyć nawet słuchawki.

Tomasz Krogulski
krogul70@gmail.com