PROJEKT NR 239

Multiefekt dla wokalistów VEP10

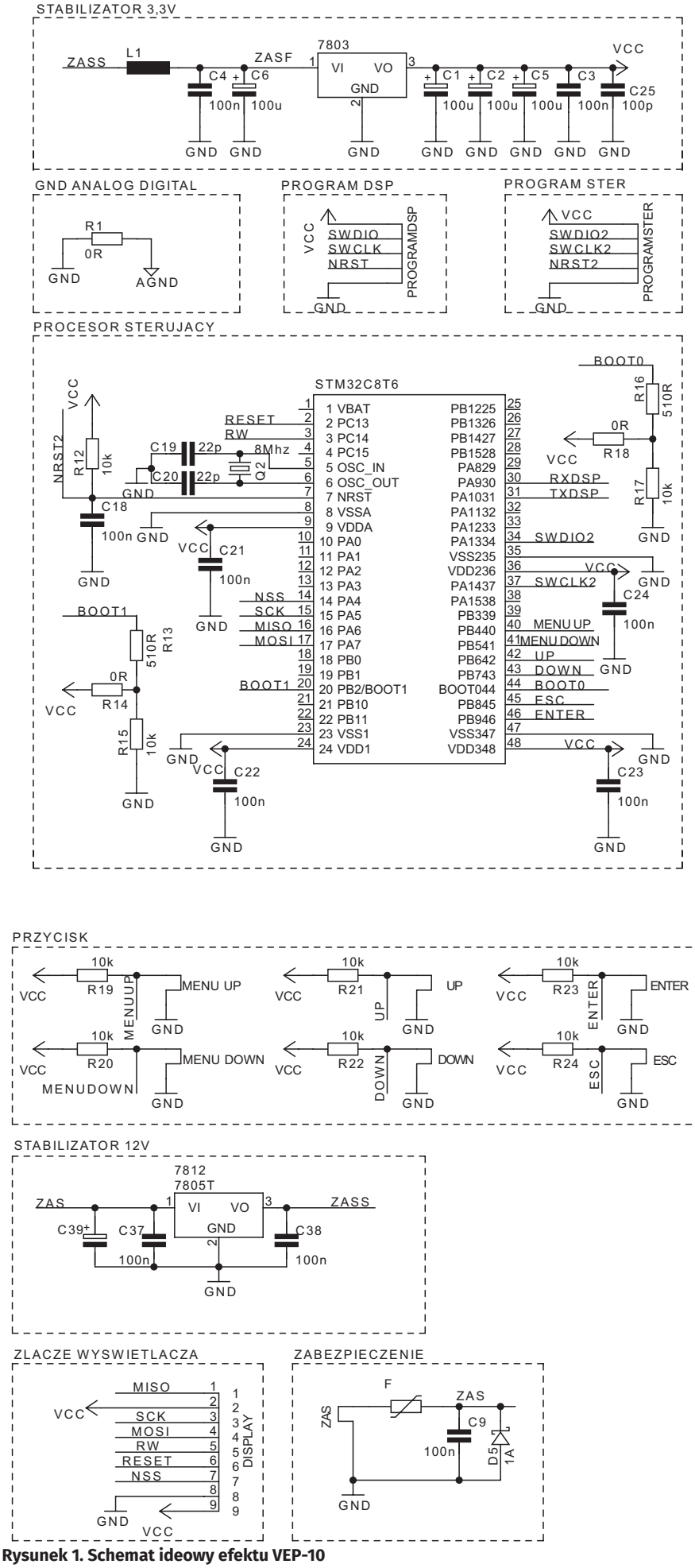
Multiefekt VEP-10 jest procesorem wokalnym, w którym użytkownik może skonfigurować 70 własnych ustawień (presetów), wykorzystując do tego aż 7 podstawowych efektów (delay, reverb, chorus, flanger, overdrive, harmonizer, ring modulation). Wszystkie podstawowe efekty można ze sobą łączyć w danym presecie. Efekt powstał jako alternatywa dla drogich multiefektów znanych producentów sprzętu wokalnego. Motywacją do jego zbudowania była także chęć rozszerzenia opisywanego wcześniej efektu DRP-10, który zawiera tylko jeden preset użytkownika oraz dwa podstawowe efekty.

Urządzenie VEP-10 tak samo jak DRP-10 ma wbudowany przedwzmacniacz. Dlatego oprócz zastosowań na scenie, można go także wykorzystać do domowego karaoke. Współpracuje przy tym tylko z mikrofonami dynamicznymi. Przed przejściem do dokładnego opisu budowy urządzenia zaprezentuję krótki opis generowanych efektów wraz z parametrami, które użytkownik może zmieniać:

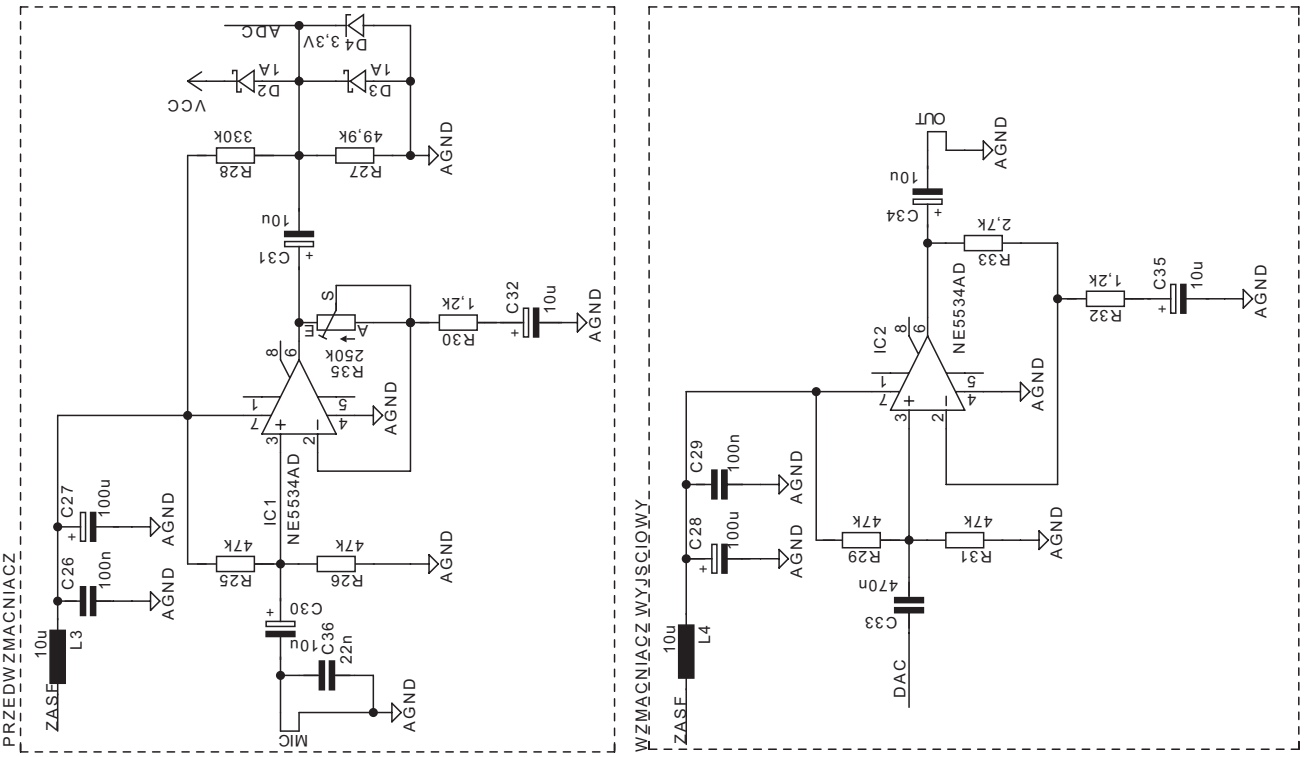
- **Delay** – efekt, w którym sygnał dźwiękowy zostaje powtórzony na wyjściu z określonym opóźnieniem (parametr OPOZNIENIE), po odpowiednim stłumieniu (TLUMIENIE). Efekt delay ma możliwość załączenia wielokrotnego powtórzenia (KROTNOŚĆ), wtedy efekt przypomina idealne echo.
- **Chorus** – efekt, w którym ustawiamy dwa różne opóźnienia swoich głosów (OPOZNIENIE1, OPOZNIENIE2). Można załączyć dwa opóźnienia lub tylko jedno (ZALCH1, ZALCH2). Opóźnienia dla efektu chorus powinny być możliwie małe. Zmieniają się one w czasie z odpowiednim skokiem czasowym (PRZYROST1, PRZYROST2) w określonym

przedziale (SZEROKOSC1, SZEROKOSC2). Dla poszczególnych opóźnień można określić tłumienia (TLUMIENIE1, TŁUMIENIE2).

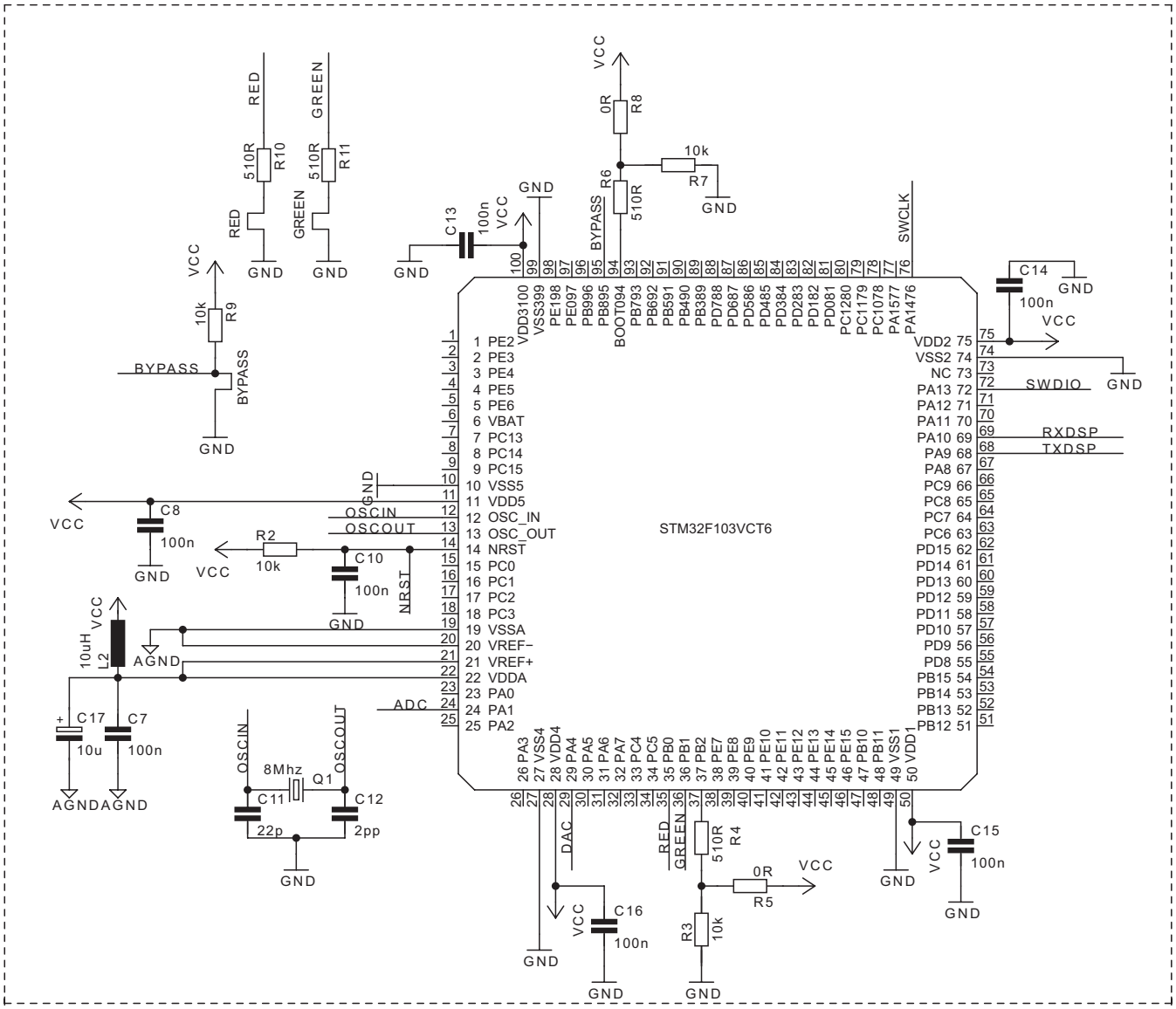
- **Flanger** – efekt, w którym także opóźnienie jest zmienne w czasie. Prędkość zmian opóźnienia można regulować (PRĘDKOŚĆ). Można także określić dokładnie przedział zmian opóźnienia (OPOZNIENIE MAX, OPOŹNIENIE MIN).
- **Reverb** – efekt rzeczywistego echa. Symuluje dźwięk odbić dźwięku od ścian pustego budynku. Możemy ustawić cztery opóźnienia symulujące odbicia od czterech ścian (OPOZNIENIE1, OPOZNIENIE2, OPOZNIENIE3, OPOZNIENIE4) oraz dwa opóźnienia symulujące odbicia wielokrotne (OPOZNIENIEC1, OPOZNIENIEC2). Dla każdego opóźnienia można określić tłumienie (TLUMIENIE1, TLUMIENIE2, TLUMIENIE3, TLUMIENIE4, TŁUMIENIEC1, TŁUMIENIEC2).
- **Overdrive** – efekt symulujący przesterowanie mikrofonu, przy czym w przeciwieństwie do efektu fuzz użytkownik może ustawić intensywność obcinania sygnału dźwiękowego w jego dolnej części i górnej (NACHYLENIEUP, NACHYLENIEDOWN). Użytkownik może sobie ustawić granicę górną, a także dolną, w których następuje rozpoczęcie obcinania sygnału (OVERDRIVEUP, OVERDRIVEDOWN).
- **Harmonize**r – efekt, który powoduje zwiększenie lub zmniejszenie częstotliwości sygnału dźwiękowego o określony interwał. Pracuje w trybie automatycznym i nie harmonizuje do częstotliwości instrumentu np. gitary. Dla harmonizera można ustawić opóźnienie dźwięku (OPOZNIENIE), ustawić odpowiedni interwał dla zwiększenia częstotliwości (INTERWAL1) lub zmniejszenia częstotliwości (INTERWAL2). Można ustawić odpowiedni tryb dla zwiększenia częstotliwości, zmniejszenia lub obu jednocześnie (TRYB). Dla zwiększenia częstotliwości można ustawić tłumienie (TLUMIENIE1) oraz dla zmniejszenia częstotliwości można ustawić tłumienie (TLUMIENIE2).
- **Ring modulator** – efekt symulujący dźwięk robota. Można zmienić intensywność modulacji poprzez zmianę prędkości działania efektu (PRĘDKOŚĆ).
- **GAIN** – tłumienie sygnału wyjściowego. Sygnał wyjściowy jest sumą poszczególnych efektów, dlatego



Rysunek 1. Schemat ideowy efektu VEP-10



PROCESOR DSP



w momencie wystąpienia przesterowania lub sprzężenia należy zwiększyć tłumienie wyjściowe (zbyt duży sygnał wyjściowy). Należy pamiętać, że podczas śpiewania i przełączania presetów mogą wystąpić dodatkowe zakłócenia spowodowane ustawieniem różnego tłumienia wyjściowego dla poszczególnych presetów, dlatego należy starać się wykonywać daną aranżację na presetach, które znajdują się w szeregu obok siebie oraz mają ustawione takie samo tłumienie.

Budowa

Na **rysunku 1** pokazano schemat ideowy multiefektu. Całość jest zasilana ze stabilizowanego zasilacza impulsowego 15 V. Napięcie to jest filtrowane i stabilizowane za pomocą 12-woltowego stabilizatora liniowego 7812. Napięcie +12 V jest oznaczone na schemacie etykietą ZASS. Przez filtr złożony z dławika L1 i kondensatora C6 jest zasilany stabilizator 3,3 V – 7803. Oba stabilizatory mają obudowy TO220. Największe straty mocy wydzielają się na stabilizatorze 3,3 V, ponieważ wyświetlacz kolorowy użyty w projekcie pobiera dużo prądu. Z mojego doświadczenia wynika, że stabilizatory impulsowe wprowadzają zakłócenia do słyszanego dźwięku i dlatego zdecydowałem się na zasilanie „hybrydowe". Rezystor R1 (0 Ω) rozdziela masy analogową i cyfrową oraz zwiera je tylko w jednym punkcie.

W urządzeniu użyto dwóch mikrokontrolerów z rodziny STM32. Pierwszy z nich STM32F103VCT6 działa jako procesor DS – pobiera próbki dźwięku za pomocą przetwornika A/C, a następnie je przetwarza. Przetworzone próbki są zamieniane na sygnał analogowy za pomocą przetwornika C/A. Procesor ma odpowiednio dużą pamięć RAM, aby buforować niezbędną liczbę próbek. Oprócz tego mikrokontroler ma moduł DMA umożliwiający pobieranie próbek bez angażowania CPU. Rdzeń może być taktowany z częstotliwością 72 MHz, co jest wystarczające w tej aplikacji.

Drugi mikrokontroler to STM32F103C8T6 odpowiedzialny za obsługę wyświetlacza oraz ustawienia poszczególnych presetów. Ma 64 kB pamięci Flash niezbędnej do zapamiętania ustawień wszystkich presetów. Przy zmianie numeru presetu procesor wysyła wszystkie ustawienia za pomocą interfejsu UART do procesora odpowiedzialnego za przetwarzanie sygnału dźwiękowego. Pary rezystor-kondensator R12/C18 oraz R2/C10 zapewniają prawidłowe zerowanie procesorów po włączeniu zasilania. Rezystory R3…R8, R13…R18 zapewniają prawidłowe bootowanie pamięci Flash mikrokontrolerów. Rezystorów R5, R8, R14, R18 nie należy lutować.

Zasilanie przetworników C/A i A/C procesora STM32F103VCT6 odbywa się za pomocą filtra LC (L2, C7, C17). Rezystory R10, R11 ograniczają prąd płynący przez diodę RG o wspólnej katodzie.

Złącze DISPLAY służy do podłączenia wyświetlacza kolorowego o przekątnej 2,2" ze sterownikiem ILI9341. Do obsługi wyświetlacza użyto biblioteki programowej dostarczonej przez producenta i odpowiednio ją zmodyfikowano dla STM32. Złącza PROGRAMDSP, PROGRAMSTER służą do zaprogramowania mikrokontrolera odpowiedzialnego za przetwarzanie sygnału (PROGRAMDSP – STM32F103VCT6) oraz procesora odpowiedzialnego za obsługę interfejsu użytkownika (PROGRAMSTER – STM32F103C8T6). Gotowe pliki wsadowe można wgrać, używając programatora ST-Link V2 i oprogramowania STM32-ST-Link Utility.

Bezpiecznik F powinien mieć wartość 800 mA. Dioda D5 zabezpiecza cały układ przed odwrotną polaryzacją napięcia zasilającego. Wzmacniacze operacyjne IC1, IC2 (NE5534) są zasilane napięciem

Wykaz elementów:
Wzmacniacz mocy
Rezystory: (SMD 1206)
R1, R5, R8, R14, R18: 0 Ω
R2, R3, R7, R9, R12, R15, R17, R19…R24: 10 kΩ
R4, R6, R10, R11, R13, R16: 510 Ω
R25, R26, R29, R31: 47 kΩ
R28: 330 kΩ
R27: 49,9 kΩ
R30, R32: 1,2 kΩ
R33: 2,7 kΩ
Kondensatory: (SMD 1206)
C3, C4, C7…C10, C13…C16, C18, C21…C24, C26, C29, C37, C38: 100 nF
C25: 100 pF
C36: 22 nF
C33: 470 nF
C11, C12, C19, C20: 22 pF
C1, C2, C5, C6, C27, C28, C39: 100 µF/25 V (elektrolit. SMD)
C17, C30…C32, C34, C35: 10 µF/16 V (elektrolit. SMD)
Półprzewodniki:
7812 (TO220)
7803 (TO220)
STM32F103C8T6
STM32F103VCT6
IC1, IC2: NE5534
D2…D5: dioda Schottky (mini MELF)
Inne:
L1: 330 µH/1 A
L2…L4: 10 µH (SMD 1206)

Listing 1. Fragment kodu odpowiedzialny za efekt chorus

```
if(zalCHORUS1 || zalCHORUS2)
{
    if(!timerCHORUS1)
    {
        timerCHORUS1=zadanytimerCHORUS1;
        if(liczgoradolCHORUS1) delaymsch1+=changeCHORUS1;
        else delaymsch1-=changeCHORUS1;
        if(delaymsch1<=((opoznienieCHORUS1-deltaCHORUS1)*50)) liczgoradolCHORUS1=1;
        if(delaymsch1>=((opoznienieCHORUS1+deltaCHORUS1)*50)) liczgoradolCHORUS1=0;
    }
    if(!timerCHORUS2)
    {
        timerCHORUS2=zadanytimerCHORUS2;
        if(liczgoradolCHORUS2) delaymsch2+=changeCHORUS2;
        else delaymsch2-=changeCHORUS2;
        if(delaymsch2<=((opoznienieCHORUS2-deltaCHORUS2)*50)) liczgoradolCHORUS2=1;
        if(delaymsch2>=((opoznienieCHORUS2+deltaCHORUS2)*50)) liczgoradolCHORUS2=0;
    }
    if(s>=delaymsch1) l2=s-delaymsch1; else l2=11000-(delaymsch1-s);
    if(s>=delaymsch2) l3=s-delaymsch2; else l3=11000-(delaymsch2-s);
    temp11=zalCHORUS1*buffor[l2];
    TEMPGAIN=temp11*wzmocnienie[wzmocnienieCHORUS1];
    temp11=TEMPGAIN/10000;
    temp22=zalCHORUS2*buffor[l3];
    TEMPGAIN2=temp22*wzmocnienie[wzmocnienieCHORUS2];
    temp22=TEMPGAIN2/10000;
    wartoscCHORUS=temp11+temp22;
}
```

Listing 2. Fragment kodu realizujący efekt flanger

```
if(zalaczFLANGER)
{
    if(!timerFLANGER)
    {
        timerFLANGER=zadanytimerFLANGER;
        if(flagaFLANGERdodajodejmij) delaymsfl++; else delaymsfl--;
        if(delaymsfl>=(opoznienieFLANGERmax*50)) flagaFLANGERdodajodejmij=0;
        if(delaymsfl<=(opoznienieFLANGERmin*50)) flagaFLANGERdodajodejmij=1;
    }
    if(s>=delaymsfl) l4=s-delaymsfl; else l4=11000-(delaymsfl-s);
    TEMPGAIN=(buffor[l4]*wzmocnienie[wzmocnienieFLANGER]);
    wartoscFLANGER=TEMPGAIN/10000;
}
```

Listing 3. Fragment kodu wykrywający, czy próbkę należy zastąpić, czy stłumić

```
if(zalOVERDRIVE)
{
    if(flagaOVERDRIVE)
    {
        overdrivegain=buffor[s]-skladowastala;
        overdrivegain=overdrivegain*2;
        bufforoverdrive=overdrivegain+skladowastala;
        if((bufforoverdrive>(wartoscOVERDRIVEgora-zmiennaoverdrive1))
            && (bufforoverdrive<wartoscOVERDRIVEgora)) flagaOVERDRIVEgora=1;
        else if((bufforoverdrive<(wartoscOVERDRIVEdol+zmiennaoverdrive2))
                && (bufforoverdrive>wartoscOVERDRIVEdol)) flagaOVERDRIVEdol=1;
            else if(bufforoverdrive>=wartoscOVERDRIVEgora) flagaOVERDRIVEobcieciegora=1;
                else if(bufforoverdrive<=wartoscOVERDRIVEdol) flagaOVERDRIVEobcieciedol=1;
                    else flagaOVERDRIVEsrodek=1;
    }
}
```

asymetrycznym +12 V poprzez filtry LC (L3, C26, C27 oraz L4, C28, C29). Na złącze MIC dociera sygnał z mikrofonu dynamicznego. Kondensator C36 służy do filtrowania zaburzeń odbieranych przez przewód mikrofonowy. Kondensator C30 służy do usunięcia składowej stałej z sygnału dźwiękowego. Rezystory R25, R26 podnoszą sygnał dźwiękowy o składową stałą równą połowie zasilania wzmacniacza operacyjnego. Kondensator C32 służy do ustalenia na nim napięcia 6 V z powodu niesymetrycznego zasilania wzmacniacza operacyjnego. Potencjometr R35 oraz rezystor R30 ustalają wzmocnienie sygnału według zależności k_u=(1+R35/R30). Kondensator C31 służy do usunięcia składowej stałej. Sygnał bez składowej stałej następnie jest przesuwany o około +1,5 V za pomocą rezystorów R27, R28 w celu dopasowania do przetwornika A/C. Diody Schottky'ego D2, D3, D4 zabezpieczają przetwornik przed napięciem wyższym od 3,5 V lub mniejszym niż –0,2 V. Sygnał przetworzony z procesora jest pozbawiany składowej stałej za pomocą kondensatora C33, a następnie jest do niego dodawana składowa +6 V za pomocą rezystorów R29, R31. Podniesiony sygnał jest wzmacniany ze wzmocnieniem k_u=3,25 ustalonym za pomocą rezystorów R32, R33. Kondensator C34 usuwa ze wzmocnionego sygnału składową stałą. Złącze OUT służy do przylutowania gniazda wyjściowego, do którego podłącza się wzmacniacz.

Oprogramowanie

Główne zadanie wykonuje mikrokontroler odpowiedzialny za przetwarzanie sygnału – STM32F103VCT6. Zasada działania opiera się na pobraniu określonej liczby próbek sygnału dźwiękowego i przechowywaniu ich w pamięci RAM. Próbki są pobierane z częstotliwością powyżej 40 kHz. Gdy bufor zostaje przepełniony, wtedy zapisywanie w buforze zaczyna się na nowo, a stare próbki są nadpisywane nowymi.

Pobrane próbki są przekształcane przez CPU w celu uzyskania odpowiedniego efektu dźwiękowego. Efekt delay polega na tym, że do aktualnej pobranej próbki są dodawane próbki pobrane z opóźnieniem 0…200 ms, a następnie wartość ta jest przekazywana na przetwornik C/A. Próbki opóźnione są odpowiednio stłumione według skali logarytmicznej. Delay wielokrotny jest realizowany w taki sposób, że zamiast zapisywać do bufora aktualną próbkę, jest w nim zapamiętywana suma aktualnej próbki oraz próbki opóźnionej i stłumionej. Efekt reverbu realizowany jest w taki sposób, że cztery próbki o różnych opóźnieniach oraz różnych tłumieniach są sumowane, a następnie zapisywane do dwóch dodatkowych buforów. Bufory służą do wprowadzenia dodatkowych opóźnień i mają symulować wielokrotne odbicia oraz nakładanie się fal dźwiękowych w zamkniętym pomieszczeniu. Próbki o różnych opóźnieniach z dodatkowych buforów są odpowiednio tłumione, a następnie są sumowane ze sobą oraz z czterema wcześniej opóźnionymi próbkami. Próbki będące sumą są przekazywane na przetwornik C/A wraz z aktualną próbką sygnału dźwiękowego. Kod programu dla efektu delay oraz reverb jest taki sam, jak w efekcie wokalnym DRP-10.

Efekt chorus (**listing 1**) polega na zsumowaniu dwóch próbek opóźnionych. Opóźnienia dla tego efektu mają małą wartość, różnią się od siebie oraz zmieniają się w czasie. Opóźnione próbki następnie są odpowiednio tłumione oraz sumowane z aktualnie pobraną próbką.

Efekt flanger (**listing 2**) działa tak samo, jak delay, lecz wartość opóźnienia zmienia się w czasie w szerokim przedziale. Bardzo istotna jest możliwość zmiany prędkości zmian opóźnienia, co przyczynia się do możliwości uzyskania różnych efektów. W programie można także definiować maksymalne oraz minimalne opóźnienie, czyli przedział wartości, w którym zmienia się wartość opóźnienia.

Efekt overdrive (**listing 3**) jest efektem symulującym przesterowanie. Działanie polega na pobraniu próbki sygnału, a następnie sprawdzeniu przez dwa komparatory, czy amplituda sygnału mieści się w określonym przedziale [k_d, k_g]. Jeżeli wartość próbki nie spełnia zależności k_d<U_{probki}<k_g, wtedy zostaje ona odpowiednio stłumiona. W zależności od tego jaką chce się uzyskać wartość efektu przesterowania, tłumienie może być mocniejsze lub słabsze. Dla silnego efektu przesterowania próbka zostaje zastąpiona

Listing 4. Kod obliczający intensywność tłumienia

```
if(flagaOVERDRIVEgora)
{
    flagaOVERDRIVEgora=0;
    overdriveTEMPgora=bufforoverdrive-wartoscOVERDRIVEgora-zmiennaoverdrive1;
    x1g=overdriveTEMPgora;
    x2g=x1g*x1g;
    x3g=x1g*x1g*x1g;
    x2g=x2g/40;
    x3g=x3g/2400;
    overdriveTEMPgora=x1g-x2g+x3g;
    overdriveTEMPgora=overdriveTEMPgora*zmiennaoverdrive1;
    overdriveTEMPgora=overdriveTEMPgora/20;
    overdriveTEMPgora=overdriveTEMPgora+wartoscOVERDRIVEgora-zmiennaoverdrive1;
    wartoscDAC=overdriveTEMPgora+wartoscDELAY+wartoscCHORUS+wartoscFLANGER
              +zalaczREVERB1*(wartoscREV1)+wartoscHARMONIZER + wartoscHARMONIZER2;
}

if(flagaOVERDRIVEdol)
{
    flagaOVERDRIVEdol=0;
    overdriveTEMPdol=wartoscOVERDRIVEdol+zmiennaoverdrive2-bufforoverdrive;
    x1d=overdriveTEMPdol;
    x2d=x1d*x1d;
    x3d=x1d*x1d*x1d;
    x2d=x2d/40;
    x3d=x3d/2400;
    overdriveTEMPdol=x1d-x2d+x3d;
    overdriveTEMPdol=overdriveTEMPdol*zmiennaoverdrive2;
    overdriveTEMPdol=overdriveTEMPdol/20;
    overdriveTEMPdol=(wartoscOVERDRIVEdol+zmiennaoverdrive2-overdriveTEMPdol;
    wartoscDAC=overdriveTEMPdol+wartoscDELAY+wartoscCHORUS+wartoscFLANGER
              +zalaczREVERB1*(wartoscREV1)+wartoscHARMONIZER + wartoscHARMONIZER2;
}

if(flagaOVERDRIVEsrodek)
{
    flagaOVERDRIVEsrodek=0;
    wartoscDAC=bufforoverdrive+wartoscDELAY+wartoscCHORUS+wartoscFLANGER
              +zalaczREVERB1*(wartoscREV1)+wartoscHARMONIZER + wartoscHARMONIZER2;
}

if(flagaOVERDRIVEobcieciegora)
{
    flagaOVERDRIVEobcieciegora=0;
    overdriveTEMPgora=zmiennaoverdrive1;
    x1g=overdriveTEMPgora;
    x2g=x1g*x1g;
    x3g=x1g*x1g*x1g;
    x2g=x2g/40;
    x3g=x3g/2400;
    overdriveTEMPgora=x1g-x2g+x3g;
    overdriveTEMPgora=overdriveTEMPgora*zmiennaoverdrive1;
    overdriveTEMPgora=overdriveTEMPgora/20;
    overdriveTEMPgora=overdriveTEMPgora+wartoscOVERDRIVEgora-zmiennaoverdrive1;
    wartoscDAC=overdriveTEMPgora+wartoscDELAY+wartoscCHORUS+wartoscFLANGER
              +zalaczREVERB1*(wartoscREV1)+wartoscHARMONIZER + wartoscHARMONIZER2;
}

if(flagaOVERDRIVEobciecieddol)
{
    flagaOVERDRIVEobciecieddol=0;
    overdriveTEMPdol=zmiennaoverdrive2;
    x1d=overdriveTEMPdol;
    x2d=x1d*x1d;
    x3d=x1d*x1d*x1d;
    x2d=x2d/40;
    x3d=x3d/2400;
    overdriveTEMPdol=x1d-x2d+x3d;
    overdriveTEMPdol=overdriveTEMPdol*zmiennaoverdrive2;
    overdriveTEMPdol=overdriveTEMPdol/20;
    overdriveTEMPdol=wartoscOVERDRIVEdol+zmiennaoverdrive2-overdriveTEMPdol;
    wartoscDAC=overdriveTEMPdol+wartoscDELAY+wartoscCHORUS+wartoscFLANGER
              +zalaczREVERB1*(wartoscREV1)+wartoscHARMONIZER+wartoscHARMONIZER2;
}
```

wartością k_g (gdy $U_{probki} > k_g$) lub k_d (gdy $U_{probki} < k_d$), natomiast dla słabego efektu wierzchołek sygnału zostaje stłumiony. Intensywność efektu przesterowania może być regulowana. Na **listingu 4** zaprezentowano kod obliczający intensywność tłumienia.

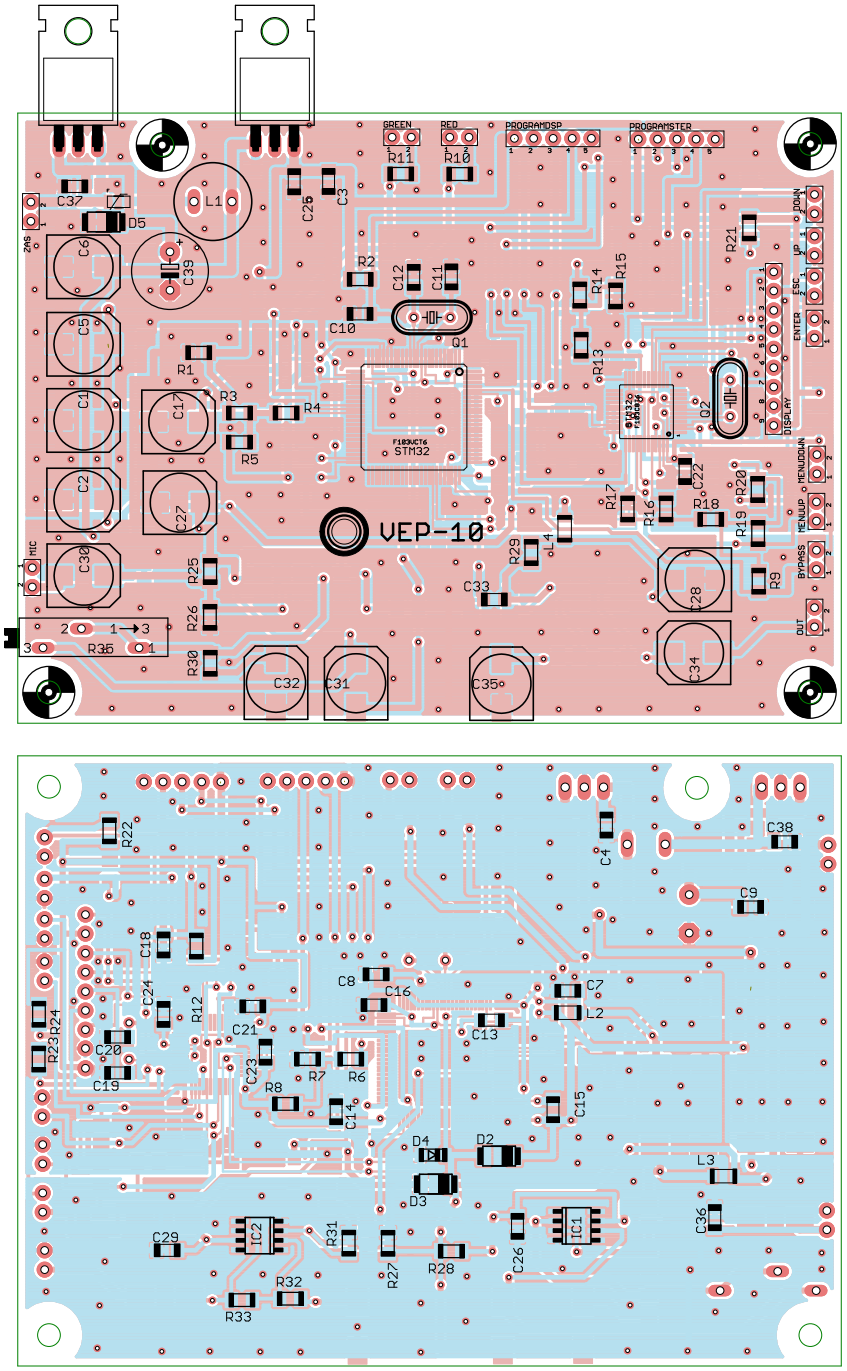
Efekt harmonizera (**listing 5**), który zwiększa lub zmniejsza częstotliwość dźwięku, jest realizowany w taki sposób, że do dodatkowego bufora są przepisywane wybrane próbki opóźnione, które nie spowodują zniekształceń sygnału dźwiękowego, natomiast spowodują zwiększenie częstotliwości dźwięku (niektóre próbki z bufora głównego są pomijane). W przypadku gdy częstotliwość ma być zmniejszona, do dodatkowego bufora są wielokrotnie przepisywane niektóre próbki (niektóre próbki z bufora głównego są powtarzane). Odstępy pomiędzy próbkami powtarzanymi lub pomijanymi musi być stała. Zmieniając odstępy pomiędzy próbkami powtarzanymi (lub pomijanymi), zmieniamy częstotliwość. Próbki z bufora dodatkowego po odpowiednim stłumieniu są sumowane z aktualnie pobrana próbką i przekazywane do przetwornika DAC procesora.

Ring modulation (**listing 6**) powstaje poprzez przemnożenie aktualnego sygnału dźwiękowego i sygnału sinusoidalnie zmiennego. W programie jest to realizowane w taki sposób, że w momencie uruchomienia efektu jest pobierana wartość składowej stałej na przetworniku A/C (około 1,5 V), a następnie z pobranych próbek sygnału jest usuwana ta składowa. Próbki bez składowej stałej są mnożone przez sygnał sinusoidalny. Sygnał sinusoidalny został zamieszczony w postaci tablicy w programie. Na sam koniec do próbek będących wynikiem mnożenia dodawana jest wcześniej usunięta wartość składowej stałej. Zmieniając częstotliwość sygnału sinusoidalnego, zmieniamy intensywność efektu.

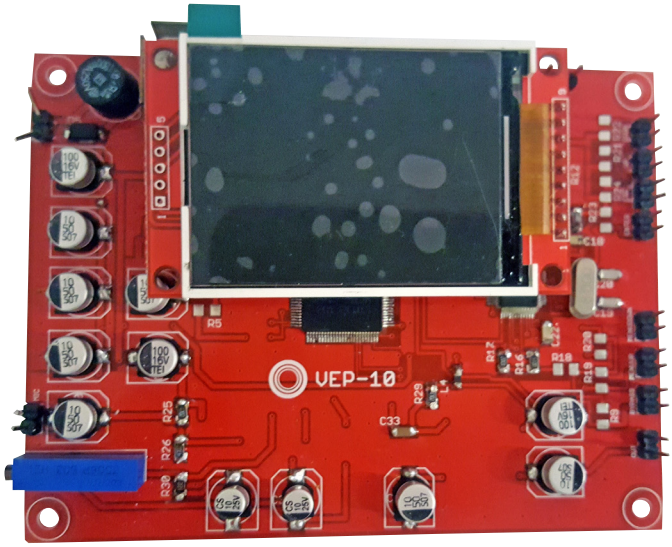
Montaż i uruchomienie

Schemat montażowy efektu VFP-10 pokazano na **rysunku 2**. Podczas projektowania płytki drukowanej puste miejsca pokryto powierzchnią masy w celu poprawienia ekranowania. Ponadto masę rozdzielono na cyfrową oraz analogową i połączono w jednym punkcie (R1). W celu lepszego połączenia masy na górze płytki z masą na spodzie wykonano dodatkowe przelotki.

Montaż należy rozpocząć od przylutowania wszystkich elementów zasilania, a następnie dołączenia zasilania i sprawdzenia poprawności napięć +12 V oraz +3,3 V. Jeżeli napięcia zasilania są prawidłowe, należy przylutować mikrokontrolery. Do lutowania mikrokontrolerów zaleca się zastosowanie grotu typu minifala. Po przylutowaniu procesorów należy przylutować wzmacniacze operacyjne, a następnie pozostałe elementy SMD. Na koniec lutujemy elementy przewlekane

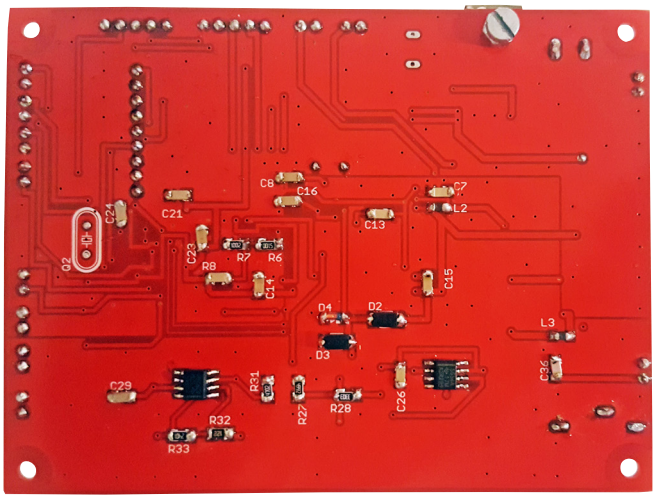


Rysunek 2. Schemat montażowy efektu VFP-10



Fotografia 3. Zmontowana płytka w widoku od góry

```
Listing 5. Fragment kodu realizujący efekt harmonizera
if(zalHarmonizer)
{
    if(flagaHARMONIZER) //flaga co 0,02ms
    {
        flagaHARMONIZER=0;
        delayHARMONIZER=50*opoznienieHARMONIZER;
        if(licznikharmonizer>wspolczynnikHARMONIZER)
        {
            r++;
            licznikharmonizer=0;
        }
        if(s>delayHARMONIZER) lh=s-delayHARMONIZER; else lh=10999-delayHARMONIZER-s;
        if((zwiekszzmniejszHARMONIZER==1)  || (zwiekszzmniejszHARMONIZER==3))
        {
            o=lh+2+r; //wspolczynnik zwiekszenia czestotliwosci
            if((2+r)>10999) r=0;
            if(o>10999) o=o-10999;
            wartoscHARMONIZER=buffor[o];
        }
        if((zwiekszzmniejszHARMONIZER==2) || (zwiekszzmniejszHARMONIZER==3))
        {
            if(c==1)
            {
                u=lh; //wpisanie opoznienia do C
                c=0; //wykasowanie flagi
            }
            d=d+wspolczynnikHARMONIZER2;
            if(d>100)
            {
                d=d-100;
                u+=1;
                if(u>10999) u=u-10999;
            }
            wartoscHARMONIZER2=buffor[u];
        }
        TEMPGAIN=wartoscHARMONIZER*wzmocnienie[wzmocnienieHARMONIZER];
        wartoscHARMONIZER=TEMPGAIN/10000;
        TEMPGAIN2=wartoscHARMONIZER2*wzmocnienie[wzmocnienieHARMONIZER2];
        wartoscHARMONIZER2=TEMPGAIN2/10000;
    }
}
if((zalHarmonizer==0) || (zwiekszzmniejszHARMONIZER==1)) c=1;
```



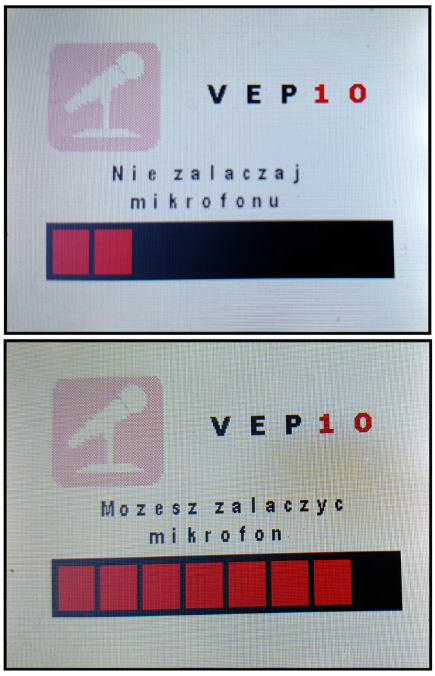
Fotografia 4. Zmontowana płytka w widoku od spodu

(diodę RG, rezonatory kwarcowe) Zmontowaną płytkę drukowaną zaprezentowano na **fotografiach 3** i **4**. Potencjometr należy dołączyć do płytki za pomocą dobrego przewodu ekranowanego, podobnie jak gniazda XLR (mikrofonu i wyjściowe). W pobliżu gniazda XLR należy przylutować kondensator 10 nF pomiędzy gniazdem sygnałowym a masą w celu dodatkowej eliminacji zaburzeń. Przed dołączeniem wyświetlacza, trzeba zaprogramować oba procesory. Efekt należy umieścić w aluminiowej obudowie, żeby cały był ekranowany.

Użytkowanie

Przed pierwszym uruchomieniem w układzie z mikrofonem oraz wzmacniaczem należy wyciszyć kanał, który jest podłączony

```
Listing 6. Fragment kodu realizującego efekt ring modulation
if(!zalRING) buffor[k]=(uint16_t)
(buforADC+(zalDELAYwielokrotny*wartoscDELAY)+(zalFLANGERwielokrotny*wartoscFLANGER));
else
{
    temporary=buforADC-skladowastala;
    temporary=temporary*sinus[tempsinus1];
    temporary=temporary/2000;
    buffor[s]=temporary+skladowastala;
    if((buffor[s]>(skladowastala+blokowanie))||(buffor[s]<(skladowastala-blokowanie)))
        buffor[s]+=(zalDELAYwielokrotny*wartoscDELAY+(zalFLANGERwielokrotny*wartoscFLANGER);
    else
buffor[s]=buforADC+(zalDELAYwielokrotny*wartoscDELAY)+(zalFLANGERwielokrotny*wartoscFLANGER);
}
if(zalRING && inkrementacjaRING>45) inkrementacjaRING=45;
```



Rysunek 5. Komunikat wyświetlany po załączeniu urządzenia

Fotografia 6. Widok menu głównego

do efektu na wzmacniaczu, wyłączyć mikrofon, aby uniknąć niepotrzebnych sprzężeń. Należy pamiętać, aby przy każdym uruchamianiu efektu wyłączyć mikrofon lub odłączyć go od wejścia MIC efektu w celu prawidłowej kalibracji. W trakcie ładowania ustawień efektu na wyświetlaczu zostanie wyświetlony komunikat o możliwości przyłączenia mikrofonu (**rysunek 5**).

Przed przystąpieniem do używania efektów należy ustawić odpowiednie wzmocnienie dla danego mikrofonu, który będzie używany z efektem. Śpiewając do mikrofonu z wyłączonymi efektami (BYPASS załączony), należy obserwować diodę LED RG, informującą o prawidłowym wyregulowaniu wzmocnienia. Dioda nie może świecić na czerwono. Wzmocnienie powinno być ustawione na granicy zmiany koloru diody, tzn. dioda zaświeca się na zielono, ale delikatne zwiększenie wzmocnienia lub głośniejsze śpiewanie do mikrofonu powoduje delikatne zaświecenie diody czerwonej. Zmiany wzmocnienia dokonujemy za pomocą potencjometru GAIN.

Wciśnięcie przycisku BYPASS powoduje wyłączenie wszystkich efektów. Sygnał dźwiękowy zostaje wtedy z mikrofonu na wyjściu OUT tylko i wyłącznie wzmocniony przez przedwzmacniacz. Ponowne

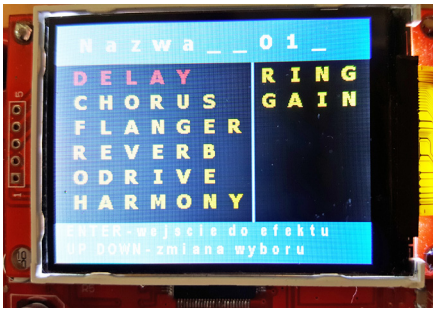
Listing 7. Kod programu, w którym następuje pobranie składowej

```
delay(250000); //odczekaj 5s
bufforskladowa[0]=buforADC;
delay(500); //odczekaj 10ms
bufforskladowa[1]=buforADC;
delay(500); //odczekaj 10ms
bufforskladowa[2]=buforADC;
delay(500); //odczekaj 10ms
bufforskladowa[3]=buforADC;
skladowastala=bufforskladowa[0]+bufforskladowa[1]+bufforskladowa[2]+bufforskladowa[3];
skladowastala=skladowastala/4;
```

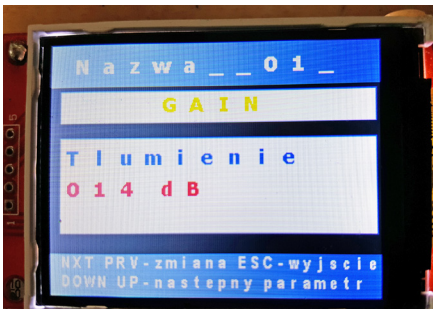
wciśnięcie przycisku BYPASS powoduje załączenie ustawionych wcześniej efektów.

Do poruszania się po menu służą przyciski ENTER, ESC, UP, DOWN oraz przyciski, które służą do zmiany numeru presetu, tj. NEXT, PREV. Na dolnym pasku wyświetlacza pojawiają się komunikaty ułatwiające poruszanie się po menu. Na **fotografii 6** pokazano menu główne, które pojawia się po załadowaniu urządzenia. Na środku wyświetlacza widnieje numer aktualnie wybranego presetu. Aby zmienić numer presetu, należy nacisnąć NEXT lub PREV. Aby wejść do podmenu wyboru edycji efektu, należy nacisnąć ENTER. Naciśnięcie przycisku ESC powoduje załączenie edycji nazwy danego presetu. Przy załączeniu edycji nazwy, naciskając przyciski UP, DOWN, należy edytować nazwę efektu. Aby wyjść z edycji nazwy, należy nacisnąć przycisk ENTER lub ESC. Naciśnięcie przycisku ENTER powoduje wyjście bez zapisu nazwy tzn. przy ponownym uruchomieniu efektu na danym numerze presetu będzie widnieć stara nazwa. Wyjście z edycji nazwy poprzez naciśniecie ESC powoduje zapisanie do pamięci flash efektu nowej nazwy, dzięki czemu po ponownym uruchomieniu efektu będzie widoczna na danym presecie nowa nazwa.

Po wejściu z głównego menu do podmenu wyboru edycji efektu poprzez naciśniecie przycisku ENTER na wyświetlaczu pokazuje się okno jak na **fotografii 7**. Naciskając UP, DOWN, możemy zmieniać zaznaczenie danego efektu, który będziemy edytować. Aktualnie zaznaczony efekt jest napisany kolorem czerwonym. Naciśnięcie ESC powoduje wyjście do głównego menu, natomiast naciśnięcie ENTER powoduje wejście do edycji danego efektu w aktualnie ustawionym presecie. Naciśnięcie przycisku NEXT, PREV powoduje powrót do głównego podmenu wraz z zapisem do pamięci flash wprowadzonych zmian. Po zapisie i ponownym uruchomieniu efektu wprowadzone zmiany będą pamiętane przez efekt. Po wyborze danego efektu i wejściu do jego edycji poprzez naciśnięcie przycisku ENTER na wyświetlaczu



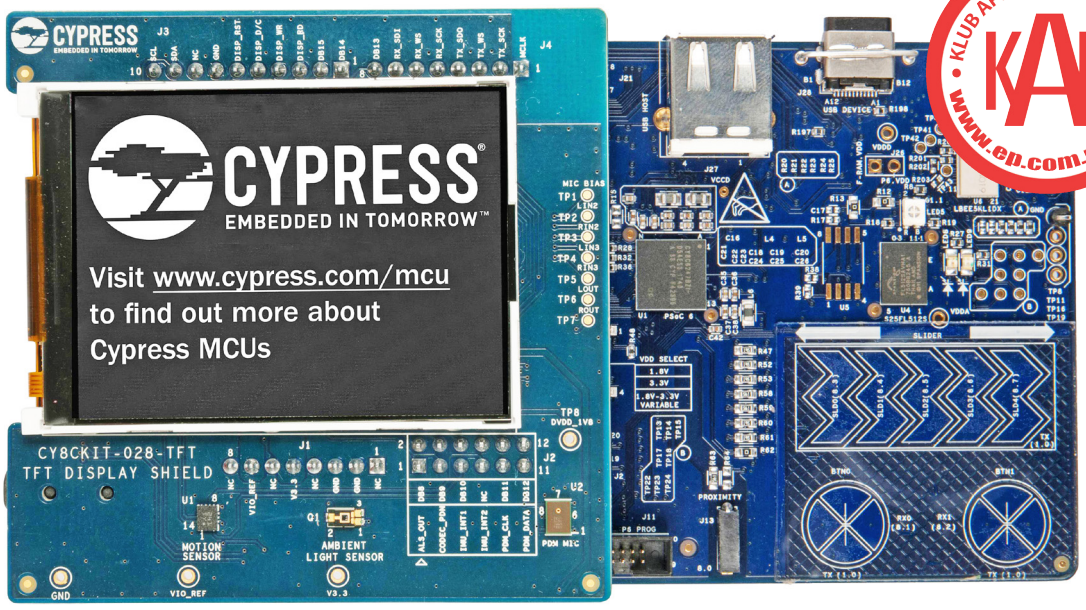
Fotografia 7. Podmenu wyboru edycji efektu



Fotografia 8. Okno edycji danego efektu w aktualnie ustawionym presecie

jest pokazywane okno jak na **fotografii 8**. Naciskając UP, DOWN, możemy zmieniać parametr efektu, który chcemy edytować. Aby zmienić wartość aktualnie ustawionego parametru, należy nacisnąć lub nacisnąć i przytrzymać przycisk NEXT lub PREV.

Krzysztof Miękus
lordwest1989@tlen.pl



Cypress Semiconductor CY8CKIT-062-BLE Pioneer Kit.

Dzięki uprzejmości firmy Farnell Element14 mamy do przekazania wybranemu czytelnikowi zestaw rozwojowy z mikrokontrolerem PSoC 6 firmy Cypress Semiconductor. MCU zawiera po jednym rdzeniu Cortex-M4 i Cortex-M0+. W zestawie dostępne jest 1 MB pamięci Flash, 288 kB pamięci SRAM i 78 linii GPIO, a także 7 programowalnych bloków analogowych i 56 programowalnych bloków cyfrowych. Układ wspiera Bluetooth Low Energy. Cennym dodatkiem jest dostarczany w zestawie wyświetlacz E-Ink o przekątnej 2,7", zamontowany na płytce z termistorem, 6-osiowym czujnikiem ruchu i cyfrowym mikrofonem. Dostępny jest też suwak dotykowy, wbudowany programator i wyprowadzenie do podłączania płytek rozszerzeń Arduino.
Wartość zestawu to około 260 zł. Aby otrzymać zestaw należy wysłać swoje zgłoszenie na adres grzegorz.becker@ep.com.pl.