Politechnika Wrocławska
Wydział Elektroniki, Fotoniki i Mikrosystemów

KIERUNEK: Automatyka i Robotyka (AIR)

PRACA DYPLOMOWA INŻYNIERSKA

TYTUŁ PRACY:
Cyfrowe przetwarzanie sygnałów audio na potrzeby interakcji człowiek–robot

AUTOR:
Piotr Siembab

PROMOTOR:
Dr inż. Robert Muszyński,
Katedra Cybernetyki i Robotyki

WROCŁAW 2025

Streszczenie

Przedmiotem pracy jest projekt oraz realizacja systemu cyfrowego przetwarzania sygnałów audio w czasie rzeczywistym, przeznaczonego do zastosowań w interakcji człowiek–robot. Celem było stworzenie platformy umożliwiającej modyfikację głosu robota w celu poprawy naturalności komunikacji.

W części teoretycznej przeanalizowano zagadnienia psychoakustyczne wpływające na percepcję maszyn, w tym rolę barwy dźwięku i zjawisko doliny niesamowitości. Warstwa implementacyjna, oparta na architekturze ARM Cortex-M7, wykorzystuje zaawansowane algorytmy DSP. Zastosowano filtry IIR w strukturze bi-kwadratowej do korekcji barwy, algorytmy splotu dyskretnego do symulacji akustyki pomieszczeń oraz nieliniowe przekształcenia amplitudy. Dynamikę sygnału kształtuje bramka szumów z algorytmem śledzenia obwiedni.

Stabilność strumienia audio zapewniono dzięki wykorzystaniu mechanizmu DMA i podwójnego buforowania. Przeprowadzone testy odsłuchowe i wydajnościowe potwierdziły poprawność działania algorytmów oraz osiągnięcie założonych celów projektowych.

Słowa kluczowe: cyfrowe przetwarzanie sygnałów, DSP, interakcja człowiek–robot, systemy wbudowane, filtry IIR, splot dyskretny, psychoakustyka

Abstract

The subject of this thesis is the design and implementation of a real-time digital audio signal processing system intended for human-robot interaction applications. The goal was to create a platform enabling the modification of a robot's voice to improve communication naturalness.

The theoretical part analyzes psychoacoustic factors affecting machine perception, including timbre and the uncanny valley phenomenon. The implementation layer, based on the ARM Cortex-M7 architecture, utilizes advanced DSP algorithms. IIR filters in a biquad structure were applied for equalization, along with discrete convolution algorithms for room acoustics simulation and nonlinear amplitude transformations. Signal dynamics are shaped by a noise gate with an envelope tracking algorithm.

Audio stream stability was ensured through the use of DMA and double-buffering mechanisms. Conducted listening and performance tests confirmed the correct operation of the algorithms and the achievement of the project objectives.

Keywords: digital signal processing, DSP, human-robot interaction, embedded systems, IIR filters, discrete convolution, psychoacoustics

Pracę dedykuję mojej rodzinie, która wspierała mnie na każdym kroku ścieżki edukacji. Szczególne podziękowania dla Pana dr. inż. Roberta Muszyńskiego za wytrwałość w pomocy i zaangażowanie.

Spis treści

1 Wstęp **3**

2 Wykorzystanie algorytmów DSP w interakcji człowiek-robot **5**

2.1 Modulacja widmowa a percepcja fizyczności 5

2.2 Nieliniowości jako biologiczne sygnały pobudzenia 6

2.3 Zarządzanie ciszą i poczuciem obecności 6

2.4 Przestrzeń i ucieleśnienie 7

2.5 Wnioski projektowe 7

3 Przetwarzanie sygnałów o częstotliwościach akustycznych **8**

3.1 Biblioteki i narzędzia do przetwarzania sygnałów 9

3.2 Korektor pasmowy 10

3.2.1 Projektowanie charakterystyk filtrów 10

3.2.2 Filtracja IIR drugiego rzędu 11

3.3 Algorytmy przetwarzania dynamiki 12

3.3.1 Bramka szumów 13

3.4 Nieliniowe zniekształcenia sygnału 14

3.4.1 Overdrive 16

3.4.2 Fuzz 16

3.4.3 Distortion 18

3.5 Efekty oparte na liniach opóźniających 19

3.5.1 Delay 19

3.5.2 Chorus 20

3.6 Przetwarzanie splotowe 21

4 Projekt systemu **24**

4.1 Założenia projektowe 24

4.1.1 Wymagania funkcjonalne 24

4.1.2 Wymagania niefunkcjonalne 25

4.2 Dobór platformy sprzętowej 25

4.2.1 Wybór mikrokontrolera 26

4.2.2 Peryferia audio 26

4.3 Realizacja sprzętowa 27

4.3.1 Zasilanie układów 27

4.3.2 Tor przetwarzania sygnału analogowego 27

4.3.3 Konfiguracja peryferiów i mapowanie wyprowadzeń 29

4.4 Architektura oprogramowania 29

4.4.1 Przepływ danych i technika podwójnego buforowania 31

4.4.2 Potok przetwarzania sygnału . . . 33
4.4.3 Organizacja pamięci w STM32H7 . . . 33
4.4.4 Warstwa interfejsu i sterowania . . . 33

5 Weryfikacja działania systemu i testy 34
5.1 Analiza toru przetwarzania . . . 34
5.2 Efekty nieliniowe . . . 35
5.2.1 Overdrive . . . 35
5.2.2 Distortion . . . 35
5.2.3 Fuzz . . . 36
5.3 Korektor pasmowy (Equalizer) . . . 37
5.3.1 Pasmo niskie (Low Shelf, 100 Hz) . . . 37
5.3.2 Pasmo środkowe (Peaking, 1000 Hz) . . . 37
5.3.3 Pasmo wysokie (High Shelf, 2000 Hz) . . . 38
5.4 Bramka szumów . . . 38
5.5 Efekty czasowe . . . 40
5.5.1 Delay . . . 40
5.5.2 Chorus . . . 40
5.6 Przetwarzanie splotowe . . . 41
5.6.1 Symulacja „Master of Puppets” . . . 41
5.6.2 Symulacja „Kill 'Em All” . . . 41

6 Podsumowanie 43

Literatura 44

Spis rysunków 48

A Schemat układu elektronicznego 49

B Zdjęcia Projektu 51

Do składu pracy wykorzystano system przygotowania dokumentów LaTeX, opracowany przez L. Lamporta [Lam94], będący nakładką systemu TeX, [Knu86a,Knu86b]. Matematyczne czcionki o nazwie AMS Euler, których używamy w tej pracy, zostały przygotowane przez H. Zapfa [KZ86], przy współpracy z D. Knuthem i jego studentami, na zlecenie Amerykańskiego Towarzystwa Matematycznego. Wybrane czcionki składu tekstu, Antykwa Toruńska [Now97] – jeden z nielicznych krojów pisma zaprojektowany specjalnie dla języka polskiego w sposób uwzględniający jego rytm – w odczuciu autora doskonale współgrają z kształtem czcionki AMS Euler, pozwalając na uzyskanie harmonijnej całości. Składu bezszeryfowego tekstu maszynowego dokonano z użyciem opracowanej przez R. Leviena czcionki o nazwie Inconsolata

[Knu86a] D. E. Knuth, The TeXbook, volume A of Computers and Typesetting. Addison-Wesley, Reading, 1986.

[Knu86b] D. E. Knuth, TeX: The Program, volume B of Computers and Typesetting. Addison-Wesley, Reading, 1986.

[KZ86] D. E. Knuth i H. Zapf, AMS Euler — A new typeface for mathematics. Scholary Publishing, 20:131–157, 1986.

[Lam94] L. Lamport, LaTeX: A Document Preparation System. Addison–Wesley, Reading, 1994.

[Lev15] R. Levien, Inconsolata. https://levien.com/type/myfonts/inconsolata.html, 2015.

[Now97] J. Nowacki, Antykwa Toruńska — od początku do końca polska czcionka. *Biuletyn Polskiej Grupy Użytkowników Systemu TeX*, 9:26–27, 1997.

Rozdział 1

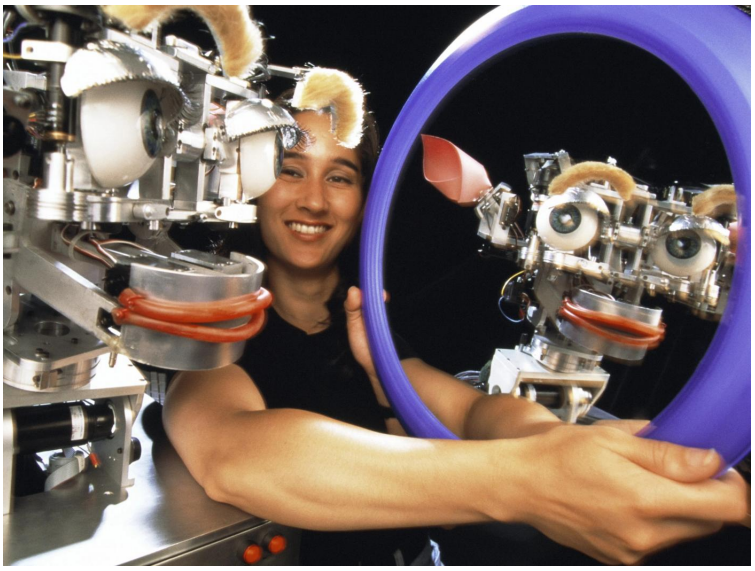
Wstęp

Roboty coraz częściej pełnią funkcje asystentów. Znajdują zastosowanie w interakcjach z dziećmi, osobami starszymi czy w środowiskach edukacyjnych. W tych kontekstach kluczowe jest, aby roboty mogły efektywnie komunikować się z ludźmi. Sukces tej integracji jest ściśle związany z fundamentalną zdolnością maszyn do naturalnego i intuicyjnego nawiązania kontaktu z użytkownikami naśladując bogactwo ludzkiej komunikacji. Odziaływanie na wszystkie zmysły odbiorcy pozwala nawiązać głębszą więź i zwiększyć efektywność interakcji. Dźwięk jest nie tylko nośnikiem informacji, ale również potężnym narzędziem wyrażania emocji i kształtowania percepcji robota przez użytkownika. Zdolność robota do generowania kontekstowo i emocjonalnie adekwatnych sygnałów dźwiękowych jest koniecznym warunkiem stworzenia systemu, który będzie postrzegany jako wiarygodny i kompetentny partner społeczny.

Atutem dźwięku w kontekście nośnika informacji oraz narzędzia do nawiązania relacji jest jego wielokierunkowość [ZF23]. W przeciwieństwie do komunikacji wizualnej, aby nawiązać kontakt przy komunikacji dźwiękowej robot nie musi stale znajdować się w zasięgu wzroku odbiorcy. Ta cecha jest niezwykle istotna kiedy interakcja ma miejsce w dynamicznych, współdzielonych przestrzeniach. W ten sposób maszyna może jawnie zakomunikować swój zamiar, położenie w otoczeniu lub wysłać ostrzeżenie. Co ważne, badania wykazały, że przestrzennie zlokalizowany sygnał dźwiękowy jest wysoce skuteczny w przyciąganiu uwagi wzrokowej odbiorcy [HS05].

Prace robotyczki Cynthii Breazeal z Massachusetts Institute of Technology rzuciły nowe światło na interakcje człowiek-robot przenosząc punkt ciężkości z funkcjonalności na aspekt społeczny. W książce „Designing sociable robots" [Bre02] definiuje ona kluczowe komponenty i pojęcia związane z robotyką społeczną, proponuje ramy projektowe oraz omawia zagadnienia i wyzwania związane z ich realizacją. Swoje doświadczenia zgromadziła przy pracy z robotem Kismet. Zgodnie z wizją Breazeal, roboty miały być tworzone jako „syntetyczne istoty" a nie jedynie zaawansowane narzędzia. Naśladując ludzką kompetencję społeczną, miały być zdolne do nawiązywania relacji z ludźmi w sposób osobisty i intuicyjny. Wyposażenie maszyn w wachlarz niewerbalnych sygnałów społecznych takich jak mimika lub gesty było kluczem do osiągnięcia tego celu.

Wokalizacje niewerbalne stanowiły integralną część repertuaru komunikacyjnego Kismeta. Robot wykorzystywał różnorodne dźwięki do wyrażania emocji, intencji



Rysunek 1.1: Robot Kismet przeglądający się w lustrze [Rob]

i reakcji na otoczenie. Tylko i wyłącznie za pomocą połączenia bodźców wizualnych i akustycznych robot nabiera „sprawczości społecznej" i jest antropomorfizowany w oczach użytkownika, które jest nieosiągalne przy użyciu jednej z tych metod komunikacji. Skuteczność tych sygnałów zależy od ich spójności multimodalnej. Ruch, „wyraz twarzy" oraz dźwięki wydawane przez robota muszą być zsynchronizowane, aby w połączeniu ze sobą tworzyły przekonującą reprezentację emocji lub intencji i wywoływały zamierzoną reakcję odbiorcy. Kiedy technologia zachowuje się w sposób kompetentny społecznie, ludzie podświadomie aktywują swoje ewolucyjnie ukształtowane mechanizmy społeczne, aby z nią interagować.

Dźwięk w interakcji człowiek-robot pełni funkcję nie tylko informacyjną, ale również głęboko psychologiczną, kształtując postawy, emocje i zachowania użytkownika. Odpowiednie zaprojektowanie warstwy akustycznej interfejsu może znacząco wpłynąć na jakość i długoterminowy sukces relacji HRI.

Celem niniejszej pracy jest zaprojektowanie i realizacja autonomicznego systemu sprzętowo-programowego do modyfikacji głosu robota w czasie rzeczywistym, mającego na celu poprawę naturalności komunikacji. Zakres pracy obejmuje opracowanie warstwy sprzętowej opartej na mikrokontrolerze z rodziny STM32H7 oraz implementację algorytmów cyfrowego przetwarzania sygnałów, umożliwiających kształtowanie barwy i charakteru brzmienia maszyny.

Praca została podzielona na sześć rozdziałów. Rozdział drugi przybliża teoretyczne aspekty wykorzystania algorytmów cyfrowego przetwarzania sygnałów w robotyce społecznej, omawiając wpływ modulacji widmowej i efektów dźwiękowych na percepcję robota przez człowieka. W rozdziale trzecim szczegółowo opisano podstawy matematyczne i zasady działania zaimplementowanych algorytmów. Rozdział czwarty przedstawia proces projektowania systemu, dobór platformy sprzętowej opartej na mikrokontrolerze STM32H7 oraz architekturę oprogramowania. Rozdział piąty zawiera wyniki testów weryfikacyjnych wraz z analizą przebiegów sygnałów zarejestrowanych oscyloskopem. Całość zamyka podsumowanie uzyskanych wyników oraz wnioski końcowe.

Rozdział 2

Wykorzystanie algorytmów DSP w interakcji człowiek-robot

Tradycyjna inżynieria dźwięku koncentruje się na parametrach obiektywnych sygnału, takich jak częstotliwość, amplituda czy faza. Jednak w kontekście robotyki społecznej i systemów HRI (ang. *Human-Robot Interaction*), te same parametry nabierają znaczenia subiektywnego, stając się krytycznymi modulatorami poznawczymi [iZo25]. Dźwięk generowany przez robota nie jest jedynie nośnikiem informacji semantycznej, lecz złożonym sygnałem afektywnym, dekodowanym przez ewolucyjnie ukształtowane mechanizmy psychoakustyczne człowieka [LDB22].

Zastosowanie zaawansowanego przetwarzania sygnałów w systemach robotycznych umożliwia przejście od projektowania dźwięków „przyjemnych" do dźwięków „funkcjonalnych", które precyzyjnie sterują stanem emocjonalnym użytkownika oraz jego oceną intencji maszyny [Aud25]. Poniżej omówiono wpływ poszczególnych klas algorytmów na percepcję robota, opierając się na badaniach z zakresu psychofizyki słuchu i neurobiologii [Rep25].

2.1 Modulacja widmowa a percepcja fizyczności

Kluczowym aspektem interakcji z robotem jest ocena jego fizyczności i nastawienia, której ludzki mózg dokonuje poprzez analizę tzw. nachylenia widmowego (ang. *spectral tilt*) [MA17, Ste20]. Zjawisko to bazuje na naturalnej korelacji występującej w przyrodzie: większe rezonatory (np. klatki piersiowe dużych ssaków, długie przewody głosowe) generują dźwięki o silniejszej składowej niskoczęstotliwościowej [VDAP87].

Wykorzystując korekcję barwy (korektor pasmowy), możliwe jest wywołanie u użytkownika audialnej iluzji rozmiaru-wagi (ang. *size-weight illusion*). Badania nad syntezą mowy wykazują, że podbicie pasma w zakresie 80Hz–150Hz sprawia, że głos jest oceniany jako pochodzący od bytu większego, silniejszego i bardziej dominującego [MA17]. Taka konfiguracja spektralna znajduje zastosowanie w scenariuszach związanych z bezpieczeństwem (np. roboty strażnicze), gdzie autorytet maszyny musi zostać zakomunikowany natychmiastowo, jeszcze przed zdekodowaniem treści werbalnej [LZC25].

Z drugiej strony, manipulacja w paśmie 150Hz–400Hz wpływa na percepcję „cie-

pła" (ang. *warmth*) głosu [Jor25]. Badania HRI wskazują na korelację między wzmocnieniem tego pasma a poziomem zaufania użytkownika i jego skłonnością do akceptacji sugestii robota. Zbyt silne tłumienie tego zakresu skutkuje brzmieniem „klinicznym" i bezdusznym, co może negatywnie wpływać na relacje w przypadku robotów opiekuńczych czy terapeutycznych.

2.2 Nieliniowości jako biologiczne sygnały pobudzenia

Algorytmy wprowadzające nieliniowe zniekształcenia sygnału pełnią w HRI rolę wskaźników wysokiego pobudzenia. Mechanizm ten jest głęboko zakorzeniony w neurobiologii. Badania wykazują, że ludzki krzyk posiada unikalną sygnaturę akustyczną, charakteryzującą się wysokim poziomem tzw. szorstkości (ang. *auditory roughness*), wynikającą z szybkich modulacji amplitudy w zakresie 30Hz–150Hz [AFK+15, KK23].

Neuroobrazowanie potwierdza, że dźwięki o wysokiej szorstkości selektywnie i natychmiastowo aktywują ciało migdałowate – strukturę mózgu odpowiedzialną za przetwarzanie strachu i zagrożenia [AFK+15]. Dlatego też, zniekształcenia sygnału w głosie robota nie są interpretowane przez mózg jako błąd techniczny, lecz jako biologiczny sygnał alarmowy o wysokim priorytecie [JC24].

W projektowaniu interakcji pozwala to na funkcjonalne wykorzystanie różnych typów przesterowania:

- **Sonifikacja wysiłku** – Algorytmy miękkiego obcinania (ang. *Soft Clipping*, np. Overdrive) symulują pracę układu pod obciążeniem. Zastosowanie takiej modulacji w momencie wykonywania przez robota trudnych zadań fizycznych lub obliczeniowych może intuicyjnie komunikować, że maszyna używa dużej mocy, co potencjalnie wzbudza empatię użytkownika [Siu24].

- **Sygnalizacja awarii** – Ekstremalne przesterowanie niszczące dynamikę sygnału (np. Fuzz, fala prostokątna) jest skutecznym audialnym sygnifikatorem błędu krytycznego lub „śmierci" procesu cyfrowego. Taki dźwięk wywołuje u użytkownika natychmiastową chęć interwencji naprawczej.

Skuteczność tych sygnałów zależy od kontekstu audialnego. Badania sugerują, że wokalizacje o wysokim pobudzeniu są oceniane jako bardziej emocjonalne, gdy występują na czystym tle, co podkreśla znaczenie kontrastu w projektowaniu komunikatów dźwiękowych [PAA20, VSEB22].

2.3 Zarządzanie ciszą i poczuciem obecności

Istotnym aspektem percepcji „żywotności" robota jest zarządzanie momentami, w których maszyna nie emituje komunikatów werbalnych. Analiza psychoakustyczna wskazuje, że w naturalnym środowisku absolutna cisza (ang. *digital zero*) praktycznie nie występuje [Gra23]. Jej sztuczne wywołanie, na przykład przez agresywną bramkę szumów, może prowadzić do nienaturalnego efektu psychoakustycznego.

Nagły zanik tła akustycznego jest interpretowany przez układ słuchowy jako przerwanie ciągłości strumienia audialnego, co obniża poczucie realizmu i obecności społecznej (ang. *social presence*) robota [TMSM+13, OACP18]. Badania nad percepcją ciszy (zjawisko *one-silence-is-more*) sugerują również, że niewłaściwe bramkowanie przerw w mowie może zaburzać postrzeganie tempa konwersacji i zwiększać obciążenie poznawcze użytkownika [PRC23].

2.4 Przestrzeń i ucieleśnienie

Zastosowanie algorytmów splotowych odwzorowujących akustykę pomieszczeń pozwala na rozwiązanie problemu dysonansu poznawczego, wynikającego z niedopasowania głosu robota do otoczenia [LVK01, OGL23]. Jeśli robot wizualnie przebywa w małym pokoju, a jego głos posiada pogłos dużej hali, następuje załamanie tzw. efektu brzuchomówcy, co prowadzi do wrażenia, że głos nie pochodzi z „ciała" robota, lecz z zewnętrznego źródła.

Modulacja przestrzenna wpływa bezpośrednio na dystans społeczny i emocjonalny [WK25]. Krótkie czasy pogłosu budują intymność i wrażenie bliskości, co jest pożądane u robotów osobistych i terapeutycznych [Mag25]. Z kolei długie pogłosy budują dystans i autorytet, co określa się mianem efektu „Głosu Boga", użytecznego w robotach pełniących funkcje ceremonialne lub informacyjne w dużych przestrzeniach publicznych [HKM+20].

Uzupełnieniem modulacji przestrzennej są efekty czasowe, takie jak Chorus. Poprzez mikromodulacje czasu i wysokości dźwięku, pozwalają one na przełamanie sterylności syntezy mowy. Subtelne, losowe odstrajanie dodaje sygnałowi organicznej nieprzewidywalności, co pomaga w uniknięciu efektu „doliny niesamowitości" (ang. *uncanny valley*), często występującego przy idealnie czystych, pozbawionych fluktuacji głosach syntezatorów [SK20, MSL+24].

2.5 Wnioski projektowe

Efektywna komunikacja w relacji człowiek-robot wymaga wyjścia poza proste odtwarzanie komunikatów i potraktowania toru audio jako elastycznego narzędzia kreacji wizerunku. Aby wyposażyć maszynę w zdolność do wielowymiarowej ekspresji, konieczne jest zaimplementowanie szerokiego wachlarza algorytmów, które czerpią z odmiennych technik przetwarzania sygnału. Fundamentem staje się precyzyjna kontrola nad barwą poprzez korekcję widmową, która pozwala definiować postrzeganą fizyczność i nastawienie robota. Jednak dopiero połączenie jej z efektami nieliniowymi otwiera drogę do sygnalizowania bardziej złożonych stanów afektywnych, takich jak wysiłek czy pilność, poprzez kontrolowaną „szorstkość" brzmienia. Dopełnieniem tej palety są techniki przestrzenne i modulacyjne, które usuwają cyfrową sterylność, osadzając głos maszyny w naturalnym kontekście akustycznym. Zestawienie tak zróżnicowanych efektów w jednym systemie tworzy uniwersalną platformę, która umożliwia swobodne łączenie poszczególnych bloków przetwarzania, dając projektantom interakcji możliwość dynamicznego dopasowywania charakteru robota do bieżącego kontekstu społecznego.

Rozdział 3

Przetwarzanie sygnałów o częstotliwościach akustycznych

Sygnał jest abstrakcyjnym modelem dowolnej mierzalnej wielkości zmieniającej się w czasie. Jednym z parametrów sygnału jest częstotliwość, czyli liczba powtarzających się cykli sygnału w jednostce czasu [Wikh]. Pośród sygnałów możemy wyróżnić sygnał akustyczny, który jest falą toteż zaburzeniem gęstości i ciśnienia rozchodzącym się w ośrodku w postaci fali podłużnej, któremu towarzyszą drgania ośrodka. Fale w zakresie słyszalnym przez ludzi (od 20Hz do 20KHz) nazywamy dźwiękami [Wikd].

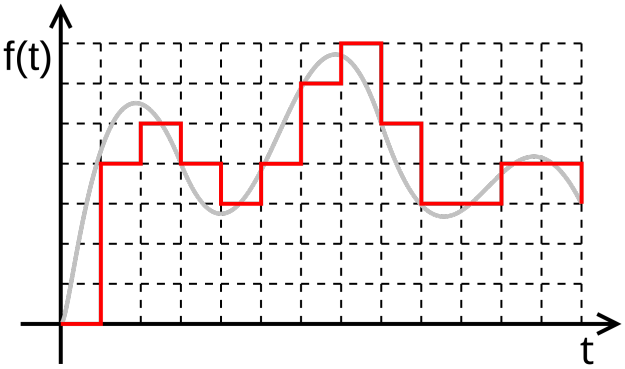
Dźwięki są sygnałami analogowymi co oznacza, że mogą przyjmować dowolną wartość z ciągłego przedziału [Wiki]. Aby przetwarzać takie sygnały musimy przekształcić je do postaci sygnału cyfrowego którego dziedzina i zbiór wartości są dyskretne (zobacz rysunek 3.1) [Wikj].

Etap przekształcania sygnału analogowego na cyfrowy realizowany jest za pomocą przetwornika analogowo-cyfrowego, który poddaje sygnał próbkowaniu, kwantyzacji oraz kodowaniu [Wikg]. Próbkowanie to pomiar sygnału w określonych chwilach [HiF]. Aby przeprowadzić ten proces w sposób właściwy należy uwzględnić twierdzenie Kotielnikowa-Shannona [Sha49]:

Twierdzenie 1 *Jeśli funkcja x nie zawiera częstotliwości wyższych niż W herców, jest ona całkowicie zdeterminowana przez podanie jej rzędnych (wartości) w serii punktów oddalonych od siebie o $1/(2W)$ sekundy.*

Aby możliwa była idealna rekonstrukcja sygnału, częstotliwość próbkowania musi być co najmniej dwukrotnie większa niż maksymalna częstotliwość występująca w tym sygnale. W przypadku kiedy warunek jest niespełniony, podczas próby odtworzenia naszego sygnału możemy napotkać zjawisko zwane aliasingiem [Zie14]. Jest to nieodwracalne zniekształcenie objawiające się obecnością składowych o błędnych częstotliwościach (aliasach) w wynikowym sygnale [Wika]. Standardową częstotliwością próbkowania spełniającą Twierdzenie 1 dla fali dźwiękowej, zgodną z formatem CD-audio, jest 44100Hz [Ody].

Następnym krokiem przetwarzania sygnału analogowego na cyfrowy jest kwantyzacja. Polega ona na przyporządkowaniu bieżącym wartościom sygnału, wartości dyskretnych z określonego zbioru. Wartość sygnału jest zawsze zaokrąglana,



Rysunek 3.1: Sygnał analogowy (szary) i przykładowe przekształcenie na sygnał cyfrowy (czerwony) [Wikj]

przez co podobnym lecz różnym wartościom sygnału mogą zostać przydzielone te same wartości dyskretne. Dokładność kwantyzacji sygnału jest określana liczbą bitów, na których jest on zapisywany [Aud]. Różnica między rzeczywistą wartością próbki a wartością przypisaną jej po zaokrągleniu nazywana jest szumem kwantyzacji. Przykładowo, format CD-audio gwarantuje zapis na 16 bitach, co oferuje $2^{16} = 65536$ niezależnych wartości, które mogą być przypisane do przetwarzanego sygnału. Im wyższa rozdzielczość bitowa, tym błąd kwantyzacji jest mniejszy, co przekłada się na wyższą wierność dźwięku i lepszy stosunek sygnału do szumu.

Ostatnim etapem procesu przetwarzania sygnału analogowego na cyfrowy jest kodowanie. Polega ono na przypisaniu unikalnej wartości binarnej, inaczej zwanej słowem kodowym, każdemu z dyskretnych poziomów amplitudy wyznaczononych w procesie kwantyzacji. Dominującym standardem kodowania jest modulacja kodowo-impulsowa PCM, (ang. Pulse-Code Modulation) [Wikf].

Produktem końcowym opisanych wyżej procesów, będących etapami konwersji sygnału analogowego na cyfrowy, jest strumień danych PCM. Jest to ciągła sekwencja binarnych słów kodowych reprezentujących amplitudę sygnału w kolejnych chwilach próbkowania. Ten surowy strumień jest fundamentalną cyfrową reprezentacją fali dźwiękowej, która może zostać poddana dalszym operacjom przez algorytmy cyfrowego przetwarzania sygnałów.

3.1 Biblioteki i narzędzia do przetwarzania sygnałów

Przetwarzanie sygnałów w czasie rzeczywistym wymaga dużej mocy obliczeniowej. Realizacja tego zadania na mikrokontrolerze jest szczególnie skomplikowana ze względu na ograniczone zasoby platformy. Osiągnięcie wydajności pozwalającej na przetworzenie każdej próbki wymaga użycia zoptymalizowanych algorytmów przetwarzania sygnałów. W tym celu wykorzystano specjalistyczną bibliotekę ARM CMSIS-DSP [ARMb].

Standard CMSIS (ang. *Common Microcontroller Software Interface Standard*) [ARMa] to warstwa abstrakcji sprzętowej dla mikrokontrolerów ARM. Standaryzacja ułatwia przenośność oprogramowania między różnymi kontrolerami bazującymi na rdzeniach Cortex. Spośród dostępnych komponentów możemy wyróż-

nić bibliotekę CMSIS-DSP, która oferuje zoptymalizowane funkcje matematyczne i algorytmy przetwarzania sygnałów. Została ona zaprojektowana w taki sposób, aby w pełni wykorzystywać zaawansowane możliwości sprzętowe nowoczesnych rdzeni ARM Cortex-M. Najważniejsze cechy optymalizacji to między innymi wykorzystanie instrukcji SIMD (ang. *Single Instruction Multiple Data*) do równoległego przetwarzania wielu danych w jednym cyklu zegara lub wykorzystanie FPU (ang. *Floating Point Unit*) do przyspieszenia operacji zmiennoprzecinkowych na mikrokontrolerach z rdzeniami Cortex-M4 i nowszymi.

3.2 Korektor pasmowy

Korektor pasmowy (ang. *Equalizer*) to układ selektywny częstotliwościowo, umożliwiający niezależne kształtowanie widma sygnału poprzez podbijanie lub tłumienie wybranych pasm [Wikc]. W przeciwieństwie do filtrów o stałej charakterystyce, korektor pasmowy pozwala na precyzyjną kontrolę balansu tonalnego poprzez regulację częstotliwości środkowej f_c, szerokości pasma (Q) oraz wzmocnienia (G) dla każdego z pasm.

Standardową architekturą korektora pasmowego jest połączenie kaskadowe (szeregowe) niezależnych filtrów. W przypadku korektora trójpasmowego tor przetwarzania składa się z trzech filtrów parametrycznych, z których każdy modyfikuje określone pasmo częstotliwości: niskie (ang. *Low Shelf*), średnie (ang. *Middle Shelf*) i wysokie tony (ang. *High Shelf*). Matematycznie, transmitancja wypadkowa takiego systemu $H_{total}(z)$ jest iloczynem transmitancji $H_L(z), H_M(z), H_H(z)$ poszczególnych sekcji. Taka konfiguracja pozwala na niezależne modyfikowanie każdego pasma, a wypadkowa charakterystyka częstotliwościowa jest sumą (w skali decybelowej) charakterystyk składowych. Do grupy filtrów parametrzycznych zaliczamy filtry półkowe i filtry szczytowe.

3.2.1 Projektowanie charakterystyk filtrów

Aby zrealizować cyfrową korekcję dźwięku, konieczne jest przetłumaczenie parametrów muzycznych, zrozumiałych dla użytkownika (częstotliwość f_c, wzmocnienie G, szerokość pasma Q), na współczynniki równania różnicowego (a_i, b_i). Proces ten odbywa się w czasie rzeczywistym. Gdy użytkownik modyfikuje parametry korektora, algorytm przelicza nowe wartości współczynników, wykorzystując parametry pomocnicze wynikające z transformaty biliniowej. Dla wszystkich typów filtrów definiuje się wzmocnienie w skali liniowej $V_0 = 10^{G/20}$ oraz znormalizowany parametr częstotliwościowy $K = \tan\left(\pi \frac{f_c}{f_s}\right)$.

Filtry półkowe

Filtry półkowe (ang. *Shelving Filters*) służą do modyfikacji skrajnych zakresów pasma akustycznego. Filtr dolnopółkowy przetwarza częstotliwości poniżej częstotliwości granicznej f_c, natomiast górnopółkowy powyżej tej granicy. Ze względu na specyfikę transformacji logarytmicznej amplitudy, wzory na współczynniki różnią się w zależności od tego, czy sygnał jest wzmacniany ($G \geqslant 0$), czy tłumiony

Tabela 3.1: Współczynniki filtrów półkowych (Low/High Shelf) [Zöl11]

Typ i Warunek	b_0	b_1	b_2	a_1	a_2
Low Shelf ($G \geqslant 0$)	$\frac{1+\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
Low Shelf ($G < 0$)	$\frac{V_0(1+\sqrt{2}K+K^2)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2V_0(K^2-1)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{V_0(1-\sqrt{2}K+K^2)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2-V_0)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$
High Shelf ($G \geqslant 0$)	$\frac{V_0+\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-V_0)}{1+\sqrt{2}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
High Shelf ($G < 0$)	$\frac{V_0(1+\sqrt{2}K+K^2)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2V_0(K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{V_0(1-\sqrt{2}K+K^2)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2V_0}K+V_0K^2}$

Tabela 3.2: Współczynniki filtru szczytowego (Peak) [Zöl11]

Warunek	b_0	b_1	b_2	a_1	a_2
Boost ($G \geqslant 0$)	$\frac{1+\frac{V_0}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q}K+K^2}$	$\frac{1-\frac{V_0}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q}K+K^2}$	$\frac{1-\frac{1}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$
Cut ($G < 0$)	$\frac{1+\frac{1}{Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{1-\frac{1}{Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{1-\frac{1}{V_0Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$

($G < 0$). Kompletny zestaw wzorów na współczynniki filtrów półkowych, opracowany na podstawie [Zöl11], zestawiono w tabeli 3.1.

Filtr szczytowy

Filtr szczytowy (ang. *Peaking Filter*), który pozwala na interwencję w konkretnym wycinku widma sygnału, służy do modyfikacji środkowego pasma. Oprócz wzmocnienia V i częstotliwości K, filtr ten wykorzystuje parametr szerokości pasma Q. Wzory na współczynniki filtru szczytowego zestawiono w tabeli 3.2. Przepływ sygnału przez kaskadę trzech filtrów BiQuad przedstawiono w formie pseudokodu na rysunku 3.2.

3.2.2 Filtracja IIR drugiego rzędu

Korektor pasmowegy bazuje na zastosowaniu filtru IIR (ang. *Infinite Impulse Response*) drugiego rzędu, znanego również jako sekcja bikwadratowa (ang. *BiQuad*) [BJ05]. Nazwa wynika z faktu, że jego transmitancja w dziedzinie $\mathbb{Z}$ jest ilorazem dwóch wielomianów kwadratowych

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}, \tag{3.1}$$

Współczynniki licznika (b_0, b_1, b_2) określają zera transmitancji, natomiast współczynniki mianownika (a_1, a_2) determinują położenie biegunów, co bezpośrednio wpływa na charakterystykę częstotliwościową i stabilność filtru.

Funkcja: `przetwarzaj_korektor(x)`

// Przetwarzanie kaskadowe (szeregowe)
// Wyjście jednego filtru jest wejściem następnego

// 1. Pasmo niskie (Low Shelf)
$y_{low} \leftarrow \text{biquad\_low}(x)$

// 2. Pasmo środkowe (Peaking)
$y_{mid} \leftarrow \text{biquad\_mid}(y_{low})$

// 3. Pasmo wysokie (High Shelf)
$y_{high} \leftarrow \text{biquad\_high}(y_{mid})$

Zwróć y_{high}

Rysunek 3.2: Przepływ sygnału w korektorze trójpasmowym

Funkcja: `przetwarzaj_biquad(x,` b_0, b_1, b_2, a_1, a_2`)`

// 1. Obliczenie bieżącej próbki wyjściowej
// Wykorzystanie próbki wejściowej oraz pierwszego stanu wewnętrznego
$y \leftarrow b_0 \cdot x + s_1$

// 2. Aktualizacja stanów wewnętrznych filtru
// Przygotowanie wartości dla następnej próbki
$s_1 \leftarrow b_1 \cdot x - a_1 \cdot y + s_2$
$s_2 \leftarrow b_2 \cdot x - a_2 \cdot y$

Zwróć y

Rysunek 3.3: Ogólny algorytm filtru BiQuad

W dziedzinie czasu dyskretnego operację filtracji opisuje równanie różnicowe

$$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a_1y(n-1) - a_2y(n-2). \tag{3.2}$$

gdzie $x(n)$ to próbka wejściowa, a $y(n)$ próbka wyjściowa filtru. Implementację filtru BiQuad w formie pseudokodu przedstawiono na rysunku 3.3.

3.3 Algorytmy przetwarzania dynamiki

Przetwarzanie dynamiki odnosi się do algorytmów, w których wzmocnienie układu jest sterowane automatycznie w zależności od poziomu sygnału wejściowego. Do tej grupy należą m.in. kompresory, limitery, ekspandery oraz bramki szumów.

Ogólna zasada działania procesora dynamiki polega na podziale toru sygnałowego na dwie ścieżki: tor audio, którym przesyłany jest sygnał poddawany obróbce,

oraz tor sterujący, w którym odbywa się analiza poziomu sygnału i wyznaczanie współczynnika wzmocnienia. Sygnał wyjściowy $y(n)$ jest wynikiem przemnożenia sygnału wejściowego $x(n)$ (ewentualnie opóźnionego o D próbek w celu kompensacji czasu reakcji układu) przez zmienny współczynnik wzmocnienia $g(n)$

$$y(n) = g(n)x(n - D). \tag{3.3}$$

Procedura wyznaczania współczynnika $g(n)$ przebiega w trzech etapach

1. **Pomiar poziomu sygnału** – Sygnał wejściowy jest analizowany w celu określenia jego poziomu głośności. Może to być realizowane za pomocą różnych metod wykorzystujących wartość średnią, wartość skuteczną (ang. *RMS*) czy detektor szczytowy.

2. **Generowanie charakterystyki wzmocnienia** – Charakterystyka wzmocnienia określa, jak układ reaguje na różne poziomy sygnału. Na podstawie zmierzonego poziomu sygnału oraz ustawionych parametrów charakterystycznych dla danego procesora wyznaczany jest współczynnik wzmocnienia $g(n)$.

3. **Wygładzanie** – Bezpośrednie aplikowanie wzmocnienia może prowadzić do gwałtownych zmian amplitudy sygnału. Dlatego zmiany wzmocnienia są filtrowane dolnoprzepustowo z wykorzystaniem dwóch stałych czasowych t_{att} określającej szybkość reakcji układu na przekroczenie progu oraz t_{rel} definiującej czas powrotu układu do stanu neutralnego.

3.3.1 Bramka szumów

Bramka szumów (ang. *Noise Gate*) jest procesorem dynamiki, którego zadaniem jest całkowite lub częściowe tłumienie sygnału, gdy jego poziom spada poniżej określonego progu. Głównym celem bramki szumów jest eliminacja niepożądanych dźwięków tła, takich jak szumy, trzaski czy inne zakłócenia, które mogą być słyszalne w cichych fragmentach sygnału audio.

Do modyfikacji dostępne są trzy parametry. Parametr L_{th} określa poziom progowy, poniżej którego bramka zaczyna tłumić sygnał. Parametr t_{att} definiuje szybkość otwierania bramki po przekroczeniu progu – krótki czas ataku zapewnia natychmiastową reakcję, długi wprowadza płynne narastanie. Parametr t_{rel} kontroluje szybkość zamykania bramki po spadku sygnału poniżej progu – zbyt krótki czas może powodować odcinanie naturalnego wybrzmiewania dźwięku.

Charakterystykę statyczną bramki można opisać jako funkcję skokową. Jeśli poziom sygnału sterującego jest wyższy od progu L_{th}, bramka pozostaje otwarta (wzmocnienie $g = 1$). Gdy poziom spada poniżej progu, bramka zamyka się (wzmocnienie $g = 0$), tłumiąc sygnał wyjściowy.

Aby sterowanie przebiegało w sposób przewidywalny, bramka nie reaguje na chwilowe wartości próbek $x(n)$, lecz na ich obwiednię (ang. *envelope*). W tym celu wykorzystuje się cyfrowy detektor szczytowy pierwszego rzędu z asymetrycznymi czasami reakcji. Pozwala on na natychmiastowe wykrycie początku dźwięku (faza ataku) oraz powolne opadanie sygnału sterującego przy wybrzmiewaniu nuty (faza zwolnienia).

Algorytm ekstrakcji obwiedni $e(n)$ opisuje równanie rekurencyjne

$$e(n) = \begin{cases} \beta_{att} e(n-1) + (1-\beta_{att})|x(n)|, & \text{gdy } |x(n)| > e(n-1) \\ \beta_{rel} e(n-1) + (1-\beta_{rel})|x(n)|, & \text{gdy } |x(n)| \leqslant e(n-1), \end{cases} \tag{3.4}$$

gdzie $|x(n)|$ to wartość bezwzględna próbki wejściowej, a β_{att} i β_{rel} to współczynniki wygładzania zależne od zadanych stałych czasowych. Tak wyznaczony sygnał $e(n)$ stanowi sygnał sterujący dla układu progowego opisanego za pomocą maszyny stanów

$$g_{target}(n) = \begin{cases} 1.0, & \text{gdy } e(n) > L_{th} + \Delta, \\ 0.0, & \text{gdy } e(n) < L_{th} - \Delta, \\ g_{target}(n-1), & \text{w przeciwnym razie (strefa martwa).} \end{cases} \tag{3.5}$$

Krytycznym problemem bramek szumów jest oscylacja sygnału wokół wartości progowej, powodująca gwałtowne, naprzemienne otwieranie i zamykanie bramki. Standardową metodą eliminacji tego efektu jest zastosowanie logiki z histerezą. Układ ten definiuje dwa progi przełączania przesunięte względem siebie o margines Δ: próg otwarcia $L_{open} = L_{th} + \Delta$ oraz próg zamknięcia $L_{close} = L_{th} - \Delta$. Dzięki temu sygnał musi przekroczyć górną wartość graniczną, aby otworzyć bramkę, oraz spaść poniżej dolnej granicy, aby ją zamknąć.

Aby uniknąć trzasków przy przełączaniu, sygnał sterujący poddawany jest finalnemu wygładzaniu filtrem dolnoprzepustowym o współczynniku λ. Kompletny algorytm przetwarzania bramki szumów w formie pseudokodu przedstawiono na rysunku 3.4.

3.4 Nieliniowe zniekształcenia sygnału

Algorytmy przesterowania sygnału są wykorzystują technikę cyfrowego kształtowania fali (ang. *waveshaping*) [Ale]. Jest to nieliniowy proces polegający na przekształceniu amplitudy każdej próbki wejściowej $x(n)$ na wartość wyjściową $y(n)$ za pomocą funkcji charakterystyki przenoszenia (ang. *transfer function*). Proces ten prowadzi do zmiany kształtu przebiegu sygnału w dziedzinie czasu (np. spłaszczenia wierzchołków sinusoidy), co skutkuje wzbogaceniem widma o nowe składowe harmoniczne. O finalnej barwie i charakterystyce efektu decyduje rodzaj i amplituda tych harmonicznych.

Zaimplementowane w pracy efekty przesterowania sygnału („Overdrive", „Distortion", „Fuzz") współdzielą wspólną architekturę przetwarzania sygnału oraz parametry sterujące. Do modyfikacji dostępne są parametry G_{in} określający wzmocnienie wejściowe (intensywność przesterowania) oraz α balansuje proporcje między sygnałem czystym a efektem. Różnice w brzmieniu poszczególnych efektów wynikają z zastosowania odmiennych funkcji nieliniowych $f_{type}(\cdot)$. Ogólne równanie różnicowe dla toru przetwarzania przyjmuje postać

$$y(n) = G_{out} f_{type}(G_{in} x(n))\alpha + x(n)(1-\alpha), \tag{3.6}$$

Funkcja: `przetwarzaj_gate(x, $L_{th}$, $\Delta$, $\lambda$)`

// 1. Detekcja amplitudy (wartość bezwzględna)
$x_{abs} \leftarrow |x|$

// 2. Ekstrakcja obwiedni (Envelope Follower)
if $x_{abs} > e_{prev}$ **then**
    $e_{curr} \leftarrow \beta_{att} \cdot e_{prev} + (1 - \beta_{att}) \cdot x_{abs}$
else
    $e_{curr} \leftarrow \beta_{rel} \cdot e_{prev} + (1 - \beta_{rel}) \cdot x_{abs}$
end if

// 3. Logika progowa z histerezą
if $e_{curr} > L_{th} + \Delta$ **then**
    $g_{target} \leftarrow 1.0$
else if $e_{curr} < L_{th} - \Delta$ **then**
    $g_{target} \leftarrow 0.0$
else
    $g_{target} \leftarrow g_{prev}$
end if

// 4. Wygładzanie przejść i aplikacja wzmocnienia
$g_{smooth} \leftarrow \lambda \cdot g_{smooth\_prev} + (1 - \lambda) \cdot g_{target}$
$y \leftarrow x \cdot g_{smooth}$

Zwróć y

Rysunek 3.4: Przetwarzanie próbki w bramce szumów

gdzie $x(n)$ to znormalizowany sygnał wejściowy, G_{in} to wzmocnienie wejściowe (parametr „Drive") determinujące punkt pracy na krzywej nieliniowej, G_{out} to wzmocnienie wyjściowe zastosowane w celu kompensacji głośności, α to parametr mieszania („Mix") oraz $f_{type}(\cdot)$ to funkcja kształtująca charakterystyczna dla wybranego trybu („Overdrive", „Distortion" lub „Fuzz").

3.4.1 Overdrive

Efekt Overdrive modeluje nieliniową odpowiedź analogowych wzmacniaczy, w których przy dużym wzmocnieniu sygnał ulega zjawisku nasycenia i obcinane są wierzchołki fal. Zgodnie z definicją sformułowaną przez Zölzera w „DAFX: Digital Audio Effects" [Zöl11], efekt Overdrive charakteryzuje się płynnym przejściem między liniowym a nieliniowym zakresem pracy urządzenia, tym samym zachowuje dynamikę gry, wprowadzając „ciepłe" brzmienie wynikające z generowania nieparzystych harmonicznych. Głównym parametrem sterującym jest wzmocnienie wejściowe G_{in}, które determinuje punkt pracy na krzywej nieliniowej oraz intensywność generowanych harmonicznych.

W algorytmie wzmocnienia Overdrive zastosowano model wykorzystujący formułę Schetzena [Sch80] oznaczony symbolem $f_{over}(x)$, który realizuje symetryczne, miękkie obcinanie (ang. *soft clipping*). Charakterystyka wzmocnienia jest wyrażona za pomocą wzoru

$$f_{over}(x) = \begin{cases} 2x & \text{dla } |x| < \frac{1}{3} \\ \frac{3-(2-3|x|)^2}{3} & \text{dla } \frac{1}{3} \leqslant |x| < \frac{2}{3} \\ sgn(x) & \text{dla } |x| \geqslant \frac{2}{3} \end{cases}. \tag{3.7}$$

Model ten dzieli zakres amplitudy sygnału na trzy strefy. W obszarze liniowym ($|x| < \frac{1}{3}$) sygnał jest jedynie wzmacniany ze współczynnikiem 2. W strefie kompresji ($\frac{1}{3} \leqslant |x| < \frac{2}{3}$) następuje łagodne zaokrąglanie wierzchołków fali, co powoduje występowanie harmonicznych przy zachowaniu gładkiego brzmienia. Powyżej progu $\frac{2}{3}$ następuje twarde nasycenie do wartości ± 1.

Dzięki symetrii funkcji przekształcenia, widmo sygnału generuje harmoniczne tylko nieparzystego rzędu, przy czym najbardziej wyraźne są 3. i 5. harmoniczna. Harmoniczne pojawiają się stopniowo w miarę wzrostu poziomu sygnału powyżej progu $\frac{1}{3}$ i stabilizują się po osiągnięciu twardego nasycenia. Implementację efektu Overdrive w formie pseudokodu przedstawiono na rysunku 3.5.

3.4.2 Fuzz

Efekt Fuzz stanowi najbardziej radykalną formę przesterowania sygnału. Zgodnie z klasyfikacją przedstawioną w literaturze [Zöl11] charakteryzuje się on całkowicie nieliniową pracą układu, prowadzącą do drastycznej zmiany kształtu fali. Zastosowanie różnych progów obcinania dla dodatniej i ujemnej połówki sygnału wprowadza silną asymetrię generując parzyste harmoniczne, co nadaje brzmieniu specyficzną „szorstką" fakturę. Dodatkowo efekt Fuzz posiada parametr asymetrii A, który kontroluje stopień nierówności między obcinaniem górnej i dolnej połówki fali, wpływając bezpośrednio na charakter generowanych harmonicznych.

Funkcja: przetwarzaj_overdrive(x, G_{in}, G_{out}, α)

// 1. Wzmocnienie wejściowe
$x_{gain} \leftarrow x \cdot G_{in}$

// 2. Nieliniowe kształtowanie fali (Schetzen Formula)
if $|x_{gain}| < \frac{1}{3}$ **then**
  $y_{wet} \leftarrow 2 \cdot x_{gain}$
else if $1/3 \leqslant |x_{gain}| < \frac{2}{3}$ **then**
  $y_{wet} \leftarrow \text{sgn}(x_{gain}) \cdot \frac{3-(2-3\cdot|x_{gain}|)^2}{3}$
else
  $y_{wet} \leftarrow \text{sgn}(x_{gain})$
end if

// 3. Kompensacja wyjściowa i mieszanie
$y_{comp} \leftarrow y_{wet} \cdot G_{out}$
$y \leftarrow y_{comp} \cdot \alpha + x \cdot (1 - \alpha)$

Zwróć y

Rysunek 3.5: Przetwarzanie próbki w efekcie Overdrive

Zasadniczym elementem algorytmu w efekcie Fuzz jest funkcja transmitancji $f_{fuzz}(x)$, zdefiniowana jako funkcja o trzech zakresach działania. Granice tych zakresów wyznaczają progi: stały próg dodatni $T_+ = 0.7$ oraz zmienny próg ujemny $T_- = 0.4(1 - 0.5A)$, gdzie A to parametr asymetrii przybierający wartości z zakresu $[0, 1]$. Zwiększanie wartości parametru A powoduje obniżenie wartości bezwzględnej progu T_-, co pogłębia asymetrię przebiegu. Funkcja kształtująca przyjmuje postać

$$f_{fuzz}(x) = \begin{cases} T_+ + 0.3\tanh(2(x - T_+)) & \text{dla } x > T_+ \\ x(1 + 0.2x^2) & \text{dla } -T_- \leqslant x \leqslant T_+ \\ -T_- & \text{dla } x < -T_- \end{cases}. \tag{3.8}$$

Trzy strefy działania efektu Fuzz to:

1. **Saturacja dodatnia** ($x > T_+$) - Ciągłość pochodnej w punkcie T_+ jest zapewniona dzięki użyciu funkcji tangensa hiperbolicznego. Powoduje on również łagodne, asymptotyczne dążenie sygnału do wartości maksymalnej.

2. **Obszar quasi-liniowy** ($-T_- \leqslant x \leqslant T_+$) - W strefie środkowej zastosowano wielomian trzeciego stopnia. Człon $0.2x^3$ wprowadza nieliniowość (delikatną kompresję) jeszcze przed osiągnięciem progów obcinania, co wzbogaca sygnał przy niższych amplitudach.

3. **Ograniczanie ujemne** ($x < -T_-$) - Dla połówki ujemnej zastosowano sztywne ograniczenie (ang. *hard clipping*). Nagłe „ścięcie" wierzchołka fali wprowadza nieciągłość pochodnej, generując szerokie spektrum harmonicznych wysokiego rzędu.

Funkcja: `przetwarzaj_fuzz(x, G_in, G_out, α, A)`

// 1. Wzmocnienie wejściowe
$x_{gain} \leftarrow x \cdot G_{in}$

// 2. Nieliniowe kształtowanie fali
if $x_{gain} > T_+$ **then**
    // Strefa dodatnia: miękkie nasycenie (tanh)
    $y_{wet} \leftarrow T_+ + 0.3 \cdot \tanh(2 \cdot (x_{gain} - T_+))$
else if $x_{gain} < -T_-$ **then**
    // Strefa ujemna: twarde obcięcie
    $y_{wet} \leftarrow -T_-$
else
    // Strefa środkowa: zniekształcenie wielomianowe
    $y_{wet} \leftarrow x_{gain} \cdot (1.0 + 0.2 \cdot x_{gain}^2)$
end if

// 3. Kompensacja i mieszanie
$y_{comp} \leftarrow y_{wet} \cdot G_{out}$
$y \leftarrow y_{comp} \cdot \alpha + x \cdot (1 - \alpha)$

Zwróć y

Rysunek 3.6: Przetwarzanie próbki w efekcie Fuzz

Algorytm przetwarzania efektu Fuzz w formie pseudokodu przedstawiono na rysunku 3.6.

3.4.3 Distortion

Efekt Distortion charakteryzuje się dużą szybkością saturacji sygnału. Funkcja wykładnicza dąży do wartości granicznych (± 1) znacznie gwałtowniej niż wielomianowy model Schetzena, co skutkuje silniejszą kompresją dynamiki. Nie zaobserwujemy jednak twardego obcinania wierzchołków fali ze względu na ciągłość pochodnej w całej dziedzinie. Podobnie jak w efekcie Overdrive, głównym parametrem sterującym jest wzmocnienie wejściowe G_{in}, którego wartość znacząco wpływa na charakter brzmienia - od delikatnej koloryzacji przy niskich wartościach po agresywne, nasycone zniekształcenie przy wysokich. Modulację sygnału efektu Distortion modeluje funkcja $f_{dist}(x)$ opisana wzorem

$$f_{dist}(x) = \text{sgn}(x)(1 - e^{-|x|}), \tag{3.9}$$

gdzie $\text{sgn}(x)$ oznacza funkcję znaku (signum), a e to podstawa logarytmu naturalnego.

Analiza szeregu Taylora dla powyższej funkcji ujawnia obecność parzystych potęg w rozwinięciu. Oznacza to, że mimo nieparzystej symetrii funkcji globalnej ($f(-x) = -f(x)$), w lokalnym zachowaniu dla małych sygnałów pojawia się składnik kwadratowy. W rezultacie widmo sygnału wyjściowego wzbogacane jest zarówno o harmoniczne nieparzyste (dominujące), jak i parzyste.

Funkcja: `przetwarzaj_distortion(x, G_in, G_out, α)`

// 1. Wzmocnienie wejściowe
$x_{gain} \leftarrow x \cdot G_{in}$

// 2. Nieliniowe kształtowanie fali (Model wykładniczy)
$y_{wet} \leftarrow \text{sgn}(x_{gain}) \cdot (1.0 - e^{-|x_{gain}|})$

// 3. Kompensacja wyjściowa i mieszanie
$y_{comp} \leftarrow y_{wet} \cdot G_{out}$
$y \leftarrow y_{comp} \cdot \alpha + x \cdot (1 - \alpha)$

Zwróć y

Rysunek 3.7: Przetwarzanie próbki w efekcie Distortion

Brzmienie sygnału przetworzonego zależy ściśle od parametru wzmocnienia wejściowego G_{in}. Dla małych wartości ($G_{in} < 5$) układ wprowadza delikatną koloryzację, natomiast dla dużych wartości ($G_{in} > 50$) sygnał szybko osiąga nasycenie, upodabniając się do fali prostokątnej, co generuje silne harmoniczne wysokiego rzędu. Implementację algorytmu Distortion w formie pseudokodu przedstawiono na rysunku 3.7.

3.5 Efekty oparte na liniach opóźniających

Linia opóźniająca (ang. *delay line*) [Wike] przenosi sygnał wejściowy $x(n)$ na wyjście $y(n)$ z przesunięciem w czasie o D próbek, nie zmieniając jego amplitudy ani kształtu, zgodnie ze wzorem

$$y(n) = x(n - D). \tag{3.10}$$

W dziedzinie transformaty $\mathcal{Z}$, idealna linia opóźniająca charakteryzuje się transmitancją $H(z) = z^{-D}$.

Efekty akustyczne uzyskane za pomocą linii opóźniającej zależą od rodzaju i wartości parametru opóźnienia D, a także od konfiguracji pracy linii (sprzężenie zwrotne lub proste sumowanie). Parametr ten może być wartością stałą lub dynamicznie zmienną w czasie.

3.5.1 Delay

Efekt Delay modeluje zjawisko wielokrotnego odbicia fali akustycznej [Zöl11]. Użycie pętli sprzężenia zwrotnego tłumaczy nieskończony charakter odpowiedzi impulsowej tego algorytmu. Struktura ta pozwala na recyrkulację sygnału, co skutkuje powstawaniem serii zanikających powtórzeń oryginalnego dźwięku.

Do modyfikacji dostępne są trzy parametry. Parametr D określa czasowy odstęp między kolejnymi powtórzeniami. Parametr g definiuje współczynnik sprzężenia zwrotnego, który decyduje o liczbie słyszalnych powtórzeń – wyższa wartość

Funkcja: `przetwarzaj_delay(x, g, α, D)`

// 1. Pobranie próbki z przeszłości
$v_{past} \leftarrow w(n - D)$

// 2. Pętla sprzężenia zwrotnego
$w(n) \leftarrow x + v_{past} \cdot g$

// 3. Formowanie sygnału wyjściowego
$y \leftarrow x \cdot (1 - \alpha) + v_{past} \cdot \alpha$

Zwróć y

Rysunek 3.8: Przetwarzanie próbki w efekcie Delay

powoduje dłuższe zanikanie echa. Parametr α ustala proporcje między sygnałem oryginalnym a opóźnionym, kontrolując głośność efektu w miksie. Dla stabilności układu konieczne jest zachowanie warunku $|g| < 1$.

Model matematyczny efektu Delay opisuje układ równań różnicowych. Pierwsze równanie definiuje sygnał $w(n)$, będący sumą bieżącej próbki wejściowej oraz próbki opóźnionej, poddanej tłumieniu

$$w(n) = x(n) + gw(n - D), \tag{3.11}$$

gdzie D to stała wartość opóźnienia w próbkach, a g to współczynnik wzmocnienia w pętli sprzężenia zwrotnego.

Warunkiem stabilności takiego układu jest, aby moduł wzmocnienia pętli był mniejszy od jedności ($|g| < 1$). W przeciwnym razie sygnał w pętli narastałby w nieskończoność, prowadząc do przesterowania i zniekształceń.

Sygnał wyjściowy $y(n)$ formowany jest poprzez zsumowanie sygnału bezpośredniego (ścieżka „dry") z sygnałem przetworzonym z linii opóźniającej (ścieżka „wet")

$$y(n) = (1 - \alpha)x(n) + \alpha w(n - D), \tag{3.12}$$

gdzie α jest parametrem określającym balans między sygnałem czystym a efektem. Implementację algorytmu przedstawiono w formie pseudokodu na rysunku 3.8.

3.5.2 Chorus

Efekt Chorus służy do symulowania dźwięku wydawanego przez wielu artystów grających tę samą melodię [Zöl11]. Ponieważ instrumenty nie są ani identyczne, ani idealnie nastrojone i zsynchronizowane, powstają niewielkie różnice w wysokości dźwięku, dynamice, rytmie i barwie melodii. Algorytmiczna implementacja efektu Chorus w niniejszej pracy bazuje na opisie zawartym w książce „DAFX: Digital Audio Effects" autorstwa Udo Zölzera [Zöl11].

Zasada działania efektu polega na mieszaniu oryginalnego sygnału z jego kopią, która jest nieznacznie opóźniona w czasie. Opóźnienie to jest umieszczone w strefie fuzji Haasa (ok. 25 ms), by mózg nie interpretował go jako echo, a jedynie

zwielokrotnienie źródła dźwięku [Haa49]. Wartość opóźnienia jest losowo modulowana, co wprowadza subtelne, płynne wahania w wysokości dźwięku, tworząc iluzję wielu głosów.

Do modyfikacji dostępne są trzy parametry. Parametr D_{depth} określa głębokość modulacji, czyli zakres wahań czasu opóźnienia – wyższa wartość powoduje bardziej intensywne odstrojenie i wyraźniejszy efekt. Parametr częstotliwości modulacji definiuje szybkość zmian opóźnienia, wpływając na charakter brzmienia od powolnego, płynnego falowania do szybkiego vibrato. Parametr α balansujący proporcje między sygnałem oryginalnym a przetworzonym.

Sygnał wyjściowy $y(n)$ jest sumą sygnału bezpośredniego oraz opóźnionego, w proporcjach kontrolowanych przez parametr „Mix" α (analogicznie jak w efekcie Delay)

$$y(n) = (1 - \alpha)x(n) + \alpha x(n - d(n)), \tag{3.13}$$

gdzie $x(n)$ to sygnał wejściowy, a $d(n)$ to dynamicznie zmienna, stochastyczna wartość opóźnienia. Czas ten jest obliczany jako:

$$d(n) = d_b + Ar(n)D_{depth}, \tag{3.14}$$

gdzie d_b to wartość bazowa opóźnienia, A to maksymalny zakres modulacji, D_{depth} to parametr głębokości z zakresu $[0.0, 1.0]$, a $r(n)$ to gładko interpolowana wartość losowa z zakresu $[-1.0, 1.0]$.

Algorytm implementuje proces „podążania za celem". W stałych odstępach czasu generowana jest nowa losowa wartość docelowa r_c, a następnie przez cały okres następuje płynna interpolacja liniowa wartości modulacji $r(n)$ od jej poprzedniego stanu do nowego celu, zgodnie ze wzorem

$$r(n) = r_c + \frac{r_t - r_c}{N} n, \tag{3.15}$$

gdzie N to liczba próbek pomiędzy stanem obecnym a celem r_t, a n to aktualny numer próbki w tym przedziale.

Ponieważ czas opóźnienia $d(n)$ przyjmuje wartości ułamkowe (na przykład 12.7 próbki), a w buforze cyfrowym dostępne są jedynie wartości w punktach całkowitych, konieczne jest wyznaczenie wartości sygnału pomiędzy próbkami. Bez interpolacji próbka zostałaby zaokrąglona do najbliższej wartości całkowitej, co spowodowałoby „skoki" w module opóźnienia i słyszalne artefakty. Zastosowano interpolację liniową, gdzie odczytywana próbka $x_{int}(n)$ jest obliczana jako średnia ważona

$$x_{int}(n) = x(k) + (x(k + 1) - x(k))f, \tag{3.16}$$

gdzie k to część całkowita, a f to część ułamkowa czasu opóźnienia. Kompletny algorytm przetwarzania przedstawiono w formie pseudokodu na rysunku 3.9.

3.6 Przetwarzanie splotowe

Algorytm przetwarzania splotowego umożliwia odwzorowanie charakterystyk liniowych układów niezmiennych w czasie (LTI) [Zöl11]. W praktyce oznacza to możliwość „skopiowania" brzmienia dowolnego pomieszczenia czy urządzenia – wystarczy

Funkcja: przetwarzaj_chorus(x, α, D_{depth})

// 1. Aktualizacja modulatora losowego
$r_t \leftarrow$ aktualizuj_modulator_losowy()

// 2. Obliczenie dynamicznego czasu opóźnienia
$d_t \leftarrow$ oblicz_czas_opóźnienia(r_t, D_{depth})

// 3. Odczyt opóźnionej próbki (z interpolacją)
// Funkcja uwzględnia ułamkową część opóźnienia
$probka_{del} \leftarrow$ odczytaj_z_interpolacją(d_t)

// 4. Mieszanie sygnału
$y \leftarrow x \cdot (1 - \alpha) + probka_{del} \cdot \alpha$

Zwróć y

Rysunek 3.9: Przetwarzanie próbki w efekcie Chorus

zarejestrować jego odpowiedź impulsową (ang. *impulse response, IR*), by następnie móc nakładać tę samą charakterystykę akustyczną na inne sygnały. W dziedzinie przetwarzania dźwięku metoda ta znajduje zastosowanie w odwzorowaniu akustyki przestrzeni (sal koncertowych, studiów nagraniowych) oraz charakterystyk brzmieniowych różnorodnych urządzeń analogowych takich jak wzmacniacze, mikrofony i głośniki [Zöl11].

Odpowiedź impulsowa $h(t)$ to reakcja układu LTI na pobudzenie impulsem jednostkowym, czyli deltą Diraca $\delta(t)$. Odpowiedź impulsowa o skończonej długości jednoznacznie charakteryzuje układ, umożliwiając wyznaczenie jego charakterystyki amplitudowo-fazowej w całym paśmie częstotliwości. W systemach cyfrowych operuje się na sygnałach próbkowanych w czasie. W procesie dyskretyzacji ciągła odpowiedź $h(t)$ zostaje zamieniona na skończony ciąg próbek $h(n)$. W domenie dyskretnej odpowiednikiem pobudzenia jest impuls jednostkowy (delta Kroneckera $\delta(n)$), gdzie $\delta(0) = 1$ oraz $\delta(n) = 0$ dla $n \neq 0$.

Przejście z domeny ciągłej do dyskretnej jest realizowane przez zastąpieniem całki splotowej sumą splotową. Dla sygnałów ciągłych relację wejście-wyjście opisuje wzór

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau. \tag{3.17}$$

W dziedzinie cyfrowej, dla skończonej długości odpowiedzi impulsowej L, operacja ta sprowadza się do równania różnicowego

$$y(n) = \sum_{k=0}^{L-1} h(k)x(n-k), \tag{3.18}$$

gdzie $h(k)$ to kolejne próbki dyskretnej odpowiedzi impulsowej, a $x(n-k)$ to próbki sygnału wejściowego opóźnione o k okresów próbkowania.

Funkcja: `przetwarzaj_splot(x, h, L, G_comp)`
Zmienne: bufor (wektor przechowujący historię próbek wejściowych)
// 1. Aktualizacja bufora historii
// Przesunięcie wektora próbek i wprowadzenie nowej próbki x
aktualizuj_bufor(bufor, x)
// 2. Obliczenie sumy splotowej (Równanie 3.18)
// h to tablica próbek dyskretnej odpowiedzi impulsowej
suma ← 0.0
for k ← 0 **do** L − 1 **do**
    suma ← suma + h(k) · bufor(k)
end for
// 3. Kompensacja wzmocnienia i wyjście
y ← suma · G_comp
Zwróć y

Rysunek 3.10: Przetwarzanie próbki w efekcie przetwarzania splotowego

Z punktu widzenia realizacji algorytmicznej, operacja splotu dyskretnego jest równoznaczna z zastosowaniem filtru o skończonej odpowiedzi impulsowej (FIR, ang. *Finite Impulse Response*). W tej konfiguracji tablica próbek $h(k)$, uzyskana w procesie dyskretyzacji, pełni rolę współczynników wagi filtru. Transmitancja takiego układu w dziedzinie $\mathbb{Z}$ jest wielomianem stopnia $L-1$

$$H(z) = \sum_{k=0}^{L-1} h(k) z^{-k}. \tag{3.19}$$

Implementacja splotu w strukturze FIR charakteryzuje się bezwarunkową stabilnością, ponieważ układ nie posiada pętli sprzężenia zwrotnego – wyjście zależy wyłącznie od ważonej sumy próbek wejściowych (bieżącej i przeszłych).

Proces wyznaczania pojedynczej próbki wyjściowej $y(n)$ wymaga wykonania L operacji mnożenia i dodawania. Algorytm pobiera bieżącą próbkę $x(n)$ oraz musi mieć dostęp do $L-1$ próbek z bufora historii (linii opóźniającej). Algorytm splotu dyskretnego przedstawiono na rysunku 3.10.

Rozdział 4

Projekt systemu

Rozdział przedstawia kompletny proces projektowania będącego przedmiotem pracy systemu – począwszy od sformułowania wymagań funkcjonalnych i niefunkcjonalnych, poprzez analizę zapotrzebowania obliczeniowego i dobór platformy mikroprocesorowej, aż po realizację warstwy sprzętowej i zaprojektowanie architektury oprogramowania. W kolejnych podrozdziałach opisano zarówno podejmowane decyzje inżynierskie, jak i ich uzasadnienie w kontekście założeń projektowych oraz ograniczeń technologicznych.

4.1 Założenia projektowe

Projektowany system cyfrowego przetwarzania sygnałów audio ma stanowić autonomiczną platformę umożliwiającą modyfikację sygnałów akustycznych w czasie rzeczywistym. Celem pracy jest stworzenie kompletnego rozwiązania sprzętowo-programowego realizującego złożone algorytmy DSP przy zachowaniu niskiego opóźnienia oraz wysokiej jakości dźwięku. System może znaleźć zastosowanie w różnorodnych aplikacjach wymagających przetwarzania dźwięku, w tym w kontekście komunikacji człowiek-robot, gdzie odpowiednia modyfikacja sygnałów akustycznych może wpływać na jakość interakcji i percepcję systemu przez użytkownika.

4.1.1 Wymagania funkcjonalne

System musi zapewniać przetwarzanie sygnału audio w czasie rzeczywistym z opóźnieniem nieprzekraczającym 10 ms, co jest wartością akceptowalną dla aplikacji wymagających bezpośredniej interakcji akustycznej [LB05]. Zaimplementowany łańcuch przetwarzania obejmuje siedem efektów: bramkę szumów (Noise Gate), korektor trójpasmowy (Equalizer), nieliniowe zniekształcenia realizowane w trzech wariantach (Overdrive, Distortion, Fuzz), symulator charakterystyki akustycznej (Cabinet Simulator), oraz efekty oparte na liniach opóźniających (Chorus, Delay). Dobór efektów podyktowany został zarówno zróżnicowaniem zastosowanych technik przetwarzania (filtracja IIR, przetwarzanie dynamiki, waveshaping, splot dyskretny), jak i potencjalną użytecznością w szerokim spektrum zastosowań audio.

4.1.2 Wymagania niefunkcjonalne

Jakość przetwarzania sygnału określono na poziomie standardu CD-Audio: częstotliwość próbkowania 44100 Hz oraz rozdzielczość 16 bitów. Takie parametry zapewniają pełne odwzorowanie pasma słyszalnego (20 Hz - 20 kHz) zgodnie z twierdzeniem Kotielnikowa-Shannona [Sha49] oraz odpowiednio niski szum kwantyzacji (stosunek sygnału do szumu wynosi około 96 dB).

System musi charakteryzować się stabilnością działania, rozumianą jako brak słyszalnych „artefaktów" takich jak trzaski, przeskoki czy zniekształcenia. W praktyce, źródłem tego typu zakłóceń w systemach audio są najczęściej niedoskonałości warstwy sprzętowej, takie jak nieodpowiednie połączenia sygnałowe, niewystarczająca filtracja zasilania czy brak separacji masy analogowej i cyfrowej, prowadzące do sprzężeń i wnikania zakłóceń wysokoczęstotliwościowych do ścieżki audio [Ott88, Sel13]. Z punktu widzenia algorytmicznego, system musi być dodatkowo zabezpieczony przed niestabilnością numeryczną (np. w filtrach IIR) oraz przepełnieniami arytmetycznymi w operacjach o dużej dynamice. Wszystkie efekty muszą działać deterministycznie i przewidywalnie w całym zakresie regulacji parametrów.

Architektura systemu została zaprojektowana modularnie, gdzie każdy efekt stanowi niezależny moduł programowy z ustandaryzowanym interfejsem. Taki model umożliwia łatwe dodawanie nowych algorytmów, modyfikację istniejących oraz rekonfigurację łańcucha przetwarzania bez ingerencji w pozostałe komponenty systemu.

4.2 Dobór platformy sprzętowej

Wyzwaniem w projektowaniu cyfrowych systemów audio czasu rzeczywistego jest dotrzymanie rygorystycznych ograniczeń czasowych aby uniknąć opóźnienia wynikającego z długiego przetwarzania sygnału [Zöl11]. Dla przyjętej częstotliwości próbkowania $f_s = 44100$ Hz, okres próbkowania wynosi $T_s \approx 22{,}68$ µs. System operuje na blokach danych o długości $N = 256$ próbek, co oznacza maksymalny dostępny czas na przetworzenie jednego bufora (tzw. budżet czasowy) równy

$$t_{budget} = N \cdot T_s = 256 \cdot \frac{1}{44100} \approx 5{,}80 \text{ ms}. \tag{4.1}$$

W tym oknie czasowym procesor musi nie tylko wykonać obliczenia dla wszystkich aktywnych efektów, ale również obsłużyć transfery DMA (ang. *Direct Memory Access*), komunikację z peryferiami oraz logikę sterującą [Kat14].

Najbardziej wymagającym obliczeniowo elementem łańcucha jest przetwarzanie splotowe, realizowany jako splot sygnału z odpowiedzią impulsową o długości $L = 512$ próbek. Dla każdej próbki wyjściowej konieczne jest wykonanie 512 operacji mnożenia z akumulacją (MAC) [Smi07]. Uwzględniając pozostałe efekty (w szczególności interpolację w liniach opóźniających i filtry IIR w korektorze) oraz narzut systemowy, szacowane zapotrzebowanie na moc obliczeniową około 22 milionów instrukcji na sekundę przekracza możliwości standardowych mikrokontrolerów ogólnego przeznaczenia. Dodatkowo, ze względu na konieczność buforowania długich fragmentów sygnału (efekt Delay o czasie 500 ms wymaga ok. 88 kB pamięci RAM), platforma musi dysponować znacznymi zasobami pamięci operacyjnej.

4.2.1 Wybór mikrokontrolera

Na podstawie przeprowadzonej analizy, jako jednostkę centralną systemu wybrano mikrokontroler STM32H753ZIT6 firmy STMicroelectronics [STM21]. Układ ten oparty jest na rdzeniu ARM Cortex-M7 i oferuje parametry spełniające z zapasem wymagania projektu. Taktowanie rdzenia z częstotliwością 480 MHz zapewnia wydajność rzędu 1027 DMIPS [Wikb], co pozwala na realizację algorytmów DSP, bez ryzyka przekroczenia budżetu czasowego. Sprzętowe wsparcie dla operacji na liczbach zmiennoprzecinkowych poprzez jednostkę FPU (pojedynczej i podwójnej precyzji) [ARM14] umożliwia implementację algorytmów audio bezpośrednio na typach float. Układ dysponuje 1 MB pamięci RAM, co jest kluczowe dla implementacji efektów opartych na liniach opóźniających takich jak Delay i Chorus. Podział pamięci na różne domeny (DTCM, AXI SRAM, SRAM1-4) pozwala na optymalizację dostępu do danych poprzez umieszczenie buforów audio w pamięci dostępnej dla DMA, a stosu i zmiennych lokalnych w szybkiej pamięci DTCM. Rdzeń Cortex-M7 wspiera zestaw instrukcji SIMD, co w połączeniu z biblioteką CMSIS-DSP pozwala na znaczące przyspieszenie operacji wektorowych, takich jak filtrowanie FIR czy obliczanie iloczynów skalarnych. Wybór serii H7 zamiast popularnych w aplikacjach audio serii F4 [Ins20, Ele23] (np. STM32F407) podyktowany był przede wszystkim koniecznością obsługi długich splotów FIR w czasie rzeczywistym, co na słabszych jednostkach mogłoby wymagać drastycznego skrócenia odpowiedzi impulsowej i pogorszenia jakości symulacji.

4.2.2 Peryferia audio

Akwizycji sygnału wejściowego została zrealizowana z wykorzystaniem wbudowanego 16-bitowego przetwornika analogowo-cyfrowego ADC1 mikrokontrolera STM32H753ZI. Konwersja próbek jest wyzwalana sprzętowo przez timer TIM3, co zapewnia zachowanie częstotliwości próbkowania 44100 Hz bez udziału procesora. Próbki są automatycznie transferowane do pamięci przez kontroler DMA1 w trybie cyklicznym [STM23], co eliminuje konieczność ręcznego kopiowania danych przez CPU i minimalizuje obciążenie. Bufor wejściowy o długości 256 próbek jest podzielony na dwie sekcje, umożliwiając przetwarzanie w modelu podwójnego buforowania [ME96] – podczas gdy jedna połowa bufora jest wypełniana przez ADC, druga może być równolegle przetwarzana przez algorytmy DSP.

Tor wyjściowy audio został zaimplementowany z wykorzystaniem interfejsu I2S2 (Inter-IC Sound), który komunikuje się z zewnętrznym przetwornikiem cyfrowo-analogowym. Interfejs I2S jest standardem przemysłowym do przesyłania sygnałów audio między układami cyfrowymi [Phi96], oferującym synchroniczny przesył danych stereo z oddzielnym sygnałem zegarowym. Wybór zewnętrznego przetwornika DAC zamiast wbudowanego DAC mikrokontrolera podyktowany był jego wyższą jakością konwersji oraz niższym poziomem zniekształceń harmonicznych. Transfer danych do interfejsu I2S jest również realizowany przez DMA w trybie cyklicznym, zapewniając ciągłość strumienia audio. Bufor wyjściowy zawiera 512 próbek (256 próbek $\times$ 2 kanały stereo), co odpowiada tej samej latencji czasowej co bufor wejściowy.

4.3 Realizacja sprzętowa

Niniejszy rozdział opisuje fizyczną realizację systemu, obejmującą konstrukcję układów elektronicznych oraz konfigurację wyprowadzeń mikrokontrolera. Przedstawiono architekturę zasilania, która zapewnia separację domeny cyfrowej od analogowej, tor kondycjonowania sygnału wejściowego oraz mapowanie peryferiów w środowisku STM32CubeMX.

Schemat systemu (rysunek 4.1) podzielono na pięć sekcji funkcjonalnych. W sekcji zasilania napięcie wejściowe 10V DC konwertowane jest na 5V, a następnie na 1.65V, natomiast napięcie 3.3V uzyskiwane jest z linii USB (5V) przez wewnętrzny stabilizator. Sekcja interfejsu, zasilana napięciem 5V, zawiera enkodery oraz wyświetlacz i komunikuje się dwukierunkowo z układem sterującym. Sekcja wejściowa odpowiada za kondycjonowanie sygnału audio przy wykorzystaniu napięć 10V i 1.65V. W sekcji mikrokontrolera zintegrowano moduły ADC (wejście sygnału), Cortex-M7, DMA1 oraz I2S2. Magistrala I2S przekazuje dane do sekcji wyjściowej, gdzie znajduje się przetwornik DAC PCM5102 połączony ze złączem Jack 3.5mm. Schemat elektroniczny został zaprezentowany w dodatku A.

4.3.1 Zasilanie układów

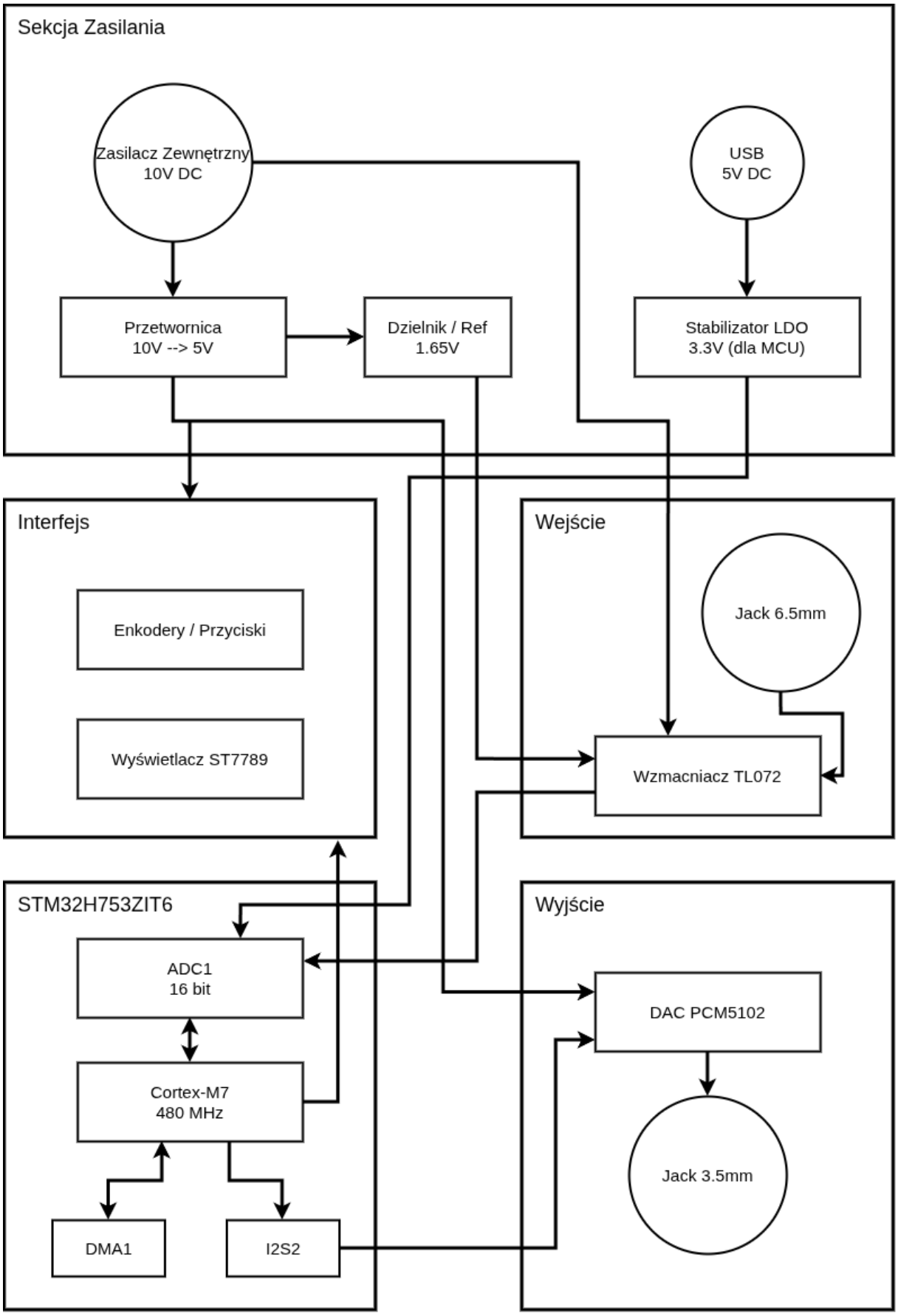
Część obliczeniowa (mikrokontroler STM32H753ZI) zasilana jest standardowym napięciem 5 V z interfejsu USB, które jest następnie stabilizowane wewnętrznie do poziomu 3.3 V. Sekcja analogowa oraz peryferia zewnętrzne korzystają z zewnętrznego zasilacza o napięciu 10 V, które jest konwertowane na napięcie 5 V za pomocą przetwornic napięcia. To napięcie 5 V zasila wyświetlacz, enkodery obrotowe oraz przetwornik DAC PCM5102.

Wyróżniającym elementem konstrukcji jest zasilanie wejściowego wzmacniacza operacyjnego TL072 w konfiguracji symetrycznej (-5 V oraz +5 V). Takie rozwiązanie zapewnia szeroki zakres dynamiki sygnału wejściowego oraz eliminuje problemy ze zniekształceniami nieliniowymi w pobliżu masy wirtualnej, typowe dla układów zasilanych napięciem pojedynczym.

Dodatkowo, w systemie wygenerowano precyzyjne napięcie referencyjne 1.65 V, służące do polaryzacji składowej stałej sygnału wejściowego, co pozwala na jego dopasowanie do zakresu pomiarowego przetwornika ADC (0–3.3 V).

4.3.2 Tor przetwarzania sygnału analogowego

Sygnał wejściowy trafia na bufor zrealizowany na wzmacniaczu operacyjnym TL072, gdzie jest przesuwany na poziom wirtualnej masy wzmacniacza (5 V). Następnie jest odpowiednio wzmocniony, aby jego wartość szczytowa nie przekroczyła 3.3 V, całkowicie wypełniając zakres pomiarowy przetwornika ADC i jednocześnie chroniąc go przed uszkodzeniem. W ostatnim kroku przygotowania sygnału następuje ponowne przeniesienie go na poziom 1.65 V. Po cyfrowej obróbce w mikrokontrolerze STM32, sygnał trafia do przetwornika cyfrowo-analogowego PCM5102. Przepływ sygnału audio odbywa się od kondycjonowanego wejścia analogowego, przez cyfrowy blok DSP w mikrokontrolerze, aż po wyjściowy przetwornik DAC.



Rysunek 4.1: Schemat blokowy systemu

4.3.3 Konfiguracja peryferiów i mapowanie wyprowadzeń

Konfigurację sprzętową przeprowadzono w środowisku STM32CubeMX [STM25], generując kod inicjalizacyjny warstwy HAL. Kompletny diagram wyprowadzeń mikrokontrolera zrzutowany ze środowiska konfiguracyjnego widoczny jest na rysunku 4.2. Aby zapewnić wystarczającą moc obliczeniową dla złożonych algorytmów DSP, układ taktowany jest z częstotliwością 480 MHz, uzyskiwaną z wewnętrznej pętli fazowej (PLL) synchronizowanej oscylatorem HSI. Magistrale peryferyjne pracują z częstotliwościami 240 MHz dla szyny AHB oraz 120 MHz dla szyn APB.

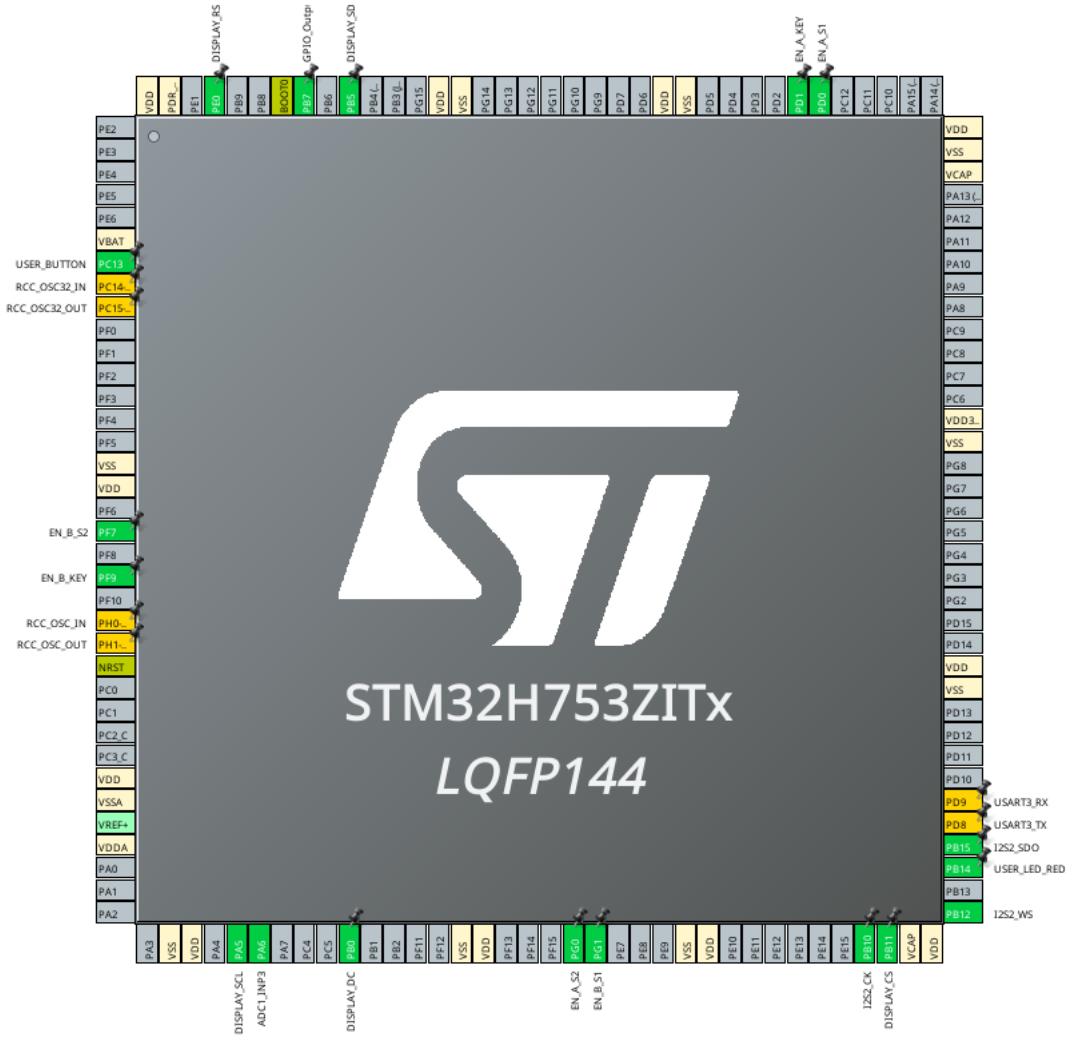
Akwizycja sygnału audio realizowana jest przez 16-bitowy przetwornik ADC1. Sygnał z układu wejściowego podawany jest na pin `PA6`, skonfigurowany jako trzeci kanał analogowy przetwornika. Kluczowym aspektem konfiguracji jest system wyzwalania konwersji, który jest sprzętowo synchronizowany przez licznik `TIM3`. Timer ten generuje zdarzenia z częstotliwością dokładnie 44100 Hz. Przetwornik współpracuje z kontrolerem DMA1, który w trybie cyklicznym automatycznie przenosi wyniki konwersji do bufora w pamięci RAM.

Emisja dźwięku odbywa się za pośrednictwem interfejsu szeregowego I2S2 (SPI2), skonfigurowanego w trybie *Master Transmit* zgodnie ze standardem Philips. Do obsługi magistrali przypisano pin `PB10` jako linia zegarowa (CK), `PB12` jako sygnał wyboru kanału (WS) oraz `PB15` jako linia danych (SD0). Podobnie jak w przypadku wejścia, transfer danych do przetwornika DAC jest w pełni obsługiwany przez kanał DMA pracujący w trybie cyklicznym. Wyświetlacz graficzny ST7789 sterowany jest sygnałami zegara i danych wyprowadzonymi na piny `PA5` i `PB5`. Pierwszy enkoder wykorzystuje piny `PD0`, `PD1` i `PG0`, natomiast drugi podłączony jest do pinów `PG1`, `PF9` i `PF7`.

4.4 Architektura oprogramowania

Oprogramowanie systemu zostało zrealizowane w architekturze *bare-metal* (bez systemu operacyjnego), opartej na modelu sterowanym przerwaniami (ang. *interrupt-driven*) [Val12]. Ze względu na rygorystyczne wymagania czasowe przetwarzania sygnału audio, kluczowym założeniem projektu było rozdzielenie zadań krytycznych czasowo od zadań obsługi interfejsu użytkownika. Architektura systemu składa się z dwóch głównych wątków wykonania:

- **Wątek przetwarzania sygnału** – Realizowany w procedurach obsługi przerwań sterownika DMA. Posiada najwyższy priorytet i realizuje akwizycję, przetwarzanie oraz emisję dźwięku.
- **Wątek tła** – Realizowany w nieskończonej pętli głównej `while(1)`. Obsługuje interfejs graficzny, odczyt stanu enkoderów oraz aktualizację parametrów efektów.



Rysunek 4.2: Konfiguracja wyprowadzeń mikrokontrolera STM32H753 w środowisku STM32CubeMX

4.4.1 Przepływ danych i technika podwójnego buforowania

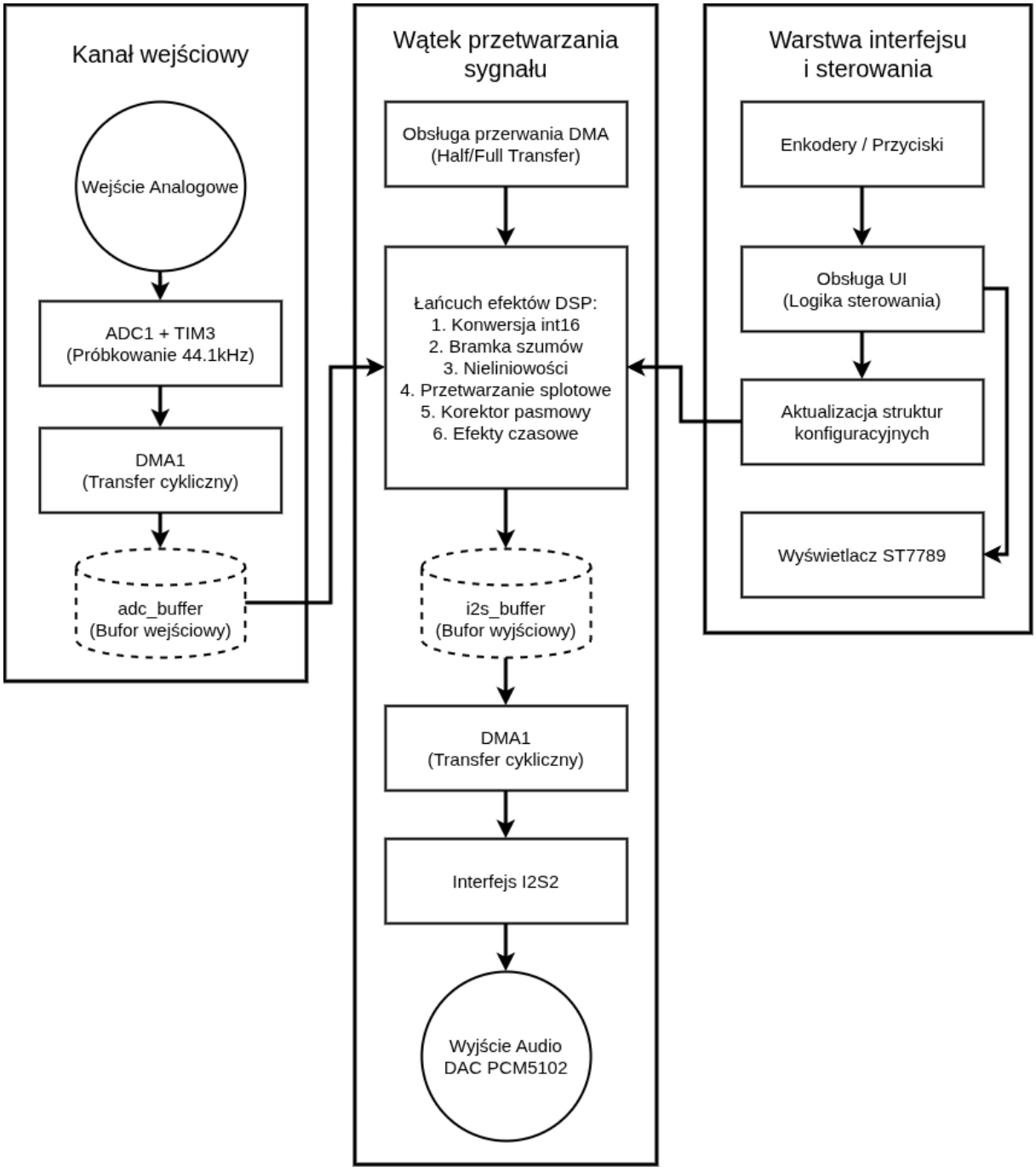
Aby zapewnić ciągłość strumienia audio bez przerw i zakłóceń, w systemie zastosowano mechanizm podwójnego buforowania. W pamięci RAM zadeklarowano dwa główne bufory cykliczne:

- `adc_buffer`: bufor wejściowy przechowujący surowe dane z przetwornika ADC,
- `i2s_buffer`: bufor wyjściowy dla interfejsu I2S.

Proces przetwarzania przebiega następująco:

1. Układ peryferyjny DMA automatycznie przepisuje dane z ADC do bufora w pamięci, nie obciążając procesora.
2. Po zapełnieniu pierwszej połowy bufora, generowane jest przerwanie. W tym czasie DMA kontynuuje zapis do drugiej połowy bufora.
3. Procesor przetwarza dane z pierwszej połowy, aplikując łańcuch efektów DSP, i zapisuje wynik do bufora wyjściowego.
4. Analogiczna operacja następuje po zapełnieniu całego bufora – procesor przetwarza drugą połowę, podczas gdy DMA nadpisuje pierwszą nowymi danymi.

Takie rozwiązanie gwarantuje, że procesor zawsze operuje na stabilnych danych, które nie są w danym momencie modyfikowane przez układ peryferyjny. Architekturę oprogramowania (rysunek 4.3) zorganizowano w trzech pionowych sekcjach funkcjonalnych. Lewa kolumna przedstawia kanał wejściowy, w którym sygnał z wejścia analogowego jest próbkowany przez układ ADC1 (sterowany timerem TIM3) i transferowany przez DMA1 do bufora adc_buffer. Środkowa sekcja obrazuje wątek przetwarzania sygnału: obsługa przerwania DMA inicjuje przepływ danych przez łańcuch efektów DSP (pobierający próbki z bufora wejściowego), a wynik trafia do bufora i2s_buffer, skąd jest transmitowany przez DMA1 i interfejs I2S2 na wyjście audio. Prawa sekcja obejmuje warstwę interfejsu i sterowania, gdzie sygnały z enkoderów trafiają do logiki obsługi UI, która steruje wyświetlaczem oraz aktualizuje struktury konfiguracyjne, modyfikujące bezpośrednio parametry łańcucha efektów DSP.



Rysunek 4.3: Schemat blokowy architektury oprogramowania i przepływu danych w systemie.

4.4.2 Potok przetwarzania sygnału

Łańcuch przetwarzania sygnału został ułożony w następującej sekwencji:

1. **Konwersja typów** – Rzutowanie 16-bitowych danych bez znaku (`uint16_t`) z ADC na typ ze znakiem (`int16_t`) poprzez odjęcie składowej stałej.
2. **Bramka szumów** – Eliminacja zakłóceń tła, przed etapami wzmocnienia.
3. **Wzmocnienie wstępne** – Liniowe podbicie sygnału.
4. **Blok nieliniowy**
   - *Overdrive:* Symulacja miękkiego nasycenia.
   - *Distortion/Fuzz:* Agresywne obcinanie sygnału.
5. **Przetwarzanie splotowe** – Splot sygnału z odpowiedzią impulsową.
6. **Korektor pasmowy** – Trójpasmowa korekcja barwy realizowana filtrami IIR drugiego rzędu.
7. **Efekty modulacyjne i czasowe**
   - *Chorus:* Modulacja linii opóźniającej dla uzyskania efektu „przestrzenności".
   - *Delay:* Efekt echa z pętlą sprzężenia zwrotnego.

Wynikowy sygnał jest kopiowany do obu kanałów bufora wyjściowego, co umożliwia współpracę ze stereofonicznym przetwornikiem DAC.

4.4.3 Organizacja pamięci w STM32H7

Kontroler DMA1 nie ma dostępu do najszybszej pamięci DTCM (ang. *Tightly Coupled Memory*), więc konieczne było jawne rozmieszczenie buforów audio w odpowiednim obszarze pamięci. W kodzie programu został wykorzystany dostępny atrybut kompilatora GCC `__attribute__((section(".dma_buffer")))`. Powyższa dyrektywa wymusza umieszczenie tablic `adc_buffer` oraz `i2s_buffer` w domenie pamięci RAM D2. Zmienne lokalne efektów oraz stos programu znajdują się w pamięci DTCM, co zapewnia maksymalną wydajność obliczeniową rdzenia Cortex-M7.

4.4.4 Warstwa interfejsu i sterowania

System wykorzystuje wzorzec odpytywania z użyciem mechanizmów programowej eliminacji drgań styków dla enkoderów obrotowych. Zmiana parametrów odbywa się poprzez modyfikację globalnych struktur konfiguracyjnych poszczególnych efektów. Dzięki atomowości zapisu zmiennych prostych na platformie 32-bitowej, nie było konieczności stosowania zaawansowanych mechanizmów synchronizacji pomiędzy wątkiem UI a wątkiem audio dla pojedynczych parametrów sterujących.

Implementacja interfejsu graficznego oparta jest na strukturach, które przechowują stan każdego modułu (włączony/wyłączony, wartość parametru, pozycja na ekranie). Odświeżanie wyświetlacza ST7789 odbywa się tylko w przypadku wykrycia zmiany wartości, co minimalizuje obciążenie procesora.

Rozdział 5

Weryfikacja działania systemu i testy

W rozdziale przedstawiono wyniki testów zaimplementowanych algorytmów przetwarzania sygnałów audio. Celem badań było potwierdzenie poprawności działania poszczególnych bloków przetwarzania, zgodności z założeniami projektowymi oraz ocena jakości sygnału wyjściowego.

Środowisko testowe składało się z laptopa pełniącego rolę generatora oraz oscyloskopu cyfrowego służącego do rejestracji przebiegów. Sygnały testowe zostały wygenerowane programowo za pomocą dedykowanego skryptu napisanego w języku Python, co pozwoliło na precyzyjną kontrolę parametrów fali. Sygnał był podawany na wejście analogowe urządzenia za pośrednictwem przewodu audio zakończonego wtykiem Jack 3.5 mm, podłączonego do wyjścia karty dźwiękowej. Pomiary oscyloskopowe realizowano w trybie sprzężenia zmiennoprądowego.

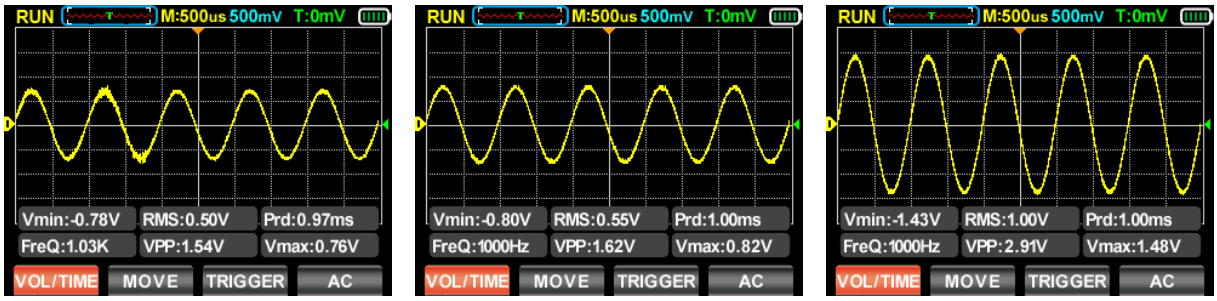
O ile w opisie konkretnego eksperymentu nie wskazano inaczej, domyślnym sygnałem wymuszającym była sinusoida o częstotliwości 1000 Hz. Amplituda sygnału wejściowego została ustalona na poziomie 1.5 V_{pp} (napięcie międzyszczytowe). Podstawa czasu oscyloskopu była dobierana dynamicznie, adekwatnie do częstotliwości badanego przebiegu. Czułość napięciowa oscyloskopu wynosiła 500 $\frac{mV}{div}$, natomiast podstawa czasowa 500µs o ile w opisie eksperymentu nie wskazano inaczej. Testowany układ został zaprezentowany w dodatku B.

5.1 Analiza toru przetwarzania

W pierwszej kolejności poddano weryfikacji poprawność działania analogowych i cyfrowych bloków toru sygnałowego w trybie przepustowym, bez aktywacji algorytmów modyfikujących brzmienie. Badanie miało na celu potwierdzenie, że system poprawnie przenosi sygnał użyteczny oraz określenie wzmocnienia w poszczególnych punktach kontrolnych.

Wyniki pomiarów przedstawiono na rysunku 5.1. Sygnał referencyjny z generatora (rysunek 5.1a)) charakteryzował się amplitudą międzyszczytową na poziomie 1.54 V_{pp}. Rysunek 5.1b) obrazuje sygnał mierzony za wejściowym wzmacniaczem operacyjnym, bezpośrednio przed przetwornikiem ADC. Układ ten wprowadził nieznaczne wzmocnienie, podnosząc poziom sygnału do wartości 1.62 V_{pp}.

Na rysunku 5.1c) widoczny jest sygnał zarejestrowany na wyjściu przetwornika cyfrowo-analogowego (DAC). W tym punkcie amplituda sygnału wyniosła 2.91 V_{pp},



a) b) c)

Rysunek 5.1: Przebiegi sygnału sinusoidalnego 1000 Hz w kluczowych punktach systemu,

co wynika z parametrów konfiguracyjnych stopnia wyjściowego. Sygnał zachowuje kształt sinusoidalny na każdym etapie przetwarzania. Nie zaobserwowano widocznych zniekształceń harmonicznych ani znaczącego poziomu szumów, co świadczy o wysokiej transparentności toru audio.

5.2 Efekty nieliniowe

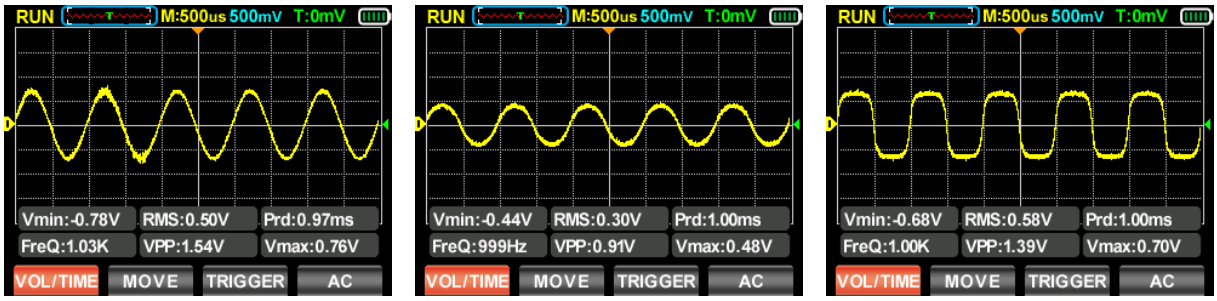
W tej części badań poddano analizie wpływ algorytmów nieliniowych na kształt fali sinusoidalnej. Dla każdego z efektów zarejestrowano przebiegi dla dwóch wartości wzmocnienia wejściowego: niskiego ($G_{in} = 2.0$) oraz wysokiego ($G_{in} = 10.0$). Jako punkt odniesienia na każdym rysunku umieszczono w części a) nieprzetworzony sygnał wejściowy.

5.2.1 Overdrive

Algorytm Overdrive symuluje łagodne nasycenie, charakterystyczne dla przesterowanych wzmacniaczy lampowych. Na rysunku 5.2b) ($G_{in} = 2.0$) widoczny jest sygnał o bardzo gładkich i zaokrąglonych wierzchołkach. Algorytm wprowadza kompresję dynamiki, nie powodując jednak ostrych załamań przebiegu. Przy znacznym zwiększeniu wzmocnienia do $G_{in} = 10.0$ (rysunek 5.2c)), sygnał zaczyna nieco bardziej przypominać falę prostokątną, jednak – co jest kluczową cechą tego efektu – fragmenty o dynamicznej zmianie wartości przebiegu nadal pozostają mocno wygładzone, co przekłada się na „ciepłą” barwę dźwięku.

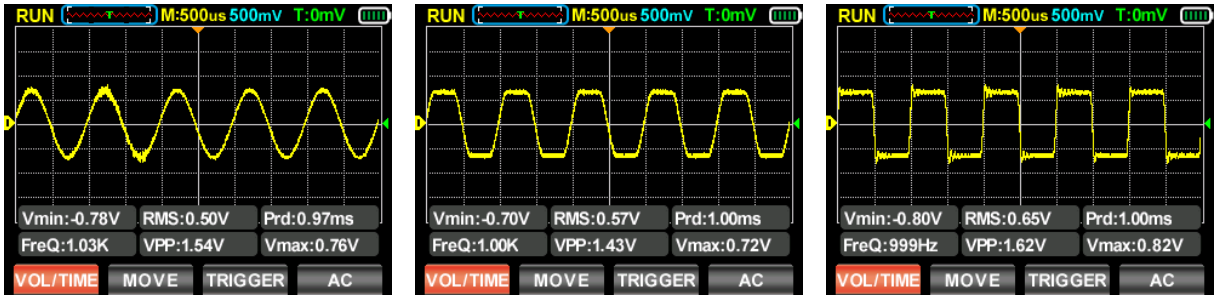
5.2.2 Distortion

Efekt Distortion realizuje bardziej agresywną formę obcinania sygnału, opartą na funkcji wykładniczej. Analiza przebiegu dla wzmocnienia $G_{in} = 2.0$ (rysunek 5.3b)) ukazuje formowanie się fali zbliżonej kształtem do trapezu, z wyraźniejszym niż w przypadku efektu Overdrive spłaszczeniem wierzchołków. Przy wysokim wzmocnieniu $G_{in} = 10.0$ (rysunek 5.3c)) sygnał ulega silnej saturacji, upodabniając się niemal całkowicie do fali prostokątnej o stromych zboczach. Barwa tego



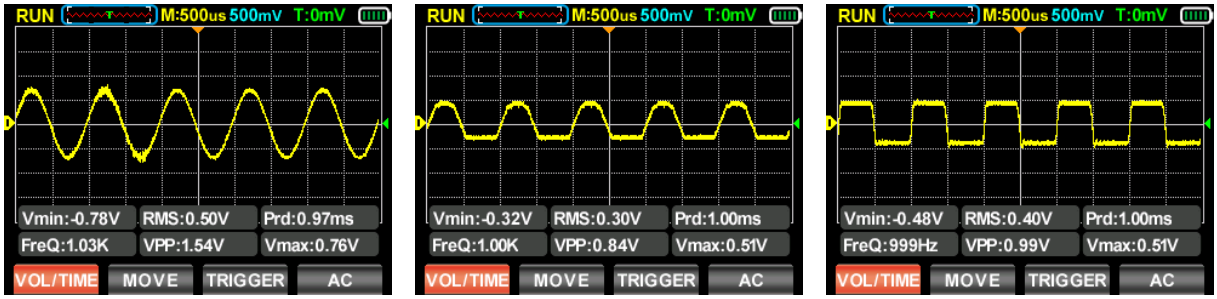
a) b) c)

Rysunek 5.2: Przetwarzanie z efektem Overdrive,



a) b) c)

Rysunek 5.3: Przetwarzanie z efektem Distortion,



a) b) c)

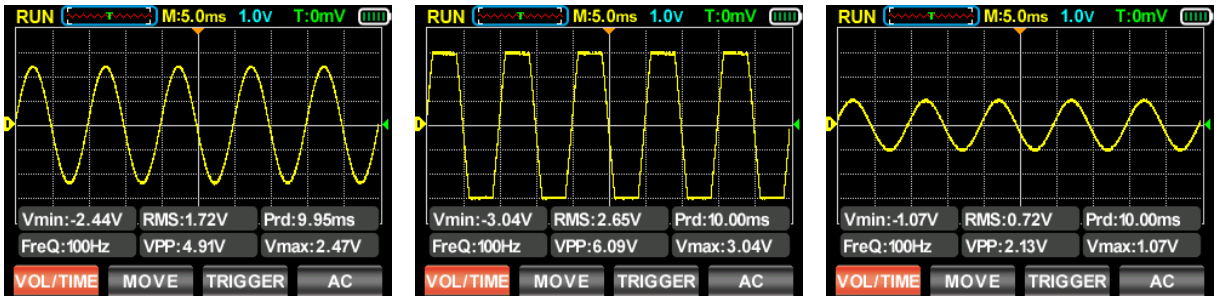
Rysunek 5.4: Przetwarzanie z efektem Fuzz,

efektu jest określana jako agresywna i nasycona. Distortion wzbogaca widmo sygnału nie tylko o harmoniczne nieparzyste, ale również o harmoniczne parzyste. Generowane są silne harmoniczne wysokiego rzędu, co nadaje brzmieniu ostrości zbliżonej do fali prostokątnej.

5.2.3 Fuzz

Efekt Fuzz charakteryzuje się niesymetrycznym działaniem, wynikającym z zastosowania różnych progów obcinania dla dodatniej i ujemnej części sygnału.

Specyficzną naturę algorytmu widać wyraźnie na rysunku 5.4b) ($G_{in} = 2.0$). Dolna połowa okresu została całkowicie spłaszczona (twarde obcięcie), podczas gdy górna połówka zachowuje kształt zbliżony do oryginalnej sinusoidy, ulegając jedy-



a) b) c)

Rysunek 5.5: Korektor niskotonowy,

nie nieznacznym modyfikacjom. Taka asymetria generuje parzyste harmoniczne. Zwiększenie wzmocnienia do $G_{in} = 10.0$ (rysunek 5.4c)) prowadzi do ekstremalnego przesterowania obu części, w wyniku czego otrzymujemy sygnał praktycznie prostokątny. Efekt Fuzz charakteryzuje się „szortską" fakturą brzmienia.

5.3 Korektor pasmowy (Equalizer)

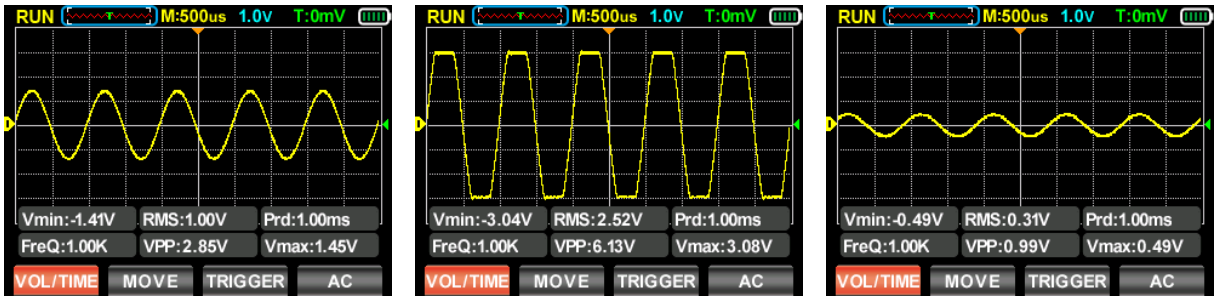
Weryfikację korektora przeprowadzono oddzielnie dla każdego z trzech pasm. Test polegał na podaniu sygnału o częstotliwości środkowej danego filtru i obserwacji zmian amplitudy. Czułość napięciowa oscyloskopu we wszystkich pomiarach wynosiła 1 $\frac{V}{div}$.

5.3.1 Pasmo niskie (Low Shelf, 100 Hz)

Dla filtru dolnopółkowego sygnałem testowym była sinusoida o częstotliwości 100 Hz. Podstawa czasu oscyloskopu została ustawiona na 5.0 ms. Rysunek 5.5a) przedstawia sygnał referencyjny przy ustawieniu neutralnym ($G = 0$ dB). Na rysunku 5.5b), obrazującym podbicie pasma ($G = +10$ dB), widoczne jest wyraźne spłaszczenie wierzchołków sinusoidy. Wzmocniony sygnał osiągnął amplitudę ok. 6 V_{pp}, przekraczając zakres dynamiki przetwornika DAC, co spowodowało jego sprzętowe nasycenie. Rysunek 5.5c) przedstawia sygnał po zastosowaniu tłumienia ($G = -10$ dB), gdzie amplituda została poprawnie zredukowana bez zniekształceń.

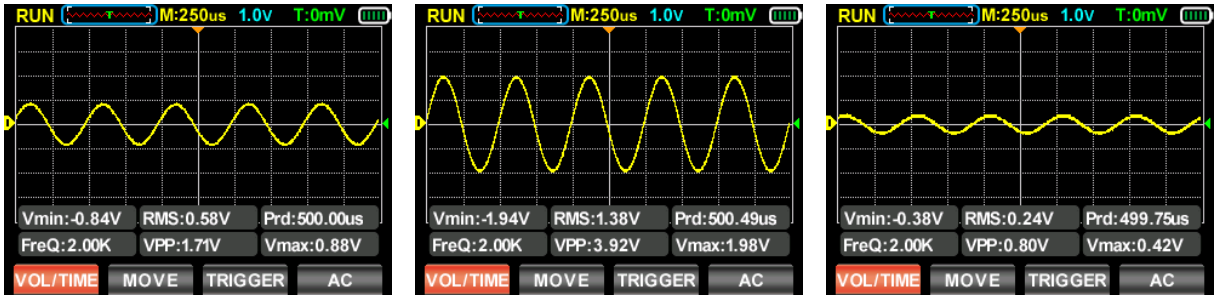
5.3.2 Pasmo środkowe (Peaking, 1000 Hz)

Test pasma środkowego przeprowadzono dla częstotliwości 1000 Hz przy podstawie czasu 500 µs. Analogicznie do pasma niskiego, rysunek 5.6a) stanowi odniesienie ($G = 0$ dB). Przy wzmocnieniu ustawionym na $G = +10$ dB (rysunek 5.6b)) ponownie zaobserwowano efekt nasycenia wyjścia DAC i obcinanie wierzchołków fali przy amplitudzie bliskiej 6 V_{pp}. Rysunek 5.6c) potwierdza skuteczność tłumienia przy nastawie $G = -10$ dB.



a) b) c)

Rysunek 5.6: Korektor środkowopasmowy,



a) b) c)

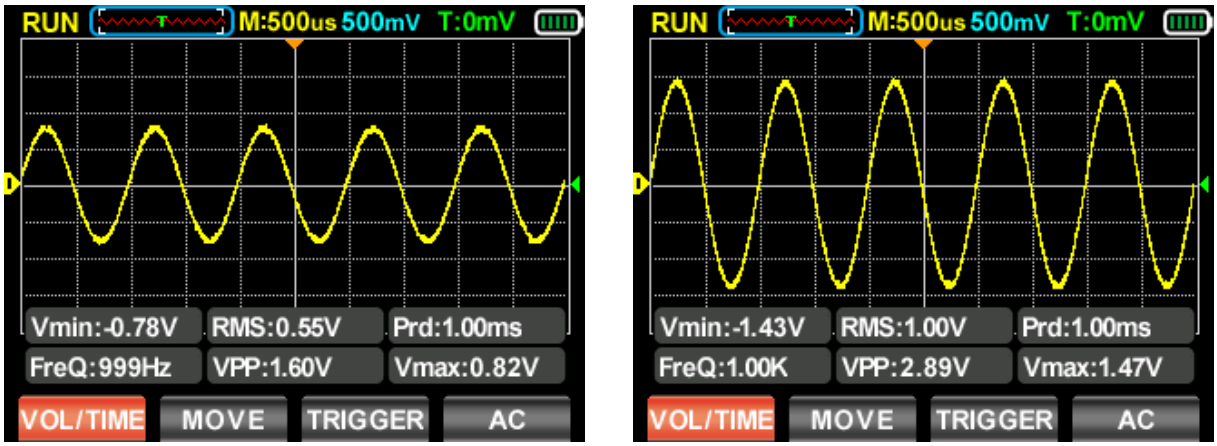
Rysunek 5.7: Korektor wysokotonowy,

5.3.3 Pasmo wysokie (High Shelf, 2000 Hz)

Dla filtru górnopółkowego użyto sygnału 2000 Hz. Podstawa czasu wynosiła 250 µs. Rysunek 5.7a) prezentuje sygnał wejściowy bez modyfikacji (G = 0 dB). Rysunek 5.7b) obrazuje działanie filtru w trybie wzmocnienia (G = +10 dB), natomiast rysunek 5.7c) przedstawia efekt tłumienia (G = −10 dB). W tym paśmie algorytm poprawnie realizuje zadaną charakterystykę, odpowiednio modyfikując amplitudę sygnału.

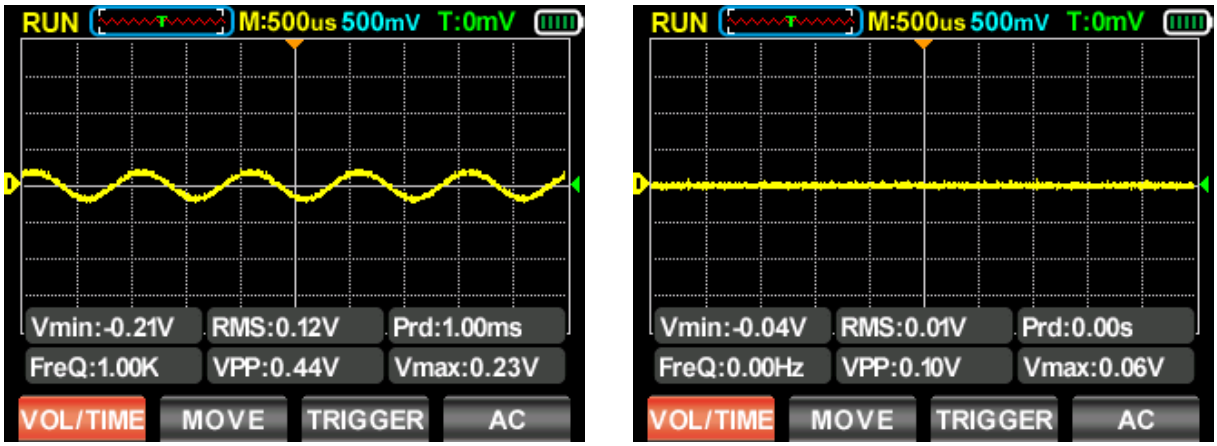
5.4 Bramka szumów

Test bramki szumów przeprowadzono dla dwóch poziomów sygnału wejściowego względem ustawionego progu L_{th}. Gdy amplituda sygnału wejściowego przekracza próg (rysunek 5.8a)), bramka otwiera się i przepuszcza sygnał na wyjście (rysunek 5.8b)). W przypadku sygnału o amplitudzie poniżej progu (rysunek 5.9a)), algorytm poprawnie blokuje transmisję, co skutkuje ciszą na wyjściu (płaska linia na rysunku 5.9b)).



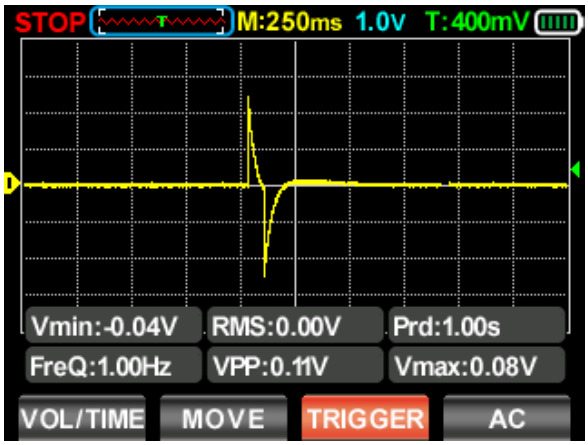
a) b)

Rysunek 5.8: Zachowanie bramki szumów w stanie otwartym,

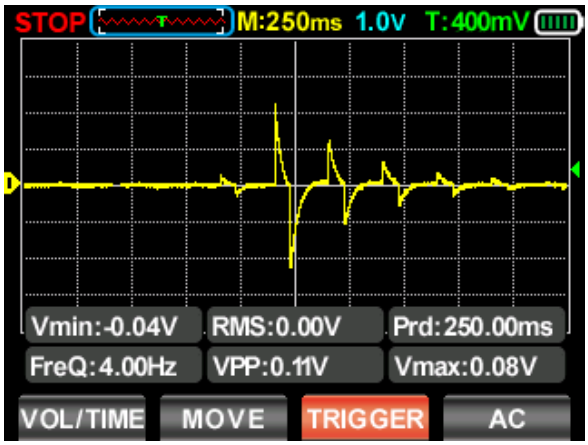


a) b)

Rysunek 5.9: Zachowanie bramki szumów w stanie zamkniętym,



a)



b)

Rysunek 5.10: Test efektu Delay,

5.5 Efekty czasowe

Weryfikację algorytmów opartych na liniach opóźniających przeprowadzono przy użyciu sygnałów impulsowych.

5.5.1 Delay

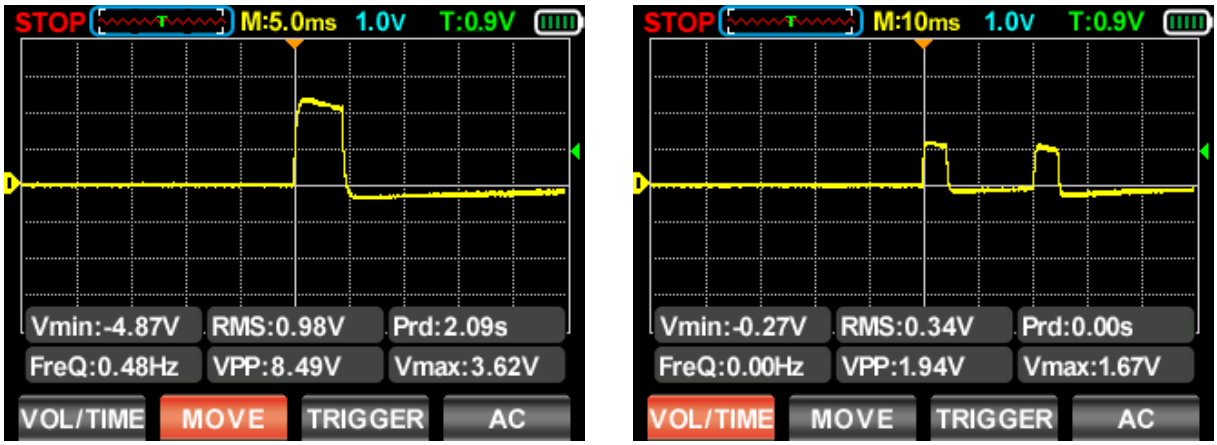
Badanie efektu Delay przeprowadzono przy użyciu impulsu prostokątnego o czasie trwania 100 ms. W algorytmie ustawiono czas opóźnienia na $D = 250$ ms oraz współczynnik sprzężenia zwrotnego na $g = 0.5$. Podstawa czasu oscyloskopu została dobrana na 250 $\frac{ms}{div}$, a czułość na $1\frac{V}{div}$.

Rysunek 5.10a) przedstawia sygnał przy wyłączonym efekcie – widoczny jest pojedynczy impuls wejściowy. Po aktywacji efektu (rysunek 5.10b)) obserwujemy serię powtórzeń. Zgodnie z zadanymi parametrami, kopia impulsu pojawia się dokładnie co 250 ms. Wartość sprzężenia zwrotnego $g = 0.5$ skutkuje w przybliżeniu dwukrotnym spadkiem amplitudy każdego kolejnego odbicia. Na zarejestrowanym przebiegu widoczne są trzy powtórzenia przed całkowitym wygaśnięciem sygnału.

5.5.2 Chorus

Do weryfikacji efektu Chorus wykorzystano krótki impuls o czasie trwania 30 ms. Parametry modulatora LFO ustawiono na: częstotliwość $f = 1$ Hz oraz głębokość $D_{depth} = 0.5$. Pomiary wykonano przy podstawie czasu 10 $\frac{ms}{div}$ i czułości $1\ \frac{V}{div}$.

Porównując przebieg referencyjny (rysunek 5.11a)) z przebiegiem przetworzonym (rysunek 5.11b)), zauważalna jest zmiana struktury czasowej sygnału. Dla zadanych parametrów, algorytm wygenerował kopię sygnału opóźnioną o 20 ms. Jest to wartość mieszcząca się w tzw. strefie Haasa, co potwierdza poprawność działania efektu – zamiast wyraźnego echa, uzyskano efekt „pogrubienia" i dublowania źródła dźwięku.



a) b)

Rysunek 5.11: Test efektu Chorus,

5.6 Przetwarzanie splotowe

Ostatnim etapem weryfikacji było przetestowanie algorytmu splotu. W badaniu wykorzystano odpowiedzi impulsowe (ang. *Impulse Response*) będące cyfrowym odwzorowaniem tonów gitarowych z dwóch klasycznych albumów zespołu Metallica: „Master of Puppets" (1986) oraz „Kill 'Em All" (1983).

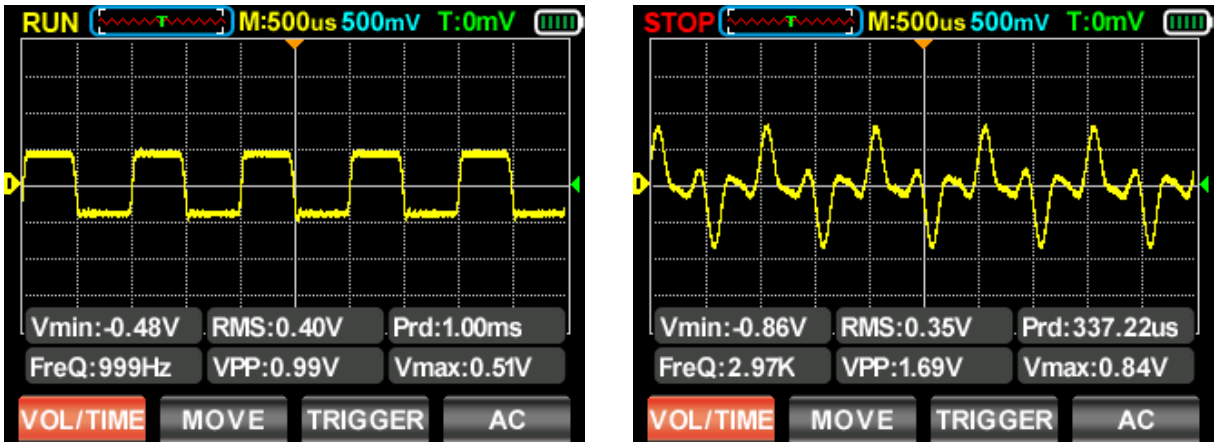
Z punktu widzenia inżynierii systemu robotycznego, test ten weryfikuje zdolność urządzenia do nakładania złożonych charakterystyk akustycznych na sygnał wyjściowy. Jako sygnał wymuszający zastosowano przebieg nasycony efektem Fuzz ($G_{in} = 10.0$). Sygnał ten, zbliżony kształtem do fali prostokątnej, posiada bogate widmo harmoniczne, co jest niezbędne do poprawnego wysterowania filtrów symulujących charakterystyki głośnikowe.

5.6.1 Symulacja „Master of Puppets"

Pierwsza badana IR („Puppet") odwzorowuje brzmienie charakteryzujące się silnym nasyceniem pasma dolnego i „ciężkim" charakterem. Porównanie sygnału wejściowego (rysunek 5.12a)) z wyjściowym (rysunek 5.12b)) ukazuje silne wygładzenie zboczy, świadczące o dolnoprzepustowym charakterze filtru. Sygnał wyjściowy cechuje się dużą amplitudą w zakresie niskich i średnich częstotliwości, co nadaje brzmieniu pełny i masywny charakter.

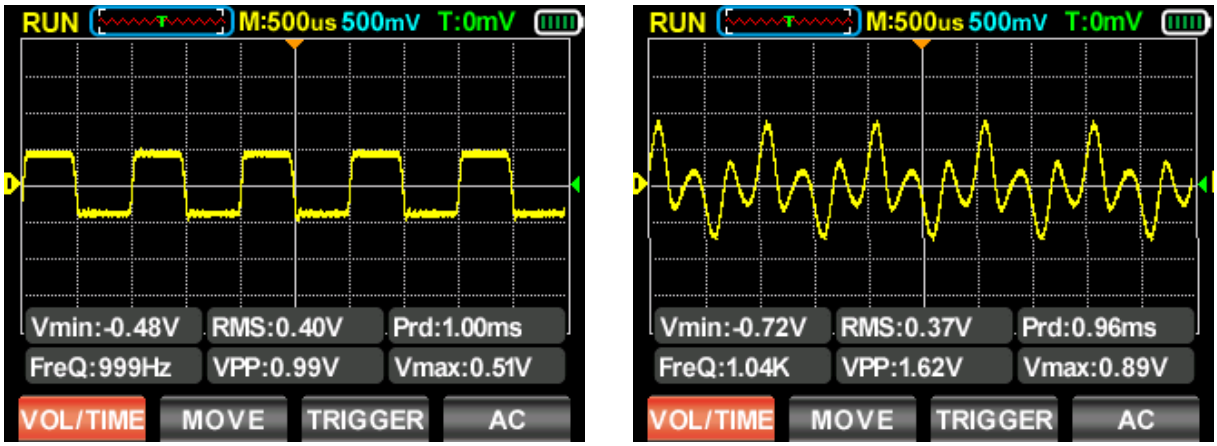
5.6.2 Symulacja „Kill 'Em All"

Druga odpowiedź impulsowa (IR „Killem") bazuje na surowym i jasnym brzmieniu debiutanckiego albumu zespołu. Analiza przebiegów (rysunek 5.13b) względem 5.13a)) ujawnia odmienną obwiednię sygnału. Węższa odpowiedź niskotonowa oraz inna struktura rezonansów generują jaśniejszą barwę w porównaniu do symulacji „Puppet". Różnice te potwierdzają, że algorytm splotu poprawnie różnicuje sygnały w zależności od zadanej charakterystyki.



a) b)

Rysunek 5.12: Splot z IR „Puppet” (Master of Puppets),



a) b)

Rysunek 5.13: Splot z IR „Killem” (Kill 'Em All),

Rozdział 6

Podsumowanie

Celem pracy było zaprojektowanie i wykonanie systemu cyfrowego przetwarzania sygnałów audio, dedykowanego do zastosowań w interakcji człowiek-robot. W ramach projektu opracowano warstwę sprzętową opartą na wydajnym mikrokontrolerze z rodziny STM32H7 oraz zaimplementowano szereg algorytmów DSP, umożliwiających kształtowanie brzmienia w czasie rzeczywistym. Powstałe urządzenie pozwala na modyfikację sygnałów akustycznych poprzez korekcję widmową, syntezę nieliniowości oraz efekty przestrzenne, co umożliwia nadawanie robotom pożądanych cech „osobowości" głosowej. Przeprowadzone testy weryfikacyjne potwierdziły poprawność działania wszystkich bloków przetwarzania oraz stabilność systemu, co pozwala stwierdzić, że cel pracy został osiągnięty.

Zaprojektowana konstrukcja sprzętowa oraz architektura oprogramowania typu bare-metal spełniły postawione wymagania funkcjonalne i niefunkcjonalne. Wykorzystanie mechanizmu podwójnego buforowania oraz transferów DMA pozwoliło na osiągnięcie niskiej latencji przetwarzania przy zachowaniu jakości dźwięku standardu CD-Audio. Zaimplementowane algorytmy, w tym zaawansowane przetwarzanie splotowe oraz nieliniowe efekty typu waveshaping, działają zgodnie z założeniami teoretycznymi, co potwierdziły pomiary oscyloskopowe. Dzięki możliwościom kreowania złożonych charakterystyk brzmieniowych, system stanowi skuteczne i funkcjonalne narzędzie wspierające projektowanie audialnej warstwy interakcji w robotyce społecznej.

Projekt pozwala na różnorodne metody jego rozwoju. Istotnym usprawnieniem warstwy programowej byłaby migracja z obecnej architektury na system operacyjny czasu rzeczywistego. Ułatwiłoby to zarządzanie wątkami i skalowanie systemu. Moc obliczeniowa mikrokontrolera pozwala również na implementację algorytmu zmiany wysokości dźwięku (ang. *Pitch Shifter*), co znacząco poszerzyłoby zakres ekspresji wokalnej robota. Atrakcyjnym kierunkiem rozwoju jest także rozbudowa interfejsu użytkownika, w tym implementacja wizualizacji przebiegu fali akustycznej na wbudowanym wyświetlaczu w czasie rzeczywistym.

Literatura

[AFK+15] L. H. Arnal, A. Flinker, A. Kleinschmidt, A. L. Giraud, D. Poeppel, Human screams occupy a privileged niche in the communication soundscape. *Current Biology*, 25(15):2051–2056, 2015. PMC4562283.

[Ale] Marco Alessi. Czym jest kształtowanie fal w muzyce? https://emastered.com/pl/blog/waveshaping.

[ARMa] ARM. CMSIS. https://www.arm.com/technologies/cmsis.

[ARMb] ARM. CMSIS-DSP. https://arm-software.github.io/CMSIS_5/DSP/html/index.html.

[ARM14] ARM Limited, *ARM Cortex-M7 Processor Technical Reference Manual*, 2014. Revision r0p2.

[Aud] Słownik Audio. Kwantyzacja. https://audio.com.pl/vademecum/slownik/19583-kwantyzacja).

[Aud25] Unison Audio. Psychoacoustics 101: How to manipulate emotions with sound. https://unison.audio/psychoacoustics/, 2025.

[BJ05] Robert Bristow-Johnson. Cookbook formulae for audio eq biquad filter coefficients. Dostępny online: https://webaudio.github.io/Audio-EQ-Cookbook/audio-eq-cookbook.html, 2005.

[Bre02] Cynthia Breazeal, *Designing Sociable Robots*. 05 2002.

[Ele23] Electrosmith. Daisy: Embedded platform for music. https://www.electro-smith.com/daisy, 2023. Platforma audio oparta na STM32H750, następca rozwiązań bazujących na STM32F4.

[Gra23] A. Grau. Is silence a sound? RACO, https://www.raco.cat/index.php/JoSSIT/article/download/399154/508613, 2023.

[Haa49] Helmut Haas. Haas effect, 1949.

[HiF] HiFi.pl. Próbkowanie sygnału. https://www.hifi.pl/slownik/probkowanie.php).

[HKM+20] D. Hartanto, I. L. Kampmann, N. Morina, P. G. M. Emmelkamp, M. A. Neerincx, W. P. Brinkman, Moderators of social facilitation effect in virtual reality: Co-presence and realism of virtual agents. *Frontiers in Psychology*, 2020. PMID: 32612559.

[HS05] Cristy Ho, Charles Spence, Assessing the effectiveness of various auditory cues in capturing a driver's visual attention. *Journal of Experimental Psychology: Applied*, 11(3):157–174, 2005.

[Ins20] Mutable Instruments. Eurorack modules hardware designs. GitHub repository: https://github.com/pichenettes/eurorack, 2020. Moduły syntezatorowe (Braids, Rings, Elements) wykorzystujące STM32F4.

[iZo25] iZotope. Psychoacoustics: how perception influences music production. https://www.izotope.com/en/learn/psychoacoustics-how-perception-influences-music-production, 2025.

[JC24] D. O. Johnson, R. H. Cuijpers. Robots have been seen and not heard: Effects of consequential sounds on human-perception of robots. arXiv:2406.02938, https://arxiv.org/html/2406.02938v2, 2024.

[Jor25] Larry Jordan. Eq: Warm a voice and improve clarity. https://larryjordan.com/articles/eq-warm-a-voice-and-improve-diction/, 2025.

[Kat14] Bob Katz, *Mastering Audio: The Art and the Science*. Focal Press, Oxford, wydanie 3, 2014.

[KK23] S. Kumar, K. V. Kriegstein, Auditory roughness: a delicate balance. *Trends in Cognitive Sciences*, 2023. PMID: 37537119.

[LB05] Mike Lester, Jon Boley, The effects of latency on live sound monitoring. *Audio Engineering Society Convention Paper*, 2005. AES Convention 118.

[LDB22] L. Lünebach, K. Diwold, C. Bartneck. Designing sound for social robots: Candidate design principles. ResearchGate, https://www.researchgate.net/publication/361174970, 2022.

[LVK01] P. Larsson, D. Västfjäll, M. Kleiner. Effects of auditory information consistency and room acoustic cues on presence in virtual environments. ResearchGate, https://www.researchgate.net/publication/245524174, 2001.

[LZC25] Y. Liu, L. Zhang, Z. Chen. Robot character generation and adaptive human-robot interaction with personality shaping. arXiv:2503.15518, https://arxiv.org/html/2503.15518v1, 2025.

[MA17] S. Matsui, T. Arai, The effect of spectral tilt on size discrimination of voiced speech sounds. *Interspeech 2017*, 2017.

[Mag25] Composer Magazine. How reverb shapes space, time and emotion in cinema. https://composer.spitfireaudio.com/en/articles/how-reverb-shapes-space-time-and-emotion-in-cinema, 2025.

[ME96] Craig Marven, Gillian Ewers, *A Simple Approach to Digital Signal Processing*. Wiley-Interscience, New York, 1996.

[MSL+24] W. J. Mitchell, K. A. Szerszen, A. S. Lu, P. W. Schermerhorn, M. Scheutz, K. F. MacDorman, Deviation from typical organic voices best explains a vocal uncanny valley. *Cognition*, 2024.

[OACP18] R. Oliveira, P. Arriaga, F. Correia, A. Paiva. The sound or silence: Investigating the influence of robot noise on proxemics. ResearchGate, https://www.researchgate.net/publication/328834773, 2018.

[Ody] Piotr Odyja. Technologie multimedialne: Formaty dźwięku. Katedra Systemów Multimedialnych, Politechnika Gdańska 2025.

[OGL23] N. Ott, J. George, C. Lutteroth. Experiencing reconstructed reality: The perception of visual-acoustic properties. HCI Group, University of Otago, https://hci.otago.ac.nz/papers/OttACMOZCHI2023.pdf, 2023.

[Ott88] Henry W. Ott, *Noise Reduction Techniques in Electronic Systems*. Wiley-Interscience, New York, wydanie 2, 1988.

[PAA20] E. Ponsot, P. Arias, J. J. Aucouturier. Sound context modulates perceived vocal emotion. bioRxiv, https://www.biorxiv.org/content/10.1101/2020.01.08.898205v1, 2020.

[Phi96] Philips Semiconductors, *I2S Bus Specification*, 1996. Revised 1996 June 5.

[PRC23] I. Phillips, R. Ryskin, M. A. Cohen, The perception of silence. *Proceedings of the National Academy of Sciences*, 120(29), 2023.

[Rep25] MUsic Technology Online Repository. Unit 3: Fundamentals of psychoacoustics. https://mutor-2.github.io/ScienceOfMusic/units/03/, 2025.

[Rob] Encyklopedia Robotów. Kismet. https://robotsguide.com/robots/kismet.

[Sch80] Martin Schetzen, Nonlinear system modeling based on the wiener theory. *Proceedings of the IEEE*, 69(12):1557–1573, 1980.

[Sel13] Douglas Self, *Audio Power Amplifier Design*. Focal Press, Oxford, wydanie 6, 2013.

[Sha49] C. E. Shannon, Communication in the presence of noise. *Proceedings of the IRE (Institute of Radio Engineers)*, strony 10–21, 1949.

[Siu24] A. F. Siu. Sonification of motion of robotic systems with many degrees-of-freedom. https://deepblue.lib.umich.edu/handle/2027.42/197377, 2024.

[SK20] R. Stower, A. Kappas. Discovering the uncanny valley for the sound of a voice. Tilburg University, http://arno.uvt.nl/show.cgi?fid=149554, 2020.

[Smi07] Julius O. Smith, *Introduction to Digital Filters with Audio Applications*. W3K Publishing, Stanford, CA, 2007.

[Ste20] J. Stevens, *Gender Perception Dependent on Fundamental Frequency, Source Spectral Tilt, and Formant Frequencies*. Praca magisterska, Bowling Green State University, 2020.

[STM21] STMicroelectronics, *STM32H753 Reference Manual*, 2021. RM0433.

[STM23] STMicroelectronics, *General-purpose DMA controller (DMA) for STM32 devices*, 2023. AN4031.

[STM25] STMicroelectronics. STM32CubeMX: Graficzne narzędzie dla mikrokontrolerów STM32. https://www.st.com/content/st_com/en/stm32cubemx.html, 2025. Accessed: 2025-12-12.

[TMSM+13] N. Tye-Murray, B. Spehar, J. Myerson, S. Hale, M. Sommers, Gated audiovisual speech identification in silence vs. noise: effects on time and accuracy. *Frontiers in Psychology*, 2013. PMC3685792.

[Val12] Jonathan W. Valvano, *Embedded Microcomputer Systems: Real Time Interfacing*. Cengage Learning, Stamford, CT, wydanie 3, 2012.

[VDAP87] J. N. Van Dijkhuizen, P. C. Anema, R. Plomp, Two-state compression of spectral tilt: individual differences and psychoacoustical limitations to the benefit from compression. *Journal of the Acoustical Society of America*, 83(2):670–679, 1987. PMID: 3430377.

[VSEB22] S. T. Völkel, C. Schneegass, M. Eiband, D. Buschek. Empathy in human-robot interaction: Designing for social robots. ResearchGate, https://www.researchgate.net/publication/358443808, 2022.

[Wika] Wikipedia. Aliasing (przetwarzanie sygnałów). https://pl.wikipedia.org/wiki/Aliasing_(przetwarzanie_sygnaÅĆÃşw).

[Wikb] Wikipedia. DMIPS. https://pl.wikipedia.org/wiki/Dhrystone.

[Wikc] Wikipedia. Equalizer. https://en.wikipedia.org/wiki/Audio_equalization.

[Wikd] Wikipedia. Fale akustyczne. https://pl.wikipedia.org/wiki/Fale_akustyczne.

[Wike] Wikipedia. Linia opóźniająca. https://en.wikipedia.org/wiki/Digital_delay_line.

[Wikf] Wikipedia. PCM. https://pl.wikipedia.org/wiki/PCM.

[Wikg] Wikipedia. Przetwornik analogowo-cyfrowy. https://pl.wikipedia.org/wiki/Przetwornik_analogowo-cyfrowy.

[Wikh] Wikipedia. Sygnał. https://pl.wikipedia.org/wiki/SygnaÅĆ.

[Wiki] Wikipedia. Sygnał analogowy. https://pl.wikipedia.org/wiki/SygnaÅĆ_analogowy.

[Wikj] Wikipedia. Sygnał cyfrowy. https://pl.wikipedia.org/wiki/SygnaÅĆ_cyfrowy.

[WK25] S. Werner, F. Klein. Effects of auditory distance cues and reverberation on spatial perception and listening strategies. arXiv:2505.18020, https://arxiv.org/html/2505.18020v1, 2025.

[ZF23] Brian Zhang, Naomi Fitter, Nonverbal sound in human-robot interaction: A systematic review. *ACM Transactions on Human-Robot Interaction*, 12, 02 2023.

[Zie14] Tomasz P. Zieliński, *Cyfrowe przetwarzanie sygnałów: od teorii do zastosowań*. Wydawnictwa Komunikacji i Łączności WKŁ, Warszawa, 2014.

[Zöl11] Udo Zölzer, redaktor, *DAFX: Digital Audio Effects*. Wiley, Chichester, wydanie 2, 2011.

Spis rysunków

1.1 Robot Kismet przeglądający się w lustrze [Rob] 4

3.1 Sygnał analogowy (szary) i przykładowe przekształcenie na sygnał cyfrowy (czerwony) [Wikj] 9
3.2 Przepływ sygnału w korektorze trójpasmowym 12
3.3 Ogólny algorytm filtru BiQuad 12
3.4 Przetwarzanie próbki w bramce szumów 15
3.5 Przetwarzanie próbki w efekcie Overdrive 17
3.6 Przetwarzanie próbki w efekcie Fuzz 18
3.7 Przetwarzanie próbki w efekcie Distortion 19
3.8 Przetwarzanie próbki w efekcie Delay 20
3.9 Przetwarzanie próbki w efekcie Chorus 22
3.10 Przetwarzanie próbki w efekcie przetwarzania splotowego 23

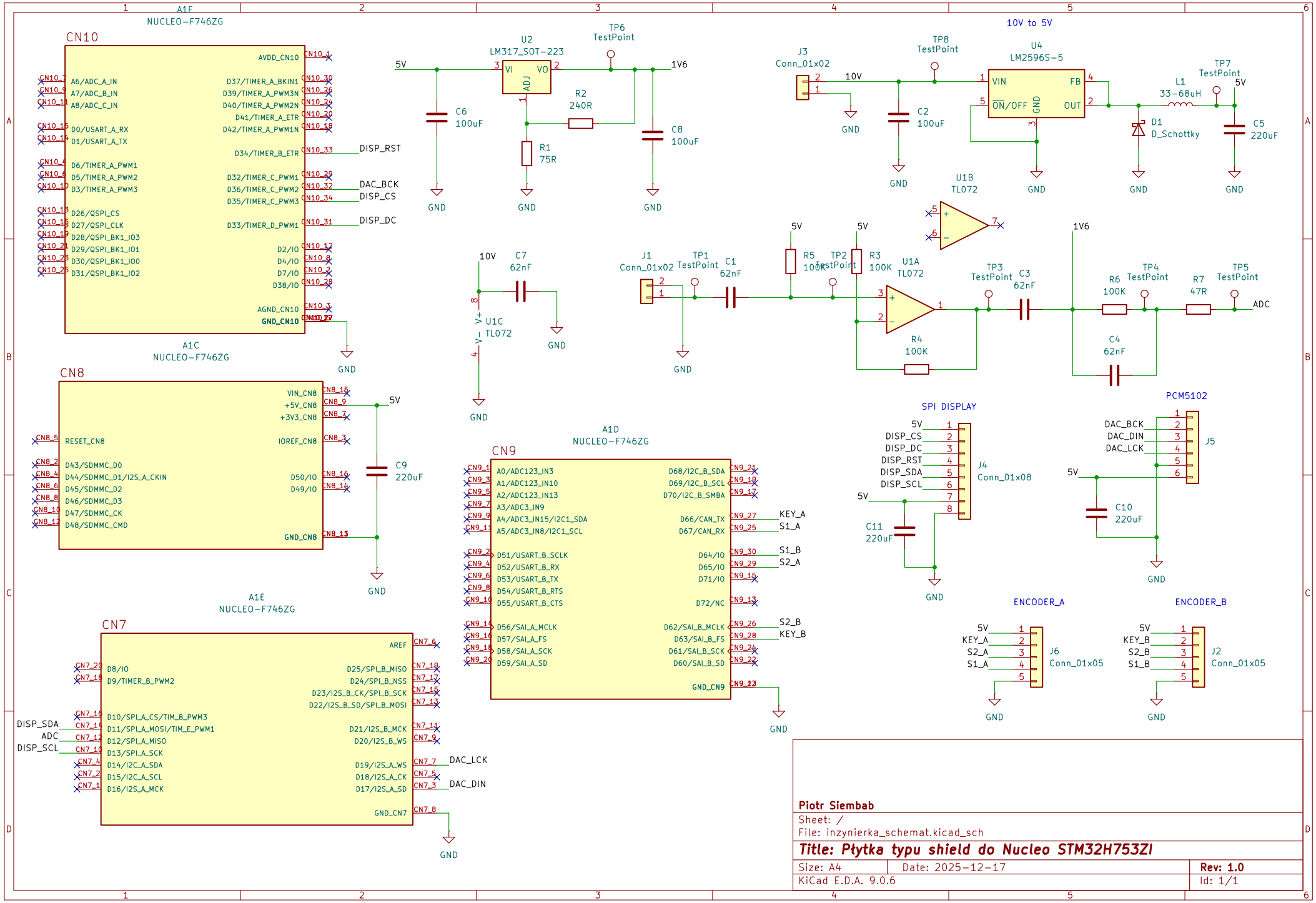
4.1 Schemat blokowy systemu 28
4.2 Konfiguracja wyprowadzeń mikrokontrolera STM32H753 w środowisku STM32CubeMX 30
4.3 Schemat blokowy architektury oprogramowania i przepływu danych w systemie. 32

5.1 Przebiegi sygnału sinusoidalnego 1000 Hz w kluczowych punktach systemu, ... 35
5.2 Przetwarzanie z efektem Overdrive, ... 36
5.3 Przetwarzanie z efektem Distortion, ... 36
5.4 Przetwarzanie z efektem Fuzz, ... 36
5.5 Korektor niskotonowy, ... 37
5.6 Korektor środkowopasmowy, ... 38
5.7 Korektor wysokotonowy, ... 38
5.8 Zachowanie bramki szumów w stanie otwartym, ... 39
5.9 Zachowanie bramki szumów w stanie zamkniętym, ... 39
5.10 Test efektu Delay, ... 40
5.11 Test efektu Chorus, ... 41
5.12 Splot z IR „Puppet” (Master of Puppets), ... 42
5.13 Splot z IR „Killem” (Kill ’Em All), ... 42

B.1 Płytka założona na Nucleo STM32H753ZI 51
B.2 Widok płytki z góry 52
B.3 Widok płytki od dołu 53

Dodatek A

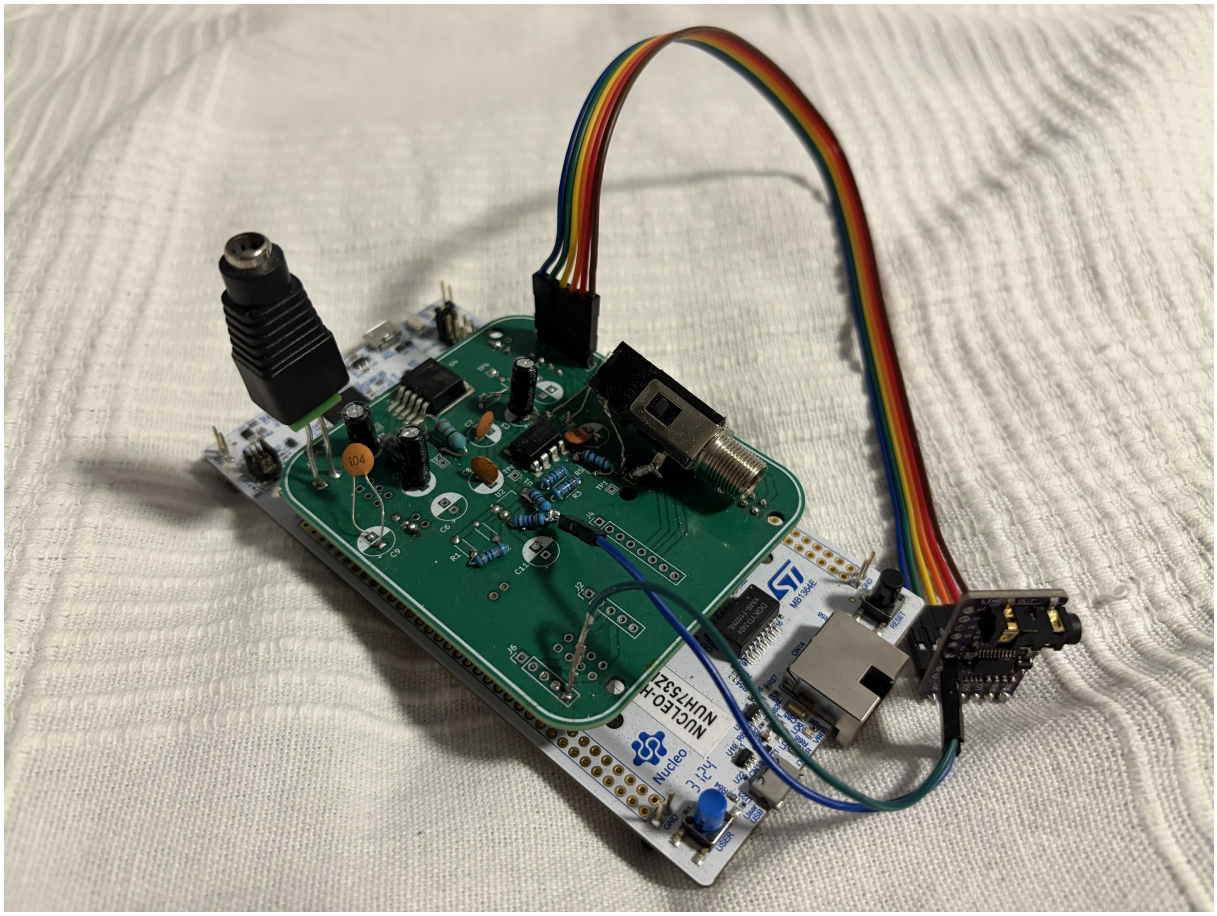
Schemat układu elektronicznego


A1F
NUCLEO-F746ZG
CN10
AVDD_CN10
A6/ADC_A_IN
A7/ADC_B_IN
A8/ADC_C_IN
D37/TIMER_A_BKIN1
D39/TIMER_A_PWM3N
D40/TIMER_A_PWM2N
D41/TIMER_A_ETR
D42/TIMER_A_PWM1N
D0/USART_A_RX
D1/USART_A_TX
D34/TIMER_B_ETR
DISP_RST
D6/TIMER_A_PWM1
D5/TIMER_A_PWM2
D3/TIMER_A_PWM3
D32/TIMER_C_PWM1
DAC_BCK
D36/TIMER_C_PWM2
DISP_CS
D35/TIMER_C_PWM3
D26/QSPI_CS
D27/QSPI_CLK
D28/QSPI_BK1_IO3
DISP_DC
D33/TIMER_D_PWM1
D29/QSPI_BK1_IO1
D30/QSPI_BK1_IO0
D31/QSPI_BK1_IO2
D2/IO
D4/IO
D7/IO
D38/IO
AGND_CN10
GND_CN10
A1C
NUCLEO-F746ZG
CN8
VIN_CN8
+5V_CN8
+3V3_CN8
RESET_CN8
IOREF_CN8
D43/SDMMC_D0
D44/SDMMC_D1/I2S_A_CKIN
D45/SDMMC_D2
D46/SDMMC_D3
D47/SDMMC_CK
D48/SDMMC_CMD
D50/IO
D49/IO
GND_CN8
A1E
NUCLEO-F746ZG
CN7
AREF
D8/IO
D9/TIMER_B_PWM2
D25/SPI_B_MISO
D24/SPI_B_NSS
D23/I2S_B_CK/SPI_B_SCK
D22/I2S_B_SD/SPI_B_MOSI
D10/SPI_A_CS/TIM_B_PWM3
D11/SPI_A_MOSI/TIM_E_PWM1
D12/SPI_A_MISO
D13/SPI_A_SCK
D21/I2S_B_MCK
D20/I2S_B_WS
D14/I2C_A_SDA
D15/I2C_A_SCL
D16/I2S_A_MCK
D19/I2S_A_WS
D18/I2S_A_CK
D17/I2S_A_SD
DAC_LCK
DAC_DIN
GND_CN7
A1D
NUCLEO-F746ZG
CN9
A0/ADC123_IN3
A1/ADC123_IN10
A2/ADC123_IN13
A3/ADC3_IN9
A4/ADC3_IN15/I2C1_SDA
A5/ADC3_IN8/I2C1_SCL
D68/I2C_B_SDA
D69/I2C_B_SCL
D70/I2C_B_SMBA
D66/CAN_TX
D67/CAN_RX
KEY_A
S1_A
D51/USART_B_SCLK
D52/USART_B_RX
D53/USART_B_TX
D54/USART_B_RTS
D55/USART_B_CTS
D64/IO
D65/IO
D71/IO
S1_B
S2_A
D72/NC
D56/SAI_A_MCLK
D57/SAI_A_FS
D58/SAI_A_SCK
D59/SAI_A_SD
D62/SAI_B_MCLK
D63/SAI_B_FS
D61/SAI_B_SCK
D60/SAI_B_SD
S2_B
KEY_B
GND_CN9
10V to 5V
U2
LM317_SOT-223
TP6
TestPoint
5V
1V6
C6
100uF
R2
240R
R1
75R
C8
100uF
J3
Conn_01x02
TP8
TestPoint
U4
LM2596S-5
10V
VIN
FB
ON/OFF
GND
OUT
C2
100uF
L1
33-68uH
TP7
TestPoint
D1
D_Schottky
C5
220uF
U1B
TL072
U1C
TL072
C7
62nF
J1
Conn_01x02
TP1
TestPoint
C1
62nF
R5
100K
TP2
TestPoint
R3
100K
U1A
TL072
R4
100K
TP3
TestPoint
C3
62nF
R6
100K
TP4
TestPoint
R7
47R
TP5
TestPoint
ADC
C4
62nF
C9
220uF
SPI DISPLAY
J4
Conn_01x08
DISP_CS
DISP_DC
DISP_RST
DISP_SDA
DISP_SCL
C11
220uF
PCM5102
J5
DAC_BCK
DAC_DIN
DAC_LCK
C10
220uF
ENCODER_A
J6
Conn_01x05
KEY_A
S2_A
S1_A
ENCODER_B
J2
Conn_01x05
KEY_B
S2_B
S1_B
GND
Piotr Siembab
Sheet: /
File: inzynierka_schemat.kicad_sch
Title: Płytka typu shield do Nucleo STM32H753ZI
Size: A4
Date: 2025-12-17
Rev: 1.0
KiCad E.D.A. 9.0.6
Id: 1/1

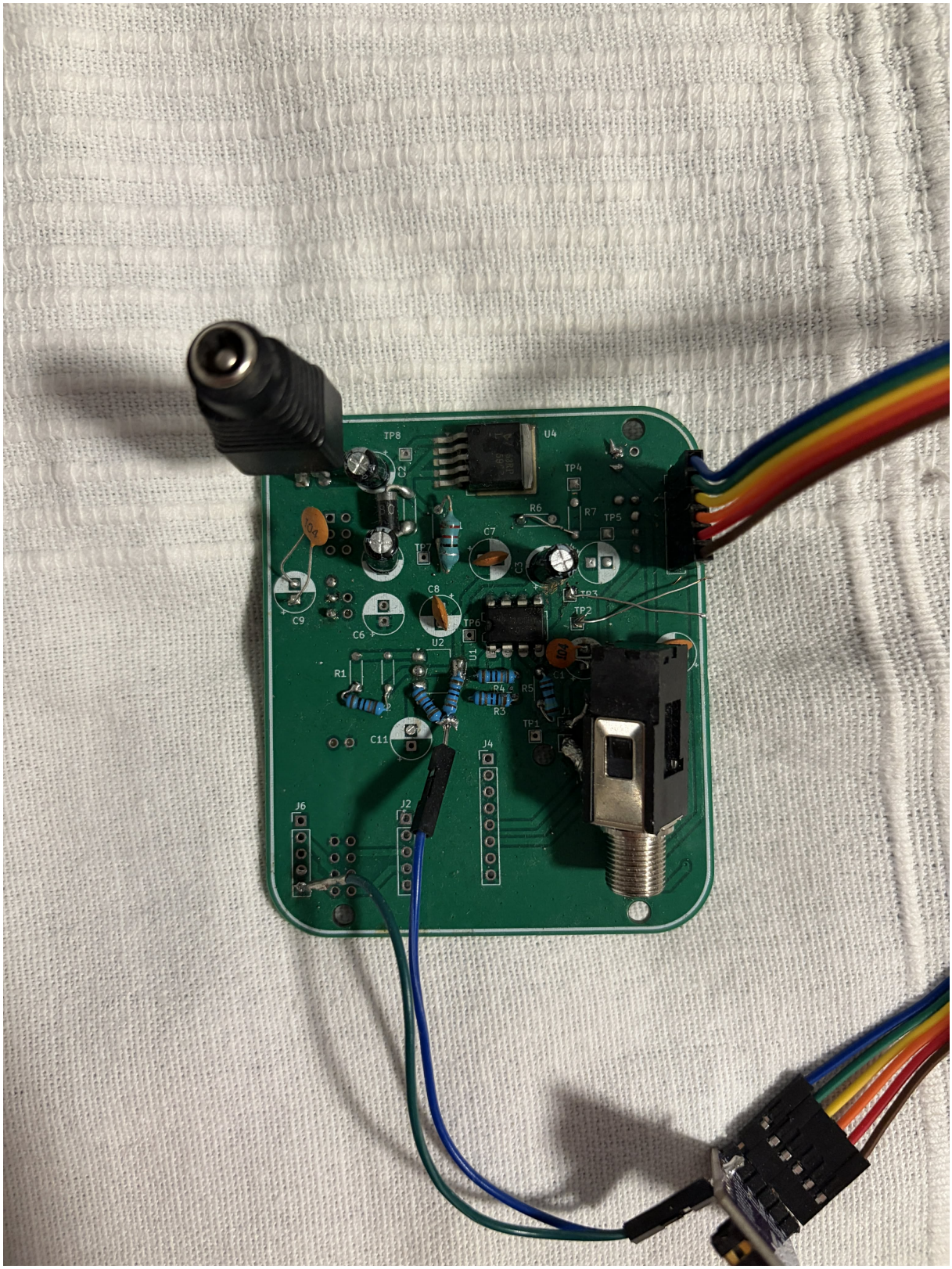
Dodatek B

Zdjęcia Projektu

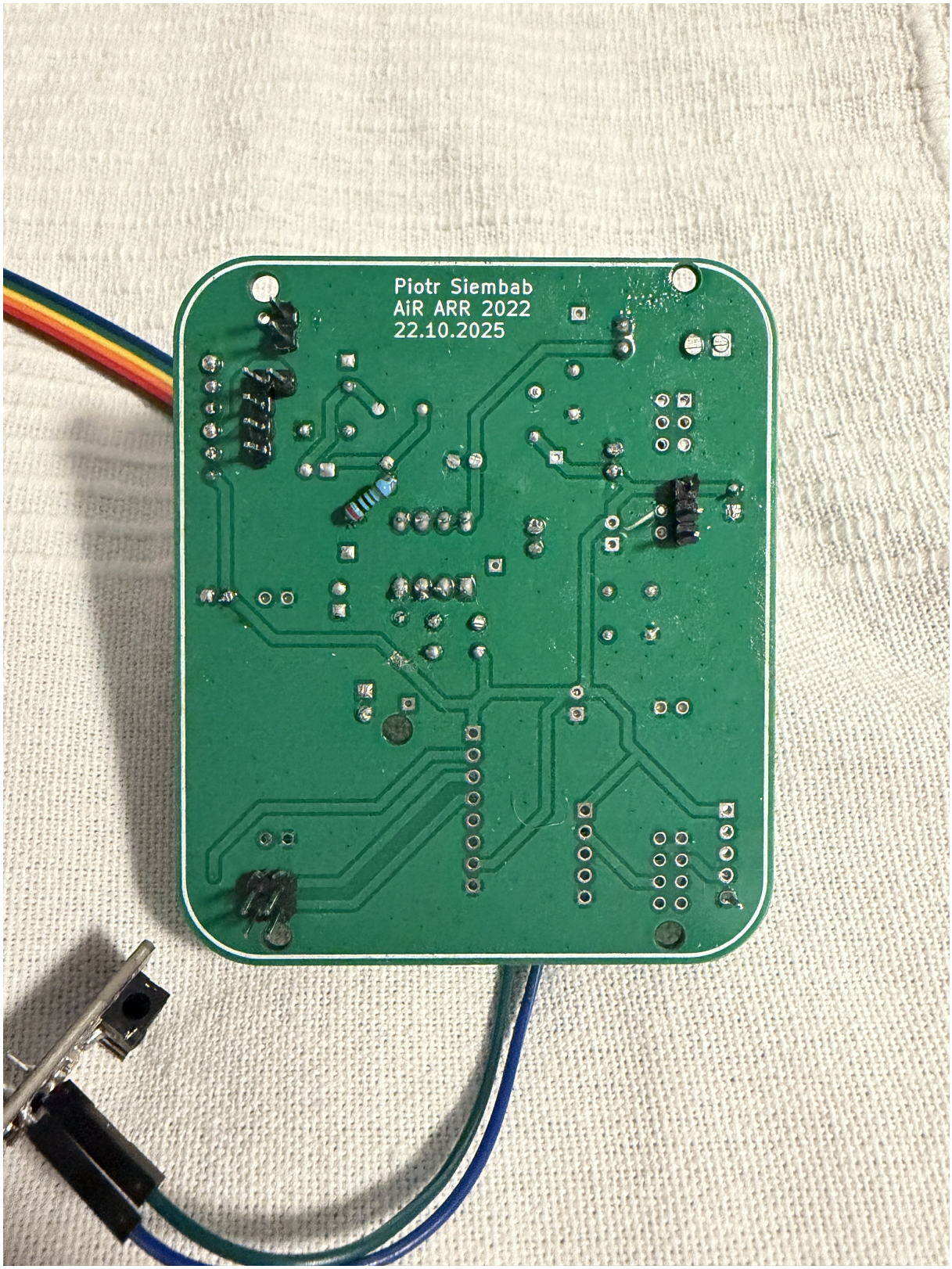
Poniższy dodatek zawiera zdjęcia projektu w formie płytki rozszerzającej typu „Shield" bez enkoderów oraz wyświetlacza. Tymczasowe pozbycie się ich miało na celu minimalizację występujących szumów i zapewnienie idealnych warunków do przeprowadzenia pomiarów oscyloskopem.



Rysunek B.1: Płytka założona na Nucleo STM32H753ZI



Rysunek B.2: Widok płytki z góry



Rysunek B.3: Widok płytki od dołu