ĐẠI HỌC HUẾ

**TRƯỜNG ĐẠI HỌC KHOA HỌC**

**KHOA ĐIỆN TỬ VIỄN THÔNG**

**Lê Trần Quốc Đạt**

**THIẾT KẾ HỆ THỐNG THU PHÁT TÍN HIỆU**

**BĂNG TẦN TIẾNG NÓI DỰA TRÊN NỀN TẢNG**

**PHÁT TRIỂN ARM CORTEX-M4F**

**Ngành: Công nghệ, kỹ thuật điện tử - truyền thông**

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**NGƯỜI HƯỚNG DẪN**

**ThS. NGUYỄN VĂN ÂN**

Huế - 2014

LỜI CẢM ƠN

Đầu tiên em muốn được gởi lời cảm ơn sâu sắc nhất đến ThS. Nguyễn Văn Ân, người hướng dẫn của em. Trước khi bắt tay vào làm đồ án, em đã có dịp được làm việc cùng thầy. Thời gian đó, em đã học thêm được rất nhiều kiến thức và cả kinh nghiệm thực tế, mà sau này chính những bài học kia là nguồn cảm hứng để em quyết định theo đuổi đề tài này. Quá trình làm đồ án, em cũng đã nhận được những hướng dẫn rất tận tình và tỉ mỉ từ thầy. Đây là điều mà em trân trọng nhất.

Em cũng muốn gởi lời cảm ơn chân thành đến tập thể giảng viên Khoa Điện tử - Viễn thông, Đại học Khoa học, Đại học Huế. Bằng kinh nghiệm và lòng nhiệt tình, các thầy các cô đã truyền cho chúng em những kiến thức và trải nghiệm quý giá trong suốt cuộc đời sinh viên của mình.

Ngoài ra, không thể không nhắc đến trong lời cảm ơn này là những người bạn trong nhóm bạn tiểu đội A35-D. Cảm ơn các bạn, những buổi trò chuyện vui vẻ và thoái mái với các bạn đã tăng thêm sức lực cho mình sau những giờ làm việc căng thẳng.

Và cuối cùng, con muốn gởi lời cảm ơn của con đến ba mẹ. Ba mẹ là chỗ dựa vững chắc giúp con yên tâm tập trung sức lực để hoàn thành đồ án này.

***Lê Trần Quốc Đạt***

DANH MỤC HÌNH ẢNH

BẢNG CÁC TỪ VIẾT TẮT

[MỞ ĐẦU 1](#_Toc389680129)

[CHƯƠNG 1: CÁC LÝ THUYẾT LIÊN QUAN 2](#_Toc389680130)

[1.1. Cơ bản về thiết kế máy thu phát tín hiệu băng tần tiếng nói: 2](#_Toc389680131)

[1.2. Số hóa tín hiệu băng tần tiếng nói 2](#_Toc389680132)

[1.2.1. Tín hiệu băng tần tiếng nói: 2](#_Toc389680133)

[1.2.2. Số hóa tín hiệu 3](#_Toc389680134)

[1.2.3. Lấy mẫu đa tốc độ: 4](#_Toc389680135)

[1.3. Bộ lọc số lọc thông thấp 4](#_Toc389680136)

[1.3.1. Ưu điểm của các bộ lọc số: 4](#_Toc389680137)

[1.3.2. Bộ lọc số lọc thông thấp: 5](#_Toc389680138)

[1.3.3. Phương pháp thiết kế bộ lọc số lọc thông thấp có đáp ứng xung chiều dài hữu hạn (FIR) 6](#_Toc389680139)

[1.4. Nền tảng phát triển ARM Cortex-M4F 9](#_Toc389680140)

[1.4.1. ARM Cortex-M4F 9](#_Toc389680141)

[1.4.2. Đơn vị dấu phẩy động – Floating Point Unit (FPU) 10](#_Toc389680142)

[1.4.3. Chuẩn giao tiếp phần mềm vi điều khiển Cortex (Cortex Microntroller Software Interface Standard – CMSIS) 10](#_Toc389680143)

[CHƯƠNG 2: THIẾT KẾ PHẦN CỨNG 12](#_Toc389680144)

[2.1. Sơ đồ kết nối phần cứng 12](#_Toc389680145)

[2.2. Nguồn nuôi 12](#_Toc389680146)

[2.3. Mạch tiền khuếch đại và mạch lọc khử chồng phổ AAF 13](#_Toc389680147)

[2.3.1. Electret Condenser Microphone (ECM) 13](#_Toc389680148)

[2.3.2. Mạch tiền khuếch đại và mạch lọc khử chồng phổ AAF 14](#_Toc389680149)

[2.4. Mạch tái tạo tín hiệu đầu ra: 15](#_Toc389680150)

[2.5. Mạch điều khiển chính - Kit phát triển Tiva C Launchpad nền tảng ARM Cortex-M4F 16](#_Toc389680151)

[CHƯƠNG 3: THIẾT KẾ PHẦN MỀM 19](#_Toc389680152)

[3.1. Tổng quan 19](#_Toc389680153)

[3.2. Lấy mẫu tín hiệu với bộ chuyển đổi A/D 20](#_Toc389680154)

[3.2.1. Bộ chuyển đổi A/D trên TM4C123GH6PM 20](#_Toc389680155)

[3.2.2. Kĩ thuật lấy mẫu tín hiệu theo chu kì 20](#_Toc389680156)

[3.3. Kỹ thuật xử lí tín hiệu đa tốc độ lấy mẫu (Multirate Signal Processing) 21](#_Toc389680157)

[3.3.1. Hạn chế trong việc thiết kế bộ lọc tích cực cho AAF 21](#_Toc389680158)

[3.3.2. Cơ sở kĩ thuật xử lí đa tốc độ lấy mẫu: 21](#_Toc389680159)

[3.3.3. Thuật toán cho bộ Decimator: 22](#_Toc389680160)

[3.4. Hệ thống quản lí file theo chuẩn FAT 22](#_Toc389680161)

[3.4.1. Đặc điểm của FAT 22](#_Toc389680162)

[3.4.2. Thư viện FatFs 23](#_Toc389680163)

[3.4.3. Ứng dụng thư viện FatFs để triển khai FAT32 trên thẻ nhớ SD 24](#_Toc389680164)

[3.4.4. Lưu trữ file dưới dạng WAVE 25](#_Toc389680165)

[3.5. Bộ đệm dữ liệu kiểu Vào trước ra trước (FIFO) 26](#_Toc389680166)

[3.5.1. Sự cần thiết của bộ đệm dữ liệu FIFO 26](#_Toc389680167)

[3.5.2. Thiết kế bộ đệm FIFO 26](#_Toc389680168)

[3.5.3. Bộ đệm FIFO với μDMA cơ chế Ping-Pong 28](#_Toc389680169)

[3.6. Bộ lọc thông thấp đáp ứng xung chiều dài hữu hạn (FIR) 30](#_Toc389680170)

[3.6.1. Thiết kế nhân bộ lọc h[i] với Scipy: 30](#_Toc389680171)

[3.6.2. Mô phỏng bộ lọc đã thiết kế trên Scipy 31](#_Toc389680172)

[3.6.3. Nhúng bộ lọc vào phần mềm điều khiển thông qua CMSIS 32](#_Toc389680173)

[3.7. Chuyển đổi số - tương tự D/A sử dụng kĩ thuật điều chế độ rộng xung PWM 33](#_Toc389680174)

[CHƯƠNG 4: MỘT SỐ KẾT QUẢ VÀ THẢO LUẬN 34](#_Toc389680175)

[4.1. Phần cứng và giao diện phần mềm: 34](#_Toc389680176)

[4.2. Đặc tuyến của các mạch tích cực 36](#_Toc389680177)

[4.2.1. Đặc tuyến biên độ của mạch tiền khuếch đại 36](#_Toc389680178)

[4.2.2. Đặc tuyến biên – tần của mạch khử chồng phổ AAF 37](#_Toc389680179)

[4.3. Hoạt động thực tế của hệ thống 37](#_Toc389680180)

[4.3.1. Tốc độ lấy mẫu và tín hiệu PWM 37](#_Toc389680181)

[4.3.1. Kiểm tra với các tín hiệu có tham số khác nhau: 37](#_Toc389680182)

[4.3.2. Hoạt động của hệ thống với tín hiệu và tạp âm: 1](#_Toc389680183)

[KẾT LUẬN VÀ ĐỀ XUẤT 2](#_Toc389680184)

PHỤ LỤC I – NHÂN BỘ LỌC THÔNG THẤP FIR TRONG THIẾT KẾ

DANH MỤC HÌNH ẢNH

[Hình 1.1 – Sơ đồ khối một thiết kế điển hình máy thu phát tín hiệu băng tần tiếng nói 2](#_Toc389680964)

[Hình 1.2 – Tín hiệu được mã hóa dạng PCM 3](#_Toc389680965)

[Hình 1.3 – Đáp ứng tần số của bộ lọc thông thấp FIR 40001 điểm 5](#_Toc389680966)

[Hình 1.4 – Đáp ứng tần số cơ bản của một bộ lọc số 6](#_Toc389680967)

[Hình 1.5 – Bộ lọc đáp ứng xung chiều dài hữu hạn FIR 7](#_Toc389680968)

[Hình 1.6 – Đáp ứng lý tưởng của bộ lọc thông thấp 7](#_Toc389680969)

[Hình 1.7 – Phương pháp cửa sổ Sinc 7](#_Toc389680970)

[Hình 1.8 – Đáp ứng khi dùng cửa sổ Hamming và Blackman 8](#_Toc389680971)

[Hình 1.9 – Đáp ứng xung bộ lọc FIR cửa số Blackman với N = 66 8](#_Toc389680972)

[Hình 1.10 – Cấu trúc cơ bản của kiến trúc ARM Cortex-M4F 9](#_Toc389680973)

[Hình 1.11 – Cấu trúc của CMSIS 11](#_Toc389680974)

[Hình 2.1 – Sơ đồ kết nối phần cứng 12](#_Toc389680975)

[Hình 2.2 – Sơ đồ nguyên lí mạch nguồn 13](#_Toc389680976)

[Hình 2.3 – Cấu hình cơ bản của ECM 13](#_Toc389680977)

[Hình 2.4 – Mạch tiền khuếch đại và lọc khử chồng phổ 14](#_Toc389680978)

[Hình 2.5 – Đáp ứng tần số mạch lọc khử chồng phổ AAF 15](#_Toc389680979)

[Hình 2.6 – Mạch tái tạo tín hiệu đầu ra 15](#_Toc389680980)

[Hình 2.7 – Đáp ứng bộ lọc thông thấp mạch tái tạo tín hiệu đầu ra 16](#_Toc389680981)

[Hình 2.8 – Sơ đồ khối Tiva C Launchpad 17](#_Toc389680982)

[Hình 2.9 – Sơ đồ kết nối các chân tín hiệu của Tiva C Launchpad trên mạch 17](#_Toc389680983)

[Hình 3.1 – Cấu trúc thiết kế phần mềm hệ thống 18](#_Toc389680984)

[Hình 3.2 – Kĩ thuật xử lí đa tốc độ lấy mẫu 20](file:///E:\DOAN_SPRING_2014\WRITING%20THESIS\draft.docx#_Toc389680985)

[Hình 3.3 – Cấu trúc của một phân vùng sử dụng FAT 22](#_Toc389680986)

[Hình 3.4 – Ví trí lớp thư viện FatFs trong các lớp của phần mềm 23](#_Toc389680987)

[Hình 3.5 – Cấu hình cơ bản của hệ thống nhúng với FatFs 23](#_Toc389680988)

[Hình 3.6 – Cấu trúc file dạng WAVE 24](#_Toc389680989)

[Hình 3.7 – Cấu trúc bộ đệm FIFO với 6 phần tử 25](#_Toc389680990)

[Hình 3.8 – Sơ đồ thuật toán yêu cầu phần tử trống từ bộ đệm FIFO 26](#_Toc389680991)

[Hình 3.9 – Sơ đồ thuật toán yêu cầu phần tử chứa dữ liệu từ bộ đệm FIFO 27](#_Toc389680992)

[Hình 3.10 – 𝜇DMA trạng thái sử dụng bộ đệm Ping 28](#_Toc389680993)

[Hình 3.11 – 𝜇DMA trạng thái sử dụng bộ đệm Pong 28](#_Toc389680994)

[Hình 3.12 – 𝜇DMA trở về trạng thái sử dụng bộ đệm Ping 28](#_Toc389680995)

[Hình 3.13 – Đáp ứng tần số của bộ lọc FIR 215 điểm 30](#_Toc389680996)

[Hình 3.14 – Bộ lọc với tín hiệu tổng hợp 100 Hz và 4100 Hz 30](#_Toc389680997)

[Hình 3.15 – Bộ lọc với tín hiệu tổng hợp 3200 Hz và 4050 Hz 31](#_Toc389680998)

[Hình 3.16 – Kĩ thuật PWM D/A 32](#_Toc389680999)

[Hình 4.1 – Mạch kết nối với Kit Tiva C Launchpad 33](#_Toc389681000)

[Hình 4.2 – Kết nối mạch với Kit Tiva C Launchpad 33](#_Toc389681001)

[Hình 4.3 – Giao diện dòng lệnh qua UART 34](#_Toc389681002)

[Hình 4.4 – Đặc tuyến biên độ mạch tiền khuếch đại 35](#_Toc389681003)

[Hình 4.5 – Đặc tuyến biên – tần mạch khử chồng phổ AAF 35](#_Toc389681004)

[Hình 4.6 – Quá trình lấy mẫu tín hiệu 36](#_Toc389681005)

[Hình 4.7 – Tín hiệu PWM 37](#_Toc389681006)

[Hình 4.8 – Tín hiệu sine 500 Hz 37](#_Toc389681007)

[Hình 4.9 – Tín hiệu sine 1000 Hz 37](#_Toc389681008)

[Hình 4.10 – Tín hiệu sine 1500 Hz 37](#_Toc389681009)

[Hình 4.11 – Tín hiệu sine 2000 Hz 37](#_Toc389681010)

[Hình 4.12 – Hoạt động của hệ thống với tín hiệu và tạp âm 1](#_Toc389681011)

[Hình 4.13 – Nhân bộ lọc thông thấp FIR trong thiết kế 4](#_Toc389681012)

BẢNG CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Chữ viết tắt** | **Nguyên bản tiếng Anh** | **Dịch nghĩa tiếng Việt** |
| AAF | Anti-Aliasing Filter | Lọc khử chồng phổ |
| ADC | Analog to Digital Conveter | Bộ chuyển đổi tín hiệu tương tự sang số |
| CMSIS | Cortex Microcontroller Software Interface Standard | Chuẩn giao tiếp phần mềm vi điều khiển Cortex |
| DAC | Digital to Analog Converter | Bộ chuyển đổi tín hiệu số sang tương tự |
| DMA | Direct Memory Access | Truy xuất bộ nhớ trực tiếp |
| DSP | Digital Signal Processing | Xử lí số tín hiệu |
| ECM | Electret Condenser Microphone | Micrô dạng điện dung biến đổi |
| FAT | File Allocation Table | Bảng cấp phát tập tin |
| FIFO | First In First Out | Vào trước – Ra trước |
| FIR | Finite Impulse Response | Đáp ứng xung chiều dài hữu hạn |
| FPU | Floating-Point Unit | Đơn vị dấu phẩy động |
| IIR | Infinite Impulse Response | Đáp ứng xung chiều dài vô hạn |
| PWM | Pulse Width Modulation | Kĩ thuật điều chế độ rộng xung |

MỞ ĐẦU

Nghiên cứu về thu phát và lưu trữ tín hiệu băng tần tiếng nói là cơ sở cho các tìm hiểu chuyên sâu về các ứng dụng liên quan đến lĩnh vực xử lí tín hiệu băng tần tiếng nói. Một trong những hướng nghiên cứu chuyên sâu nổi bật hiện nay đó là nén tín hiệu tiếng nói trong các ứng dụng truyền âm thanh thoại qua mạng IP – Voice over Internet Protocol (VoIP). Năm 2009, Skype đã ra mắt chuẩn nén tín hiệu âm thanh thoại mới mang tên SILK, cho phép chất lượng thoại được nâng cao trên cả băng hẹp và băng rộng, với lượng dữ liệu di động được sử dụng ở mức tối ưu [1].

Hiện nay, việc xử lí tín hiệu bằng các kĩ thuật xử lí số tín hiệu (Digital Signal Processing - DSP) đã trở thành xu hướng chủ đạo trong các thiết kế bởi hiệu năng và tính linh động cao. Thông thường, xử lí số tín hiệu phải sử dụng các lõi chuyên biệt gọi là Bộ xử lí số tín hiệu (DSP Processor) với giá thành cao và khó tiếp cận. Tuy nhiên, nhờ sự phát triển vượt bậc của công nghệ bán dẫn trong những năm trở lại đây, tốc độ xử lí của vi điều khiển, vốn chỉ dành cho các ứng dụng nhỏ yêu cầu tốc độ thấp, đã được nâng cao rất nhiều, đủ khả năng để đáp ứng các ứng dụng nặng về tính toán như DSP. Cùng với đó, năm 2010, ARM Holdings công bố kiến trúc nhân xử lí ARM Cortex-M4F với các chỉ lệnh DSP mở rộng cùng với Đơn vị dấu phẩy động (FPU) giúp tối ưu hóa cho các thao tác DSP [2]. Điều này đã mở ra hướng phát triển mới cho các thiết bị xử lí tín hiệu có giá thành thấp hơn và dễ dàng tiếp cận hơn.

Với mong muốn nắm bắt các kĩ thuật nền tảng trong xử lí tín hiệu cũng như tạo bước đệm cho các nghiên cứu sâu hơn về lĩnh vực xử lí tín hiệu băng tần tiếng nói trong tương lai, đề tài của đồ án được chọn là ***“Thiết kế hệ thống thu phát tín hiệu băng tần tiếng nói dựa trên nền tảng phát triển ARM Cortex-M4F”.***

Mục đích của đồ án là thiết kế phần cứng và phần mềm cho một hệ thống thu phát tín hiệu băng tần tiếng nói với nền tảng phát triển ARM Cortex-M4F. Thông qua các nghiên cứu lý thuyết, mô phỏng trên máy tính và các kiểm tra thực nghiệm, đánh giá chất lượng của thiết kế và đề xuất những hướng phát triển cao hơn.

Nội dung của đồ án bao gồm:

Chương 1: Điểm lại một số lý thuyết liên quan đề đề tài và giới thiệu về nền tảng ARM Cortex-M4F.

Chương 2: Trình bày chi tiết thiết kế phần cứng.

Chương 3: Thiết kế phần mềm

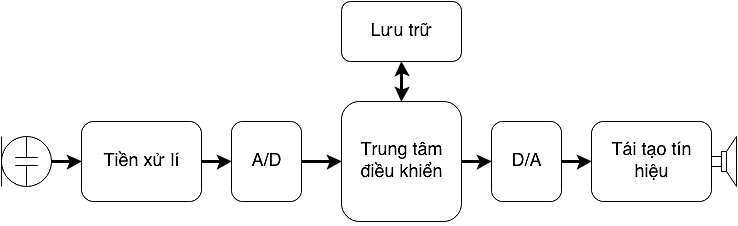
Chương 4: Kết quả thực nghiệm và thảo luận

# CÁC LÝ THUYẾT LIÊN QUAN

Chương này điểm lại một số lý thuyết liên quan đến thiết kế hệ thống thu phát tín hiệu băng tần tiếng nói. Ngoài ra, nền tảng phát triển ARM Cortex-M4F cùng với kit phát triển Tiva C Launchpad cũng được giới thiệu.

* 1. Cơ bản về thiết kế máy thu phát tín hiệu băng tần tiếng nói:

Các thiết kế máy thu phát tín hiệu băng tần tiếng nói về cơ bản bao gồm thành phần xử lí trung tâm chịu trách nhiệm lấy mẫu tín hiệu đầu vào, xử lí và lưu trữ tín hiệu đó, và cuối cùng là tái tạo lại tín hiệu ở đầu ra. Hình 1.1 là sơ đồ khối cho một thiết kế điển hình:



Hình 1.1 – Sơ đồ khối một thiết kế điển hình máy thu phát tín hiệu băng tần tiếng nói

Tín hiệu tiếng nói được đưa qua bộ tiền xử lí, chịu trách nhiệm xử lí tín hiệu ban đầu như khuếch đại, lọc nhiễu, lọc khử hiện tượng chồng phổ tín hiệu (Anti-Aliasing filter - AAF). Tín hiệu tiếp tục được trung tâm điều khiển lấy mẫu bằng Bộ chuyển đổi tín hiệu tương tự - số (ADC), sau đó xử lí và đưa vào thiết bị lưu trữ dưới dạng số. Khi có yêu cầu phát lại tín hiệu, trung tâm điều khiển sẽ lấy tín hiệu số ở thiết bị lưu trữ và đưa đến bộ phận tái tạo tín hiệu thông qua Bộ chuyển đổi tín hiệu số - tương tự (DAC). Bộ tái tạo tín hiệu sẽ tái tạo lại tín hiệu ban đầu và đưa ra thiết bị nghe.

* 1. Số hóa tín hiệu băng tần tiếng nói
     1. Tín hiệu băng tần tiếng nói:

Tiếng nói của con người thông thường có tần số cơ bản (fundamental frequency) là 120 Hz đối với nam và 210 Hz đối với nữ [3]. Tuy nhiên, khi thực hiện truyền tín hiệu tiếng nói, nếu chỉ cung cấp băng tần vừa đủ, ví dụ 250 Hz, để truyền đi thì khả năng nhận biết được giọng nói sẽ rất kém [4]. Để nghe được, việc cần thiết là phải truyền thêm cả các hài của tần số cơ bản.

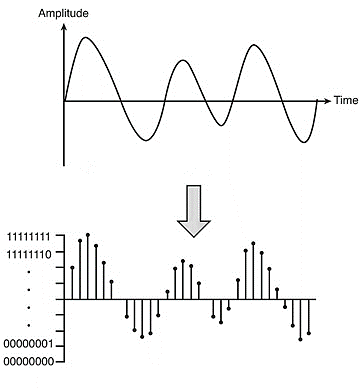
Trong công nghiệp truyền tín hiệu thoại, tín hiệu tiếng nói được truyền đi có băng thông 4000 Hz, trong đó đảm bảo các thành phần dưới 3400 Hz sẽ được truyền đi đầy đủ [5]. Dải 600 Hz còn lại được dành ra cho vùng chuyển tiếp của một bộ lọc tín hiệu lọc thông thấp với nhiệm vụ khử chồng phổ. Trong nhiều trường hợp, có thể lược bỏ cả các thành phần tần số từ DC đến 300 Hz của tín hiệu thoại. Lí do ở đây là một số thiết bị âm thanh công suất thấp có khả năng đáp ứng vùng tần số này rất kém. Việc lược bỏ đi các thành phần tần số từ DC đến 300 Hz không thay đổi khả năng nhận biết âm thanh thoại, hiệu ứng mất tần số cơ bản [6] cho thấy các tín hiệu bị loại bỏ thành phần tần số cơ bản vẫn cho cảm giác về cao độ của tần số đó.

* + 1. Số hóa tín hiệu

Để thực hiện xử lí và lưu trữ tín hiệu số, tín hiệu tiếng nói ban đầu phải được số hóa. Phương pháp điều chế tín hiệu xung – mã (Pulse-code modulation – PCM) là phương pháp phổ biến để chuyển đổi từ tín hiệu tương tự sang tín hiệu số. Một tín hiệu tương tự về bản chất có biên độ bất kì trong một vùng giá trị liên tục, ngược lại tín hiệu số chỉ có thể có được một số giá trị hữu hạn về biên độ. Việc chuyển đổi từ tín hiệu tương tự sang tín hiệu số thông qua hai quá trình gọi là lấy mẫu và lượng tử hóa.

Quá trình lấy mẫu tín hiệu phải tuân theo Định lí Nyquist – Shannon về lấy mẫu tín hiệu để tín hiệu ban đầu có thể được tái tạo lại hoàn chỉnh. Theo đó, nếu tần số cao nhất của tín hiệu là , tần số lấy mẫu là thì:

Nếu yêu cầu trên không thỏa mãn, hiện tượng chồng phổ (aliasing) sẽ xảy ra. Như vậy, nếu băng thông tín hiệu là 4000 Hz, tần số lấy mẫu tối thiểu đối với tín hiệu băng tần tiếng nói là 8000 Hz.



Hình 1.2 – Tín hiệu được mã hóa dạng PCM

Quá trình lượng tử hóa chính là thực hiện làm tròn giá trị biên độ của tín hiệu tương tự được lấy mẫu với các mức giá trị biên độ cho phép gọi là mức lượng tử [5]. Biên độ của tín hiệu tương tự x(t) nằm trong khoảng từ (-*m*, *m*) được phân thành L vùng với mỗi vùng có giá trị Δ*v* = 2*m*/L*.* Tiếp đó, giá trị biên độ các mẫu tín hiệu được xấp xỉ bởi điểm chính giữa vùng giá trị mà mẫu đó thuộc vào. Như vậy, giá trị mỗi mẫu tín hiệu sẽ tương ứng với một trong các giá trị nguyên từ 0 đến L-1. Nếu chuyển đổi các giá trị đó thành dạng nhị phân, ta sẽ có được chuỗi các mẫu tín hiệu đã được mã hóa dưới dạng nhị phân (Hình 1.2), thích hợp cho việc xử lí và lưu trữ trên các hệ thống số.

* + 1. Lấy mẫu đa tốc độ:

Một hệ thống lấy mẫu đa tốc độ (multirate system) cho phép tín hiệu lấy mẫu ở tốc độ ban đầu có khả năng tái lấy mẫu (resampling) với một tốc độ lấy mẫu khác.

Có hai loại lấy mẫu đa tốc độ:

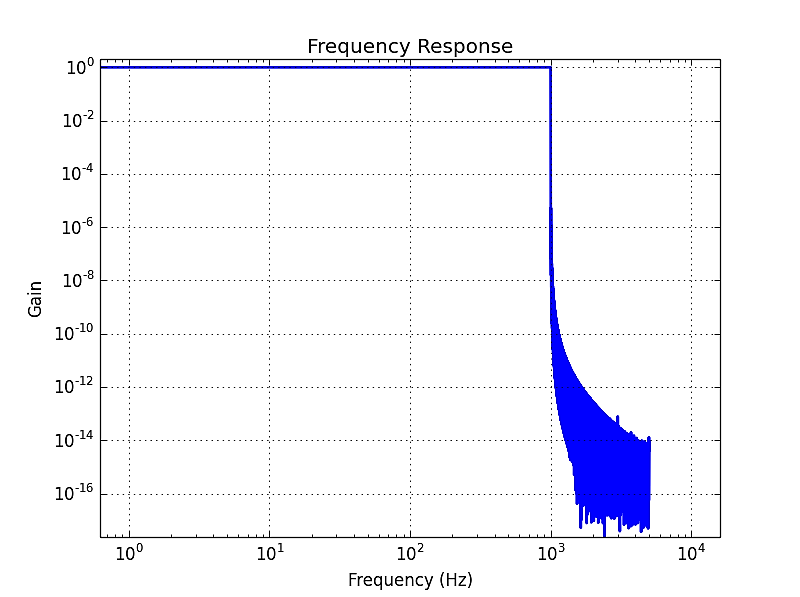
* Decimation: hạ tốc độ lấy mẫu. Tín hiệu ra có tốc độ lấy mẫu bằng số nguyên M lần so với tín hiệu vào.
* Interpolation: tăng tốc độ lấy mẫu. Tín hiệu ra có tốc độ lấy mẫu bằng nghịch đảo số nguyên L lần so với tín hiệu vào.

Lấy mẫu đa tốc độ là giải pháp linh hoạt khi các thành phần khác nhau trong hệ thống yêu cầu xử lí tín hiệu ở tốc độ lấy mẫu khác nhau. Lấy ví dụ, thao chuyển định dạng CD với tốc độ lấy mẫu 44.1 KHz sang định dạng DAT với tốc độ lấy mẫu 48 KHz có thể được thực hiện qua thao tác interpolation với L = 160 theo sau là decimation với M = 147 [7]. Trường hợp này, kết quả thu được là tốc độ lấy mẫu được chuyển đổi theo hướng tăng lên 1.09 lần, một kết quả thập phân bằng sự kết hợp của cả hai quá trình interpolation và decimation. Ngoài ra, sử dụng các kĩ thuật lấy mẫu đa tốc độ có thể giải quyết các vấn đề với các thành phần tương tự của hệ thống, như sẽ đề cập chi tiết ở mục 3.3.

* 1. Bộ lọc số lọc thông thấp
     1. Ưu điểm của các bộ lọc số:

Các bộ lọc số được sử dụng với hai mục đích chủ yếu [8]: Tách các thành phần tín hiệu của một tín hiệu tổng hợp; Khôi phục lại tín hiệu bị méo do các ảnh hưởng từ bên trong hay bên ngoài lên tín hiệu. Bộ lọc tương tự cũng có thể được dùng cho những tác vụ như trên. Tuy nhiên, sử dụng các bộ lọc số cho những kết quả tốt hơn rất nhiều so với các bộ lọc tương tự.

Ví dụ, Hình 1.3 là đáp ứng của bộ lọc thông thấp FIR với chiều dài đáp ứng xung là 40000, có độ lợi là 1 từ DC đến 1000 Hz và độ lợi bé hơn 10-5 từ 1001 Hz trở đi. Một đáp ứng như vậy gần như là điều không thể đối với các bộ lọc tương tự.



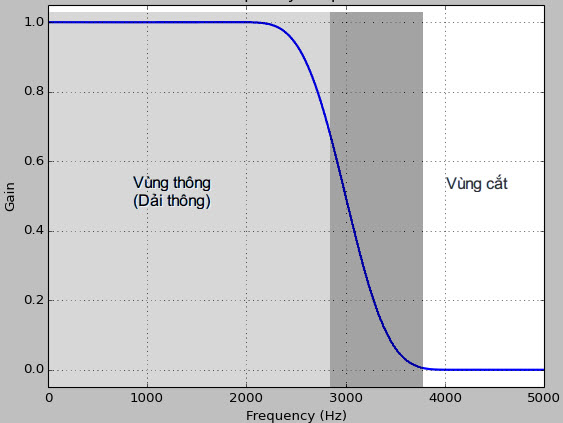
Hình 1.3 – Đáp ứng tần số của bộ lọc thông thấp FIR 40001 điểm

Ngoài ra, những lợi thế cơ bản mà một bộ lọc số mang lại so với bộ lọc tương tự là [7]:

* Độ chính xác cao, không bị ảnh hưởng bởi sai số của linh kiện.
* Có thể thiết kế bộ lọc có pha tuyến tính.
* Không có hiện tượng trượt giá trị do sự phụ thuộc của linh kiện vào các điều kiện môi trường.
* Tính linh hoạt cao và có thể xây dựng các bộ lọc có khả năng tự đáp ứng khi có sự thay đổi tính chất ở tín hiệu đầu vào.
* Dễ dàng trong việc mô phỏng và thiết kế với sự hỗ trợ của máy tính.
  + 1. Bộ lọc số lọc thông thấp:

Một bộ lọc số lọc thông thấp về cơ bản có ba vùng:

* Vùng thông (dải thông): vùng cho phép các thành phần tín hiệu có tần số bé hơn hoặc bằng tần số cắt đi qua. Tần số cắt là tần số mà tại đó độ lợi là về biên độ hay -3dB về công suất.
* Vùng chuyển tiếp: vùng chuyển giữa vùng thông sang vùng cắt. Độ lợi bé dần theo chiều tăng tần số cho đến khi đạt giá trị độ lợi vùng cắt.
* Vùng cắt: các thành phần tín hiệu có tần số trong vùng này đều bị loại bỏ.



Hình 1.4 – Đáp ứng tần số cơ bản của một bộ lọc số

Phân loại theo kiểu đáp ứng xung, có hai loại bộ lọc số lọc thông thấp là bộ lọc đáp ứng xung chiều dài hữu hạn (FIR) và bộ lọc đáp ứng xung chiều dài vô hạn (IIR). Về mặt tốc độ xử lí, các bộ lọc IIR cho tốc độ cao hơn so với FIR trong trường hợp các đặc tính của bộ lọc là giống nhau. Tuy nhiên bản chất của IIR yêu cầu tính hồi tiếp, tức là tín hiệu tiếp theo ở đầu ra không chỉ phụ thuộc vào tín hiệu đầu vào mà còn phụ thuộc vào tín hiệu hiện tại ở đầu ra, do đó bộ lọc IIR có khả năng không ổn định dẫn đến việc thiết kế phức tạp hơn so với bộ lọc FIR. Ngoài ra, bộ lọc FIR còn có một số ưu điểm nổi trội khác so với bộ lọc IIR nên vì thế bộ lọc FIR được sử dụng phổ biến trong các thiết kế. Có thể kể đến một số ưu điểm như sau [9]:

* Dễ dàng để thiết kế bộ lọc có pha tuyến tính, hay nói cách khác bộ lọc được thiết kế không làm méo pha tín hiệu đầu vào.
* Đơn giản hơn trong việc áp dụng các bộ lọc vào các vi xử lí, đối với hầu hết các vi xử lí hỗ trợ các thao tác DSP, các phép tính cho bộ lọc FIR có thể được triển khai chỉ bằng các vòng lặp nhất định một chỉ lệnh nào đó.
* Phù hợp với các hệ thống lấy mẫu đa tốc độ. Sử dụng bộ lọc FIR cho phép các tính toán không cần thiết có thể lược bỏ nhằm tối ưu hiệu năng của bộ lọc. Ngược lại, nếu sử dụng IIR, tất cả các phép tính đều phải thực hiện cho dù một số mẫu tín hiệu ở đầu ra không có ích.
  + 1. Phương pháp thiết kế bộ lọc số lọc thông thấp có đáp ứng xung chiều dài hữu hạn (FIR)

Một bộ lọc số kiểu đáp ứng xung chiều dài hữu hạn luôn luôn ổn định. Tín hiệu đầu vào đi qua qua bộ lọc số FIR, là hệ thống tuyến tính bất biến theo thời gian (Linear Time-Invariant - LTI), sẽ được tích chập với đáp ứng xung của bộ lọc (nhân của bộ lọc) để cho tín hiệu đầu ra như trên Hình 1.5.

Bộ lọc FIR

Hình 1.5 – Bộ lọc đáp ứng xung chiều dài hữu hạn FIR

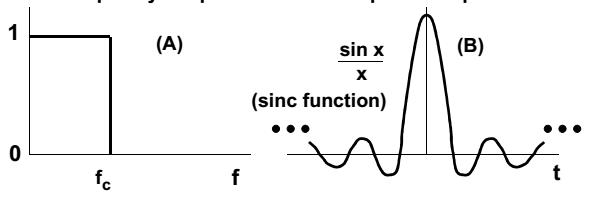
Thiết kế một bộ lọc FIR về bản chất chính là tìm đáp ứng xung h[i] cho bộ lọc đó. Theo [10], chiều dài đáp ứng xung của bộ lọc phụ thuộc vào độ rộng vùng chuyển tiếp. Độ rộng vùng chuyển tiếp càng lớn thì chiều dài tối thiểu cần thiết càng nhỏ và ngược lại. Mối quan hệ trên có thể được diễn tả như sau [10]:

Với số nguyên N là chiều dài đáp ứng xung của bộ lọc và B là độ rộng vùng chuyển tiếp được tính theo tỉ lệ so với tốc độ lấy mẫu.

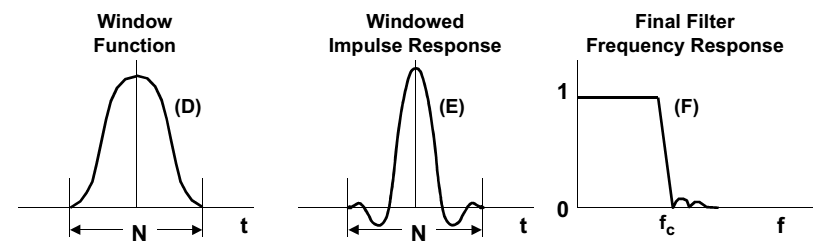
Ví dụ: Tốc độ lấy mẫu tín hiệu là 10 KHz, độ rộng vùng chuyển tiếp là 600 Hz thì B là:

Khi đó:

Một phương pháp thông dụng trong thiết kế bộ lọc FIR đó là áp dụng phương pháp cửa sổ Sinc (Windowed-Sinc). Đáp ứng lý tưởng như trên Hình 1.6, với đáp ứng xung là một hàm Sinc, là điều không thể trong thực tế. Để có một đáp ứng như vậy, chiều dài nhân bộ lọc FIR phải là vô hạn, điều mà các máy tính hiện tại không cho phép. Phương pháp cửa sổ Sinc về cơ bản là chỉ lấy một phần hữu hạn các điểm cần thiết trên hàm Sinc rồi áp dụng một hàm gọi là cửa sổ lên phần đó để tạo thành nhân của bộ lọc với các yêu cầu của thiết kế (Hình 1.7).



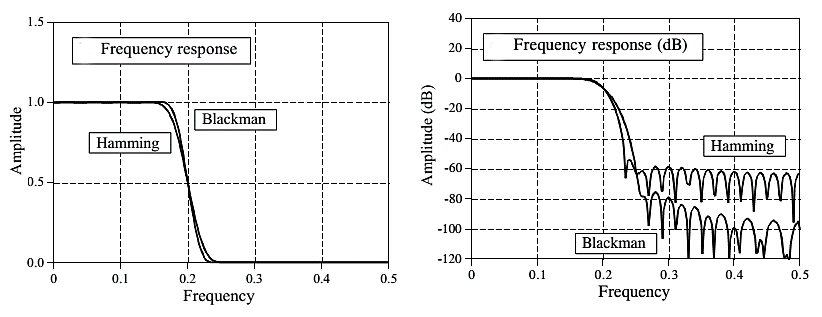
Hình 1.6 – Đáp ứng lý tưởng của bộ lọc thông thấp



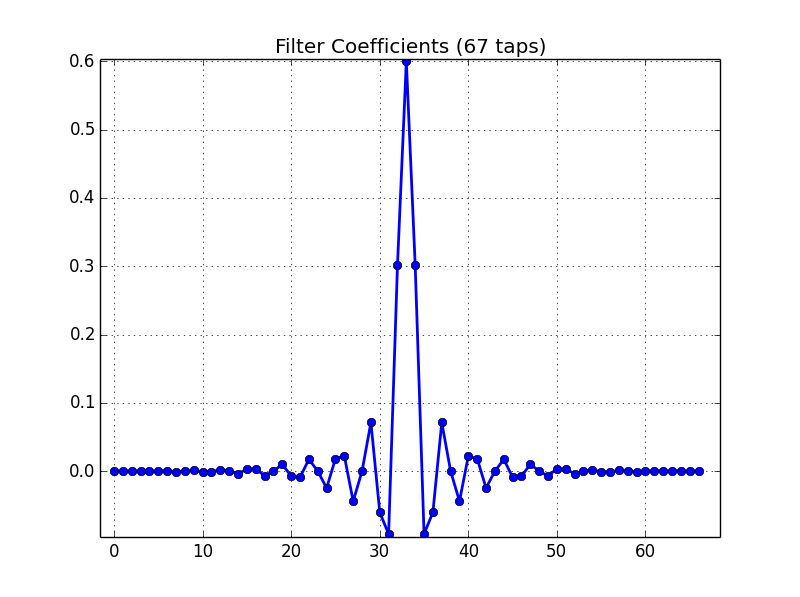
Hình 1.7 – Phương pháp cửa sổ Sinc

Các cửa sổ sẽ quy định một phần tính chất của bộ lọc mà nó tạo ra. Có nhiều loại cửa sổ, tuy nhiên hai loại thông dụng nhất là cửa sổ Hamming và Blackman [10]. Công thức của cửa sổ Hamming và Blackman lần lượt là:

Cửa sổ Hamming cho dải chuyển tiếp có độ dốc lớn hơn Blackman khoảng 20%, tuy nhiên Blackman lại có độ suy hao ở vùng cắt tốt hơn nhiều (Hình 1.8). Tùy theo yêu cầu của thiết kế để chọn loại cửa sổ phù hợp.



Hình 1.8 – Đáp ứng khi dùng cửa sổ Hamming và Blackman



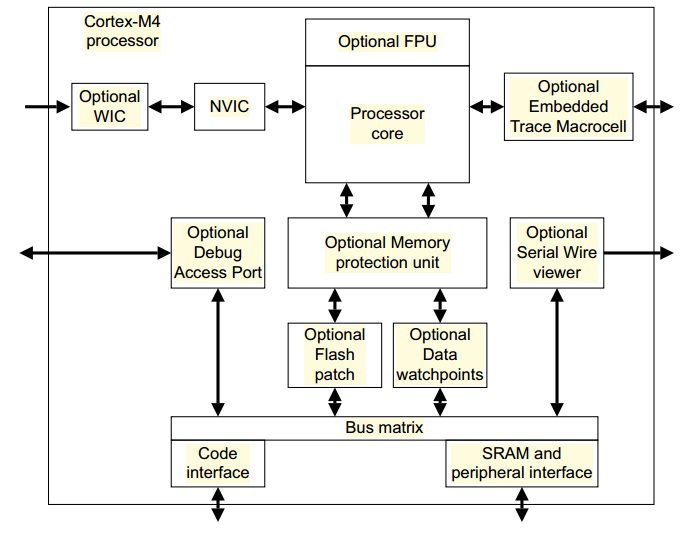
Hình 1.9 – Đáp ứng xung bộ lọc FIR cửa số Blackman với N = 66

Áp dụng cửa sổ Blackman cho thiết kế bộ lọc thông thấp với N = 66 và là 1 KHz, các giá trị của h[i] được tính theo công thức sau [10]:

Tính toán bằng máy tính cho kết quả được thể hiện trên Hình 1.9.

* 1. Nền tảng phát triển ARM Cortex-M4F
     1. ARM Cortex-M4F [11]

ARM Cortex-M4F là nhân xử lí ARM Cortex-M4 với sự hỗ trợ về năng lực xử lý dấu phẩy động của Đơn vị dấu phẩy động (FPU). Nhân xử lí ARM Cortex-M4 là nhân 32 bit được thiết kế để tích hợp vào các vi điều khiển yêu cầu khả năng xử lí cao, đáp ứng các tác vụ điều khiển và xử lí tín hiệu phức tạp. Hình 1.10 trình bày cấu trúc cơ bản của một nhân ARM Cortex-M4F.



Hình 1.10 – Cấu trúc cơ bản của kiến trúc ARM Cortex-M4F

Các đặc trưng nổi bật của nhân ARM Cortex-M4F bao gồm:

* Xây dựng trên kiến trúc ARMv7E-M.
* Tập lệnh đa dạng:
  + Thumb (đầy đủ chỉ lệnh)
  + Thumb-2 (đầy đủ chỉ lệnh)
  + Phép nhân 32 bit trong vòng một chu kỳ lệnh, phép chia 32 bit trong vòng 2-12 chu kỳ lệnh.
  + Phần mở rộng cho DSP: thao tác nhân-tích lũy MAC 16/32 bit trong vòng một chu kỳ lệnh, các thao tác Một lệnh – nhiều dữ liệu (SIMD) 8/16 bit.
* Tích hợp FPv4-SP là nhân FPU hỗ trợ các tính toán dấu phẩy động.
* Ba tầng pipeline với khả năng dự đoán tình huống rẽ nhánh.
* 1 đến 240 nguồn ngắt vật lý, độ trễ ngắt là 12 chu kỳ lệnh.
* Tốc độ xử lý 1.25 DMIPS/MHz
  + 1. Đơn vị dấu phẩy động – Floating Point Unit (FPU) [12] [13]

FPU của ARM Cortex-M4F được dựa trên kiến trúc mở rộng FPv4-SP. FPU hỗ trợ đầy đủ các thao tác cộng, trừ, nhân, chia, phép nhân và tích lũy, và phép căn bậc hai cho số thập phân có độ chính xác đơn (single-precision). Ngoài ra, FPU còn hỗ trợ các thao tác chuyển đổi giữa số nguyên và số thập phân, các chỉ lệnh trên các hằng số với số thập phân.

FPU cung cấp khả năng tính toán hợp quy theo tiêu chuẩn *ANSI/IEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic*.

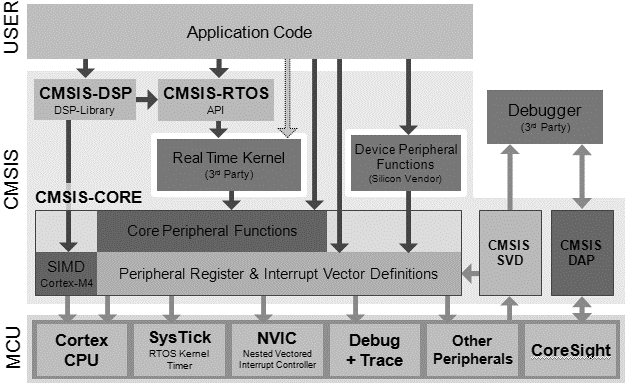
Đặc điểm của FPU:

* Các chỉ lệnh độ rộng 32 bit cho số thập phân độ chính xác đơn (kiểu C float) các thao tác xử lí dữ liệu.
* Kết hợp các chỉ lệnh nhân-tích lũy để tăng độ chính xác.
* Tăng tốc phần cứng cho các thao tác chuyển đổi, cộng, trừ, nhân với tùy chọn tích lũy, chia và căn bậc hai.
* 32 thanh ghi 32 bit độ chính xác đơn, có thể sử dụng như hai thanh ghi 16 bit.
* Ba tầng pipeline riêng biệt giúp tăng tốc độ xử lí đồng thời.
  + 1. Chuẩn giao tiếp phần mềm vi điều khiển Cortex (Cortex Microntroller Software Interface Standard – CMSIS) [14]

CMSIS là Lớp trừu tượng hóa phần cứng (Hardware Abstraction Layer – HAL) cho các vi điều khiển bất kì thuộc dòng Cortex-M, không phụ thuộc vào nhà sản xuất phần cứng. CMSIS được phát triển bởi ARM nhằm cung cấp cho người phát triển phần mềm vi điều khiển các giao tiếp đến vi xử lí và các ngoại vi một cách đơn giản thông qua các Giao diện lập trình ứng dụng (Application Programming Interface - API). Qua đó, đơn giản hóa việc tái sử dụng phần mềm, giảm thời gian tìm hiểu và giảm thời gian phát triển phần mềm cho thiết bị.

CMSIS bao gồm các thành phần sau:

* CMSIS-CORE: API cho nhân xử lí Cortex-M và các ngoại vi. Cung cấp giao tiếp được chuẩn hóa cho các nhân xử lí Cortex-M.
* CMSIS-DSP: Thư viện xử lí số tín hiệu (Digital Signal Processing – DSP). Cung cấp các thao tác phục vụ cho xử lí số tín hiệu với tối ưu hóa cho các chỉ lệnh Đơn lệnh – Đa dữ liệu (Single Instruction Multiple Data – SIMD).
* CMSIS-RTOS API: Các API cho hệ điều hành thời gian thực (RTOS).
* CMSIS-SVD: Các đặc tả hệ thống cho ngoại vi. Đặc tả các ngoại vi của thiết bị trong một file XML để có thể sử dụng trong việc hỗ trợ cho bộ gỡ lỗi và các chỉ định ngắt.
* CMSIS-DAP: Debug Access Port. Cung cấp chuẩn hóa cho thiết bị gỡ lỗi.

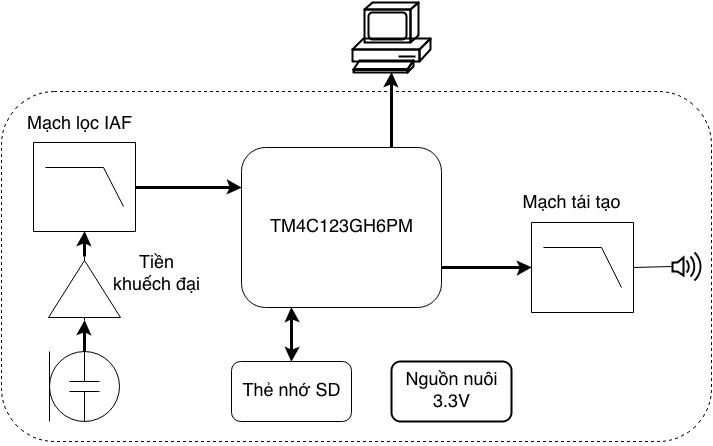


Hình 1.11 – Cấu trúc của CMSIS [14]

# THIẾT KẾ PHẦN CỨNG

Chương này đưa ra sơ đồ kết nối phần cứng cũng như chi tiết các mạch cần thiết trong hệ thống.

* 1. Sơ đồ kết nối phần cứng



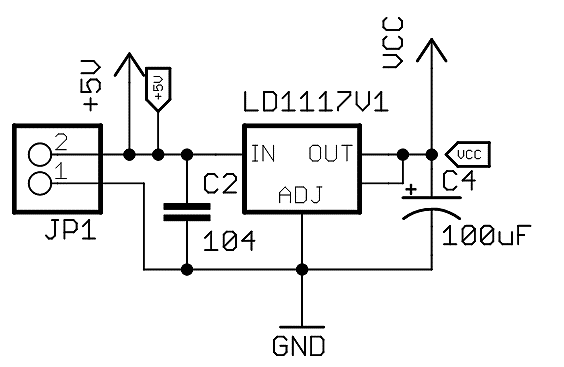
Hình 2.1 – Sơ đồ kết nối phần cứng

Phần trong đường đứt nét trên hình Hình 2.1 là sơ đồ kết nối phần cứng. Tín hiệu từ Micro được qua bộ Tiền khuếch đại rồi sau đó đưa đến Mạch lọc khử chồng phổ AAF. Tín hiệu từ AAF sẽ được xử lí bởi vi điều khiển TM4C123GH6PM trên kit Tiva C Launchpad. Thẻ nhớ SD được kết nối tới vi điều khiển để lưu trữ dữ liệu. Tín hiệu xuất ra từ vi điều khiển đi qua mạch tái tạo để tái tạo tín hiệu thu được rồi được đưa đến thiết bị âm thanh.

* 1. Nguồn nuôi

Mạch nguồn được thiết kế để cung cấp điện áp đơn +3.3V cho hoạt động của hệ thống. Việc sử dụng điện áp nguồn đơn cho phép hệ thống linh động hơn trong việc tận dùng các nguồn điện di động như Pin AA hay Pin Li-Ion, Li-Po.

Một điểm cần lưu ý là thiết kế có sử dụng opamp. Nguồn cấp thông thường cho opamp là nguồn đôi với mức điện áp tham chiếu là GND. Với nguồn đơn, điện áp tham chiếu cho opamp không thể là GND mà phải thay đổi cho phù hợp [15], thông thường chọn là ½ VCC để có độ rộng biên độ dao động tín hiệu ở mức cao nhất.

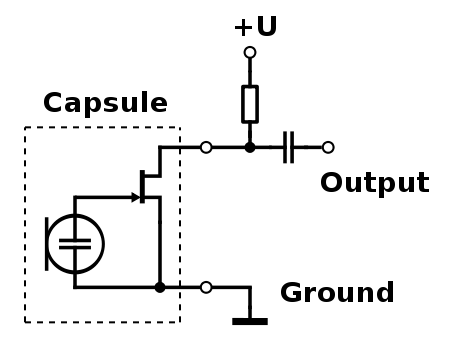


Hình 2.2 – Sơ đồ nguyên lí mạch nguồn

Sơ đồ mạch nguồn của thiết kể được thể hiện trên hình Hình 2.2. IC ổn áp được sử dụng là LD1117V33 của STMicroelectronics với các đặc điểm sau [16]:

* Điện áp dropout thấp, ~1V
* Dòng ra lớn nhất 800mA
* Tự động giới hạn dòng và giới hạn nhiệt
* Độ chính xác ±1% ở 25oC và ±2% trên toàn dải nhiệt độ hoạt động
* Khử nhiễu nguồn: 75dB (giá trị thông thường)
  1. Mạch tiền khuếch đại và mạch lọc khử chồng phổ AAF
     1. Electret Condenser Microphone (ECM)

ECM được sử dụng trong hầu hết các thiết bị âm thanh thông thường và trong các giải pháp truyền tín hiệu âm thanh [17].Dưới đây là một cấu hình cơ bản để ECM hoạt động.



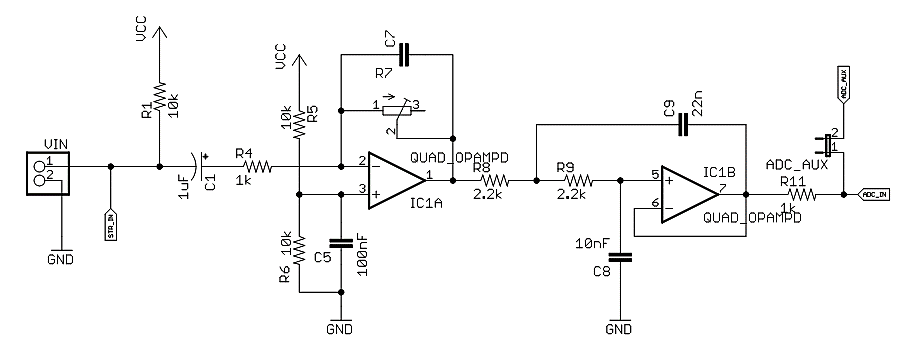
Hình 2.3 – Cấu hình cơ bản của ECM

Nguyên lí hoạt động của ECM dựa vào sự thay đổi điện dung giữa hai bản cực [18]. Bản cực thứ nhất được cố định, bản cực thứ hai có thể rung động được gọi là lớp màng, hai bản cực tạo thành một tụ điện. Tụ điện này đã được nạp sẵn điện [19], bản chất vật liệu chế tạo cho phép tụ có khả năng trữ điện trong một thời gian rất dài. Khi có âm thanh làm rung động lớp màng, điện dung của tụ điện thay đổi dẫn đến sự thay đổi điện áp giữa hai bản cực theo công thức:

Sự thay đổi này dẫn đến sự thay đổi dòng điện cực máng của JFET làm biến đổi tín hiệu ở đầu ra.

* + 1. Mạch tiền khuếch đại và mạch lọc khử chồng phổ AAF

Hình 2.4 là sơ đồ nguyên lý mạch tiền khuếch đại và mạch lọc khử chồng phổ AAF:



Hình 2.4 – Mạch tiền khuếch đại và lọc khử chồng phổ

Mạch tiền khuếch đại có hai nhiệm vụ cơ bản sau:

* Phân cực tín hiệu từ ECM dao động quanh mức tham chiếu ½ VCC.
* Khuếch đại tín hiệu.

R5, R6 là cầu phân áp làm nhiệm vụ phân cực tín hiệu ở mức tham chiếu ½ VCC. Với một opamp có khả năng cho điện áp đầu ra tiệm cận nguồn, biên độ đỉnh-đỉnh của tín hiệu từ ECM đã khuếch đại sẽ gần bằng giá trị VCC. Tín hiệu đầu ra của ECM có biên độ rất nhỏ, thông thường không quá 200mV, nên cần phải khuếch đại lên với độ lợi thích hợp để đạt được biên độ mong muốn trước khi đưa vào ADC.

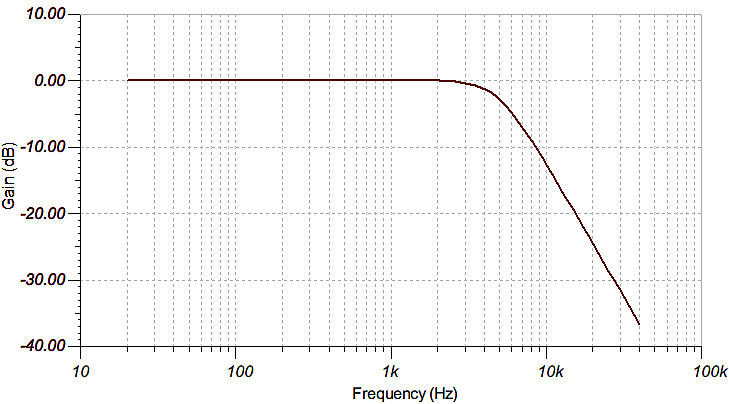
Độ lợi của bộ khuếch đại được tính theo công thức sau:

Độ lợi có thể được thay đổi theo biến trở R5.

Opamp được sử dụng ở đây là OPA4340 của Texas Instruments. OPA4340 có các đặc điểm nổi bật sau [20]:

* Phù hợp với nguồn đơn.
* Điện áp vào/ra tiệm cận điện áp nguồn.
* Nhiều thấp, phù hợp với các ứng dụng âm thanh và chuyển đổi A/D.

Mạch lọc khử chồng phổ là mạch lọc thông thấp theo cấu hình Sallen-Key với đáp ứng tần số Butterworth [21].



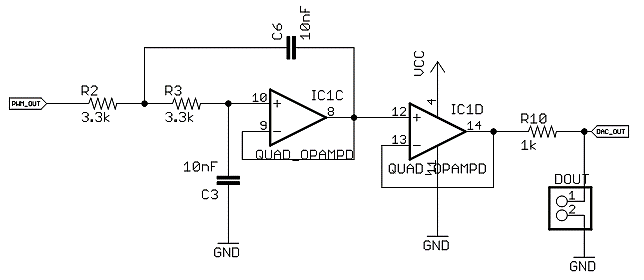
Hình 2.5 – Đáp ứng tần số mạch lọc khử chồng phổ AAF

Hình 2.5 là đáp ứng tần số của mạch được mô phỏng trên máy tính sử dụng phần mềm Tina-TI [22]. Theo kết quả mô phỏng, mạch có tần số cắt là 5.09 KHz và dải chuyển tiếp từ 5.09 KHz đến 16 KHz với suy hao tại 16 KHz là -15.43dB.

* 1. Mạch tái tạo tín hiệu đầu ra:

Mạch tái tạo tín hiệu đầu ra bao gồm một mạch lọc thông thấp và một bộ đệm tín hiệu với trở kháng ra đủ để cấp công suất cho một tai nghe.

Hình 2.6 là sơ đồ nguyên lí mạch tái tạo tín hiệu đầu ra:

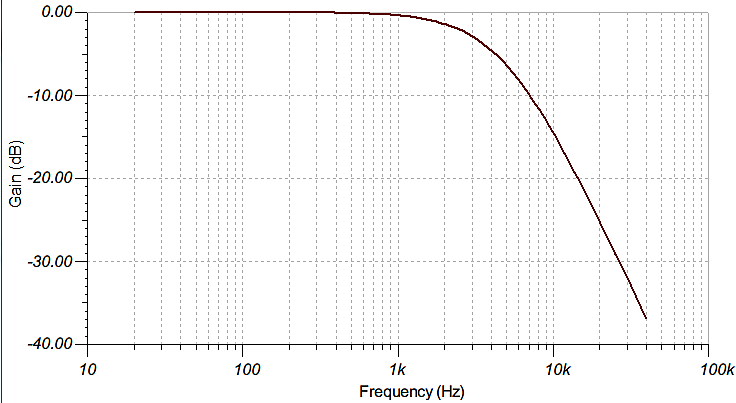


Hình 2.6 – Mạch tái tạo tín hiệu đầu ra

Tín hiệu PWM từ mạch điều khiển được đưa qua bộ lọc thông thấp với đáp ứng được thể hiện trên Hình 2.7.

Tần số cắt của mạch là 4.67 KHz.

Bộ đệm tín hiệu có chức năng cách li phần tín hiệu đầu ra từ mạch lọc với tải bên ngoài, trở kháng đầu ra của bộ đệm tín hiệu được xác định bởi R1 là 470 Ω. Với tai nghe thông thường có trở kháng đặc trưng là 32 Ω. Công suất đầu ra đến tai nghe là:



Hình 2.7 – Đáp ứng bộ lọc thông thấp mạch tái tạo tín hiệu đầu ra

* 1. Mạch điều khiển chính - Kit phát triển Tiva C Launchpad nền tảng ARM Cortex-M4F

Mạch điều khiển chính sử dụng kit phát triển Tiva C Launchpad của Texas Instrument [23]. Hình 2.8 là sơ đồ khối của kit Tiva C Launchpad.

Tiva C Launchpad sử dụng vi điều khiển TM4C123GH6PM với nhân ARM Cortex-M4F [24]. Kit cũng tích hợp phần debugger để hỗ trợ gỡ lỗi trong quá trình phát triển phần mềm.

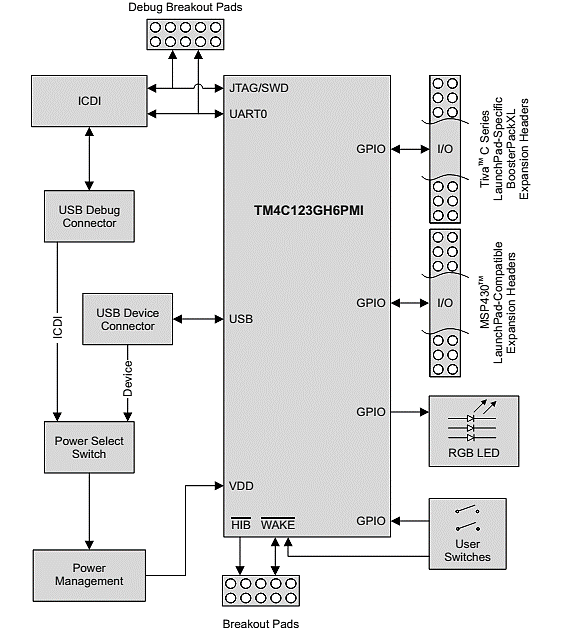
Đặc điểm của TM4C123GH6PM [13] trên kit Tiva C Launchpad:

* ARM Cortex-M4 CPU và FPU, tốc độ tối đa 80 MHz
* 256 KB Flash, 32 KB RAM và 2 KB EEPROM
* 2x 12 kênh ADC, tốc độ lấy mẫu tối đa 1 MSPS (triệu mẫu/giây)

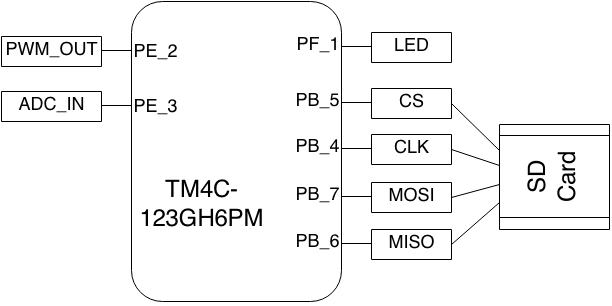
16 kênh PWM

* 24 bộ hỗn hợp Timer/Capture/PWM
* 4 bộ truyền thông đồng bộ SSI, 8 bộ truyền thông không đồng bộ UART
* 43 I/O đa mục đích
* ROM trên chip được cài sẵn driver cho các ngoại vi

Sơ đồ kết nối các chân tín hiệu của Tiva C Launchpad trên mạch được thể hiện trên Hình 2.7.



Hình 2.8 – Sơ đồ khối Tiva C Launchpad



Hình 2.9 – Sơ đồ kết nối các chân tín hiệu của Tiva C Launchpad trên mạch

# THIẾT KẾ PHẦN MỀM

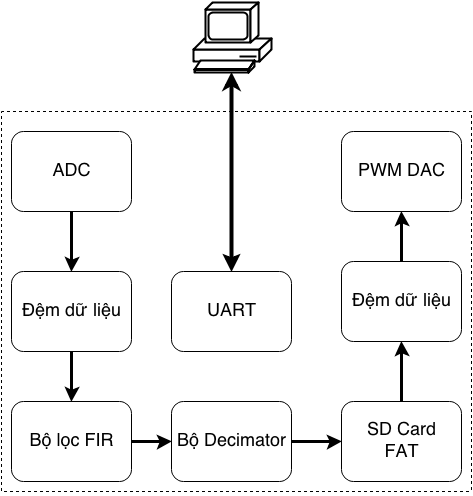
Chương này trình bày cấu trúc phần mềm hệ thống và các thiết kế mô-đun phần mềm phục vụ cho các thao tác lấy mẫu tín hiệu, xử lí tín hiệu và quản lí dữ liệu.

* 1. Tổng quan

Phần mềm điều khiển cần thực hiện các nhiệm vụ sau:

* Lấy mẫu tín hiệu và xử lí tín hiệu trong thời gian thực.
* Phát tín hiệu từ dữ liệu lưu trữ trên thẻ nhớ sử dụng PWM.
* Cung cấp giao diện quản lí dữ liệu trên thẻ nhớ SD thông qua chuẩn FAT.
* Giao tiếp với người sử dụng thông qua phương thức truyền thông nối tiếp không đồng bộ UART.

Hình 3.1 mô tả cấu trúc thiết kế phần mềm của hệ thống:



Hình 3.1 – Cấu trúc thiết kế phần mềm hệ thống

Một cách tổng quát, đối với quá trình lấy mẫu tín hiệu và ghi dữ liệu vào thẻ nhớ SD, trình tự thực hiện các bước như sau:

* Chuyển đổi tương tự - số A/D với tốc độ 32 KHz sử dụng bộ đệm dữ liệu.
* Lọc thông thấp bằng bộ lọc số FIR có tần số cắt là 3.4 KHz.
* Hạ tốc độ lấy mẫu từ 32 KHz xuống 8 KHz bằng bộ Decimator.
* Ghi dữ liệu thành file dạng .WAV lên thẻ nhớ SD chuẩn FAT.

Đối với quá trình phát tín hiệu, trình tự thực hiện các bước là:

* Đọc dữ liệu file .WAV từ thẻ nhớ SD vào bộ đệm.
* Thực hiện phát tín hiệu bằng điều chế PWM vào bộ lọc để tái tạo lại tín hiệu ban đầu.
  1. Lấy mẫu tín hiệu với bộ chuyển đổi A/D
     1. Bộ chuyển đổi A/D trên TM4C123GH6PM [13]

ADC của TM4C123GH6PM có độ phân giải 12 bit và hỗ trợ 12 kênh đầu vào cùng với một kênh cảm biến nhiệt độ bên trong. ADC chứa bốn bộ lấy mẫu liên tiếp cho phép thực hiện lấy mẫu nhiều kênh tín hiệu tương tự đồng thời mà không cần sự can thiệp của phần mềm. Mỗi bộ lấy mẫu đều cung cấp khả năng lập trình kênh đầu vào, sự kiện kích, khả năng tạo ngắt và chế độ ưu tiên. Chi tiết các đặc điểm của ADC được thể hiện dưới đây:

* 12 kênh đầu vào được chia sẻ chung
* Độ phân giải 12 bit.
* Cấu hình đầu vào đơn hay đầu vào vi sai.
* Cảm biến nhiệt độ bên trong.
* Tốc độ lấy mẫu tối đa 1 MSPS
* Có khả năng lựa chọn chế độ sai pha từ 22.5o đến 337.5o
* Bốn bộ lấy mẫu
* Hỗ trợ chế độ lấy giá trị trung bình, tối đa 64 mẫu.
* Hỗ trợ truy xuất bộ nhớ trực tiếp μDMA
  + 1. Kĩ thuật lấy mẫu tín hiệu theo chu kì

Một bộ định thời được sử dụng để cung cấp sự kiện kích lấy mẫu theo chu kì nhằm đảm bảo chu kì lấy mẫu của ADC là chính xác.

Với tốc độ lấy mẫu là 32 KHz, chu kì lấy mẫu là:

Ở tốc độ xử lí 80 MHz, bộ định thời Timer0 của TM4C123GH6PM được thiết lập ở chế độ định thời 32 bit. Số xung cần thiết cho Timer0 để kích ADC lấy mẫu là:

Như vậy, cứ mỗi 2500 xung đồng hồ, Timer0 sẽ thông qua mạch logic phần cứng để kích bộ ADC hoạt động, đồng thời sẽ nạp lại giá trị xung đồng hồ cho lần kích tiếp theo.

* 1. Kỹ thuật xử lí tín hiệu đa tốc độ lấy mẫu (Multirate Signal Processing)
     1. Hạn chế trong việc thiết kế bộ lọc tích cực cho AAF

Với tốc độ lấy mẫu tín hiệu 8 KHz, mạch AAF phải đảm bảo loại bỏ tất cả các thành phần tín hiệu có tần số trên 4 KHz. Như đã trình bày ở mục ***1.2.1***, dải chuyển tiếp của mạch AAF là 600 Hz. Thực hiện thiết kế mạch lọc tích cực cực sử dụng phần mềm FilterPro Desktop [25] với các thông số:

* Kiểu mạch lọc: Mạch lọc thông thấp
* Tần số cắt: 3.4 KHz
* Độ rộng vùng chuyển tiếp: 600 Hz
* Độ suy hao vùng chuyển tiếp: - 40 dB

Các tính toán trên phần mềm cho thấy bậc của mạch lọc tối thiểu là 10 với kiểu đáp ứng xung Chebyshev và cấu hình Sallen-Key. Với các kiểu đáp ứng khác, số bậc phải lớn hơn 10. Cần phải ghép tối thiểu 5 tầng opamp với các linh kiện thụ động khác để tạo nên mạch trên. Điều này gây ra những khó khăn như sau:

* Phức tạp khi chọn lựa các linh kiện phù hợp với mạch lọc.
* Phức tạp khi thiết kế mạch in.
* Sai số linh kiện, sai số do nhiệt cùng với nhiễu tích lũy qua nhiều tầng gây ra ảnh hưởng đến đáp ứng và gây méo tín hiệu.
  + 1. Cơ sở kĩ thuật xử lí đa tốc độ lấy mẫu:

Để giải quyết vấn đề trên, kĩ thuật xử lí đa tốc độ lấy mẫu có thể được áp dụng. Mô hình cho hệ thống xử lí đa tốc độ lấy mẫu như sau:

Tín hiệu

AAF

ADC

Bộ lọc số FIR

Decimator

Tín hiệu

Hình 3.2 – Kĩ thuật xử lí đa tốc độ lấy mẫu

Tín hiệu đầu vào được lấy mẫu với tốc độ lấy mẫu là 32 KHz. Như vậy, bộ lọc khử chồng phổ có thể được thiết kế với một dải chuyển tiếp rộng hơn nhưng vẫn đảm bảo thành phần tần số bé hơn 3.4 KHz được giữ lại hoàn toàn. Điều này làm giảm độ phức tạp cho công việc thiết kế AAF. Tiếp theo, một bộ lọc số FIR có tần số cắt 3.4 KHz với dải chuyển tiếp 600 Hz sẽ thực hiện loại bỏ các thành phần tần số từ 4 KHz đến 16 KHz. Bộ Decimator về bản chất là bộ hạ tốc độ lấy mẫu. Decimator thực hiện hạ tốc độ lấy mẫu bằng cách trích 1 mẫu trong một nhóm 4 mẫu liên tục để tạo thành tín hiệu cuối cùng với tần số lấy mẫu 8 KHz và tần số cắt là 3.4 KHz.

* + 1. Thuật toán cho bộ Decimator:

Bộ Decimator tỉ lệ 4:1 có thể được diễn giải bằng thuật toán dưới:

*Input:* Mảng in[2048] chứa các mẫu liên tiếp của tín hiệu đầu với với tần số lấy mẫu 32 KHz.

*Output:* Mảng out[512], chứa các mẫu đầu ra với tần số lấy mẫu 8 KHz.

*Bước 1:* i ← 0; t ← 0.

*Bước 2:* out[t] = in[i].

*Bước 3:* t ← t + 1; i ← t \* 4

*Bước 4:* Nếu t = 512. Kết thúc.

*Bước 5:* Quay về *Bước 2*.

* 1. Hệ thống quản lí file theo chuẩn FAT
     1. Đặc điểm của FAT

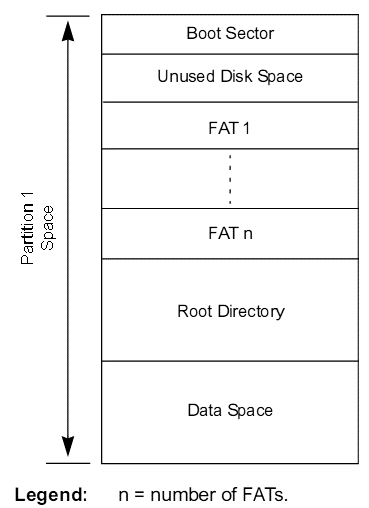
Dữ liệu ghi vào thiết bị lưu trữ cần được quản lí tốt để dễ dàng truy cập và sử dụng trong tương lai. FAT là một giải pháp tốt cho việc quản lí dữ liệu trên thiết bị lưu trữ như thẻ nhớ SD.

Bảng cấp phát tập tin (FAT) được Microsoft phát triển từ những năm 1970 [26], với phiên bản mới nhất là FAT32 được ra mắt năm 1996.

FAT cung cấp giải pháp quản lí tiện lợi, đơn giản và có thể tích hợp dễ dàng trong các phần mềm hệ thống nhúng.

Thẻ nhớ SD thông thường chỉ có một phân vùng, bao gồm các thành phần sau [27]:

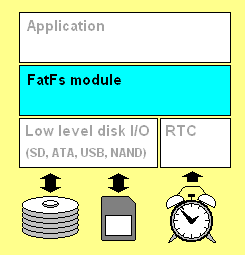
* Boot Sector: Sector đầu tiên của phân vùng chứa các thông tin cơ bản.
* Vùng FAT: Chứa bản ghi thể hiện các sắp xếp dữ liệu trên vùng dữ liệu. Thông thường có 2 bảng FAT giống nhau trong vùng này để đề phòng trường hợp xảy ra mất mát dữ liệu.
* Root Directory: Chứa bảng tập hợp các thư mục con và các file ở thư mục cấp cao nhất (root). Thông tin về các thư mục con và file cấp thấp hơn được chứa tại thư mục cấp trên nó.
* Vùng dữ liệu (Data Region): Chứa dữ liệu tệp hoặc thông tin các thư mục con. Dữ liệu của vùng này vẫn được giữ nguyên trong mọi trường hợp ngoại trừ trường hợp bị ghi đè lên.



Hình 3.3 – Cấu trúc của một phân vùng sử dụng FAT

* + 1. Thư viện FatFs [28]

FatFs là một module cung cấp giải pháp quản lí theo chuẩn FAT cho các ứng dụng nhúng. FatFs được viết theo quy chuẩn ANSI C và hoàn toàn tách biệt với lớp giao diện vào ra với thiết bị lưu trữ dữ liệu. Do đó FatFs không phụ thuộc vào kiến trúc phần cứng, nó có thể được tích hợp vào hầu hết các kiến trúc vi điều khiển mà không cần phải thực hiện bất kì thay đổi gì ở mã nguồn.

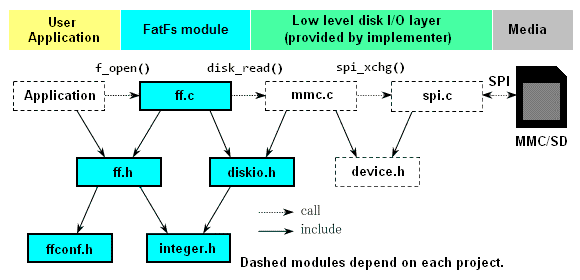


Hình 3.4 – Ví trí lớp thư viện FatFs trong các lớp của phần mềm

Các đặc trưng của FatFs:

* Tương thích với chuẩn FAT của Windows
* Không phụ thuộc vào nền tảng phần cứng, dễ dàng khi chuyển đổi nền tảng.
* Hỗ trợ kiểu tên nhiều kí tự theo chuẩn ANSI/OEM hay Unicode.
* Hỗ trợ hệ điều hành thời gian thực RTOS
* Hỗ trợ nhiều loại kích thước sector
* API được tối thiểu hóa, chế độ chỉ đọc, đệm vào/ra…
  + 1. Ứng dụng thư viện FatFs để triển khai FAT32 trên thẻ nhớ SD

Hình 3.5 đây mô tả cấu hình cơ bản của một hệ thống nhúng với FatFs [29]:



Hình 3.5 – Cấu hình cơ bản của hệ thống nhúng với FatFs

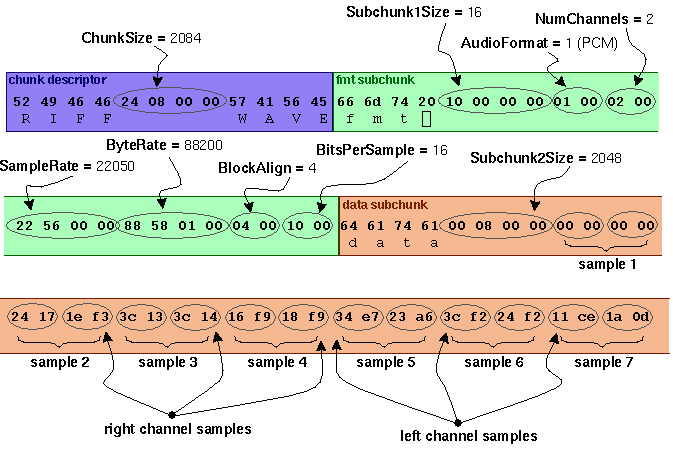
Phần mềm hệ thống phải cung cấp trình điều khiển thiết bị lưu trữ dữ liệu ở lớp I/O cho module FatFs. Cụ thể là các hàm dưới đây:

* static void **power\_on**(): Khởi tạo các I/O và module SSI cần thiết cho việc thực hiện truyền dữ liệu theo phương thức SPI.
* static BYTE **send\_cmd**(): Gởi các lệnh điều khiển đến thẻ nhớ SD.
* static BOOL **rcvr\_block**(): Nhận một gói dữ liệu từ thẻ nhớ SD.
* static BOOL **xmit\_datablock**(): Gởi một gói dữ liệu đến thẻ nhớ SD.
* DSTATUS **disk\_initialize**(): Khởi tạo phân vùng sử dụng FatFs.
* DSTATUS **disk\_status**(): Kiểm tra trạng thái phân vùng.
* DRESULT **disk\_read**(): Đọc một hoặc nhiều sector.
* DRESULT **disk\_write**(): Ghi một hoặc nhiều sector.
* DRESULT **disk\_ioctl**(): Cung cấp các chức năng phụ

Ở lớp ứng dụng, FatFs cung cấp các thao tác trên file và thư mục thông qua các API dưới đây [28]:

* **f\_mount** – Khởi tạo hay hủy bỏ phân vùng làm việc
* **f\_open** – Mở hay tạo mới một file
* **f\_close** – Đóng một file
* **f\_read** – Đọc một file
* **f\_write** – Ghi dữ liệu vào một file
* **f\_lseek** – Di chuyển con trong bên trong file để đọc ghi, mở rộng kích thước file.
* **f\_truncate** – Cắt giảm kích thước file
* **f\_stat** - Kiểm tra trạng thái của file hay thư mục
* **f\_opendir** – Mở một thư mục
* **f\_closedir** – Đóng một thư mục
* **f\_readdir** – Lấy thông tin về các file và thư mục con.
* **f\_mkdir** – Tạo thư mục con
* **f\_unlink** – Xóa một file hay một thư mục
  + 1. Lưu trữ file dưới dạng WAVE

Để dễ dàng thực hiện các thao tác truy xuất cũng như quản lí trên máy tính. Tín hiệu thu được sẽ được lưu dưới dạng WAVE, một chuẩn lưu trữ đa phương tiện phổ biến của Microsoft.



Hình 3.6 – Cấu trúc file dạng WAVE

Cấu trúc của một file dưới dạng WAVE như sau:

* Phần đặc tả RIFF: Chiều dài 12 byte. Chứa thông tin mặc định mô tả chuẩn định dạng.
* Phần mô tả đặc tính file: Chiều dài 24 byte. Chứa các thông tin về đặc tính của file như số kênh, tốc độ lấy mẫu, số bit mỗimẫu.
* Phần dữ liệu: chứa các đoạn dữ liệu với đoạn Subchunk2Size ở 8 byte đầu tiên cho biết dung lượng phần dữ liệu của file. Subchunk2Size được tính theo công thức: Số mẫu \* Số kênh \* Số bit mỗi mẫu/8

Dưới đây là các bước để thực hiện quá trình lưu trữ file dưới dạng WAVE trong phần mềm:

*Bước 1:*  Khởi tạo file mới.

*Bước 2:* Ghi trước thông tin đặc tả định dạng WAVE với 44 byte đầu tiên. Đoạn Subchunk2Size có giá trị bất kì.

*Bước 3:*Tiến hành ghi dữ liệu liên tục theo thứ tự bit là little-endian vào phần dữ liệu.

*Bước 4:* Tính toán Subchunk2Size.

*Bước 5:* Đọc dữ liệu 44 byte đầu tiên của file vào mảng tạm. Thực hiện thay đổi giá trị đoạn Subchunk2Size.

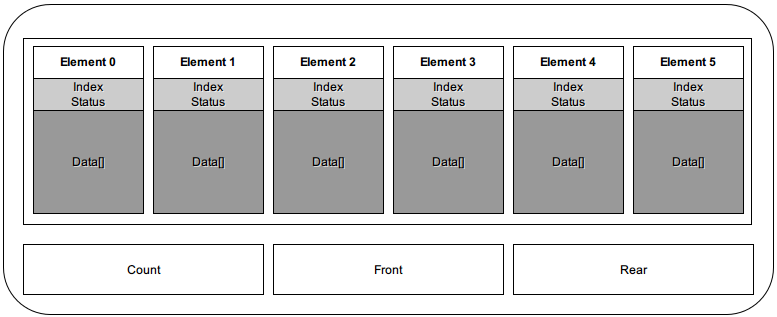
*Bước 6:* Ghi lại dữ liệu 44 byte đầu tiên. Đóng file.

* 1. Bộ đệm dữ liệu kiểu Vào trước ra trước (FIFO)
     1. Sự cần thiết của bộ đệm dữ liệu FIFO

Bộ đệm dữ liệu là một vùng dữ liệu được chia sẻ bởi các phần cứng hay các tiến trình khác nhau trong một phần mềm. Những thành phần này hoạt ở tốc độ khác nhau hay ở cấp độ ưu tiên khác nhau, bộ đệm được sử dụng để đảm bảo rằng hoạt động của tất cả các thành phần trong hệ thống không bị gián đoạn vì sự những khác biệt trên. Có hai kiểu đệm dữ liệu thông dụng đó là Vào sau ra trước (Last In Firt Out – LIFO) và Vào trước ra trước (First In First Out – FIFO), ứng với hai kiểu cấu trúc là ngăn xếp (stack) và hàng đợi (queue). Tính chất tự nhiên của hàng đợi cho thấy bộ đệm FIFO là kiểu đệm dữ liệu thích hợp cho một hệ thống thu phát tín hiệu thời gian thực.

* + 1. Thiết kế bộ đệm FIFO

Bộ đệm FIFO được thiết kế để đảm bảo sự lưu thông xuyên suốt của dữ liệu trong hai quá trình (1) Đệm dữ liệu cho bộ chuyển đổi A/D, (2) Đệm dữ liệu cho chức năng phát tín hiệu.



Hình 3.7 – Cấu trúc bộ đệm FIFO với 6 phần tử

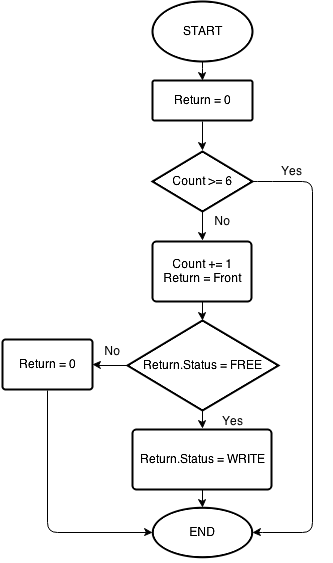
Bộ đệm FIFO trong thiết kế này cố định về dung lượng, dung lượng của bộ đệm được xác định tại thời điểm dịch chương trình. Kết cấu của bộ đệm về cơ bản bao gồm hai cấu trúc (C Structure) lồng nhau, với một cấu trúc chứa thông tin quản lí bộ đệm và cấu trúc còn lại chứa dữ liệu và thông tin của mỗi phần tử trong bộ đệm. Hình 3.7 mô tả cấu trúc của bộ đệm FIFO với 6 phần tử với các chi tiết như sau:

Ở cấu trúc quản lí:

* Count: đếm số lượng phần tử đang chiếm dụng. Giá trị của Count không bé hơn 0 và không lớn hơn tổng số phần tử được xác định của bộ đệm.
* Front: lưu giá trị Index của phần tử nằm ở đầu.
* Rear: lưu giá trị Index của phần tử nằm ở đuôi.

Ở cấu trúc phần tử:

* Index: chỉ thị vị trí của phần tử trong chuỗi các phần tử của bộ đệm.
* Status: chỉ thị trạng thái của phần tử. Có ba trạng thái:
  + FREE: phần tử được đang chờ được truy xuất.
  + READ: phần tử đang được đọc, không cho phép truy xuất tiếp theo.
  + WRITE: phần tử đang được ghi, không cho phép truy xuất tiếp theo.

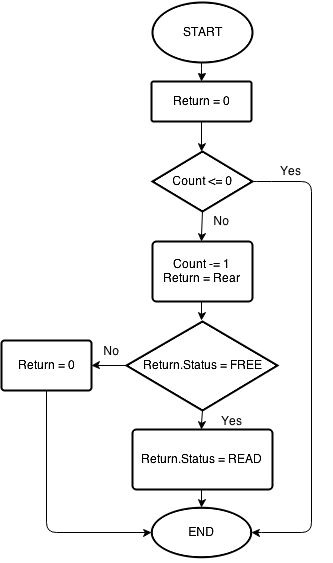


Hình 3.8 – Sơ đồ thuật toán yêu cầu phần tử trống từ bộ đệm FIFO

Quá trình yêu cầu một phần tử trống ở bộ đệm FIFO có 6 phần tử để thực hiện việc ghi dữ liệu lên được mô tả thông qua sơ đồ trên hình Hình 3.8.

Kết quả trả về Return là một biến dạng con trỏ ban đầu được gán giá trị rỗng. Nếu tất cả các phần tử của bộ đệm đều đã chứa dữ liệu chưa sử dụng (Count >= 6), thì giá trị trả về sẽ là rỗng. Nếu không, giá trị của Front sẽ được gán cho Return. Công việc tiếp theo là kiểm tra phần này đang bị chiếm dụng hay không. Nếu trạng thái của nó là FREE thì phần tử sẽ được trả về, hoặc giá trị là rỗng nếu ngược lại.

Việc yêu cầu một phần tử đã chứa dữ liệu ở bộ đệm cho thao tác đọc dữ liệu được thực hiện tương tự (Hình 3.9)

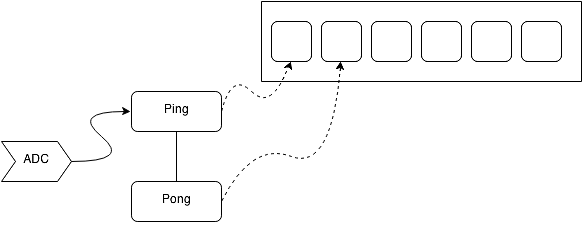


Hình 3.9 – Sơ đồ thuật toán yêu cầu phần tử chứa dữ liệu từ bộ đệm FIFO

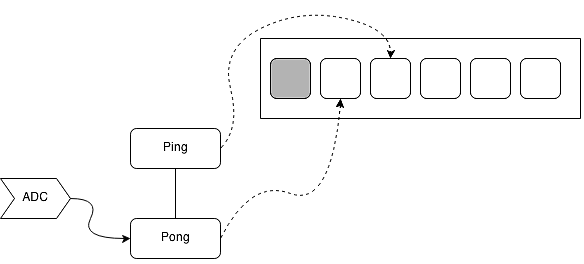
* + 1. Bộ đệm FIFO với μDMA cơ chế Ping-Pong

TM4C123GH6PM cung cấp bộ điều khiển truy xuất bộ nhớ trực tiếp (DMA) với tên gọi μDMA [13]. Chế độ hoạt động kiểu Ping-Pong của μDMA cho phép tự động chuyển đổi vùng đệm dữ liệu một cách tự động mà không cần sự can thiệp của phần mềm. Kết hợp giữa μDMA hoạt động chế độ Ping-Pong và bộ đệm FIFO giúp tối đa hóa khả năng hoạt động thời gian thực của hệ thống.

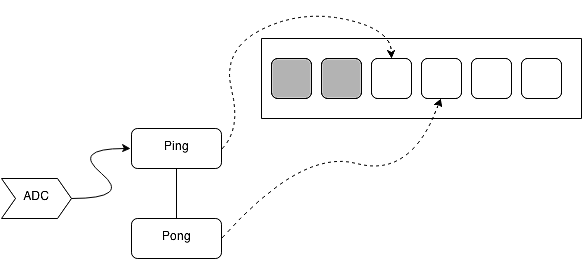
Quá trình lấy mẫu tín hiệu với μDMA hoạt động chế độ Ping-Pong và bộ đệm FIFO được mô tả lần lượt qua Hình 3.11, Hình 3.12 và Hình 3.13. Ban đầu, Ping và Pong lần lượt được trỏ đến hai phần tử đầu tiên của bộ đệm FIFO. ADC thực hiện lẫy mẫu và μDMA sẽ di chuyển dữ liệu vào Ping. Khi Ping đầy, μDMA tự động chuyển dữ liệu sang Pong, sau đó sẽ phát cờ ngắt để chương trình thiết lập Ping trỏ đến phần tử tiếp theo của bộ đệm.



Hình 3.10 – 𝜇DMA trạng thái sử dụng bộ đệm Ping



Hình 3.11 – 𝜇DMA trạng thái sử dụng bộ đệm Pong



Hình 3.12 – 𝜇DMA trở về trạng thái sử dụng bộ đệm Ping

Dữ liệu đã ghi ở bộ đệm FIFO sẽ được các tiến trình phần mềm khác xử lí. Các phần tử của bộ đệm FIFO đã được xử lí sau đó sẽ được trả về trạng thái FREE để có thể tiếp tục nhận dữ liệu mới. Quá trình này lặp lại tạo ra sự liên tục trong việc lấy mẫu tín hiệu.

* 1. Bộ lọc thông thấp đáp ứng xung chiều dài hữu hạn (FIR)
     1. Thiết kế nhân bộ lọc h[i] với Scipy:

Scipy là một hệ sinh thái dựa trên ngôn ngữ Python tập hợp các công cụ phần mềm hỗ trợ cho toán học, khoa học và kỹ thuật [30]. Về cơ bản Scipy bao gồm các công cụ sau:

* NumPy: gói hỗ trợ xử lí mảng N-chiều.
* Scipy library: thư viện nền cho các chức năng tính toán trong khoa học.
* Matplotlib: thư viện đồ họa 2 chiều
* IPython: giao diện tương tác dòng lệnh
* Sympy: thư viện kí tự trong toán học
* pandas: thư viện hỗ trợ về cấu trúc và phân tích dữ liệu

Scipy cung cấp thư viện *signal* để hỗ trợ việc thiết kế bộ lọc số. Một bộ lọc FIR có thê được thiết kế thông qua hàm *firwin* với giá trị trả về là mảng chứa các hệ số của nhân bộ lọc FIR:

**scipy.signal.firwin**(*numtaps, cutoff, width=None, window='hamming', pass\_zero=True, scale=True, nyq=1.0*)

Trong đó:

* numtaps: số phần tử của mảng, M + 1
* cutoff: tần số cắt, tính theo tỉ lệ chuẩn hóa với tần số lấy mẫu, cutoff nằm trong khoảng từ 0 đến ½ tần số lấy mẫu.
* width: tham số này không sử dụng nếu dùng cửa sổ
* window: loại cửa sổ sử dụng (Blackman, Hamming, …)
* scale: mặc định là True để đáp ứng có độ lợi là 1 ở dải thông của bộ lọc
* nyq: mặc định là 1.0

Bộ lọc cần thiết kế có tần số cắt là 3400 Hz với dải chuyển tiếp có độ rộng 600 Hz. Với tần số lấy mẫu là 32 KHz, từ 1.2.2 tính được chiều dài của nhân bộ lọc M là 214. Như vậy h[i] sẽ có tổng cộng 215 phần tử.

Sử dụng cửa sổ Blackman, việc tính toán h[i] được thực hiện thông qua mã nguồn sau:

sample\_rate = 32000.0

nyq\_rate = sample\_rate / 2.0

cutoff\_hz = 3400.0

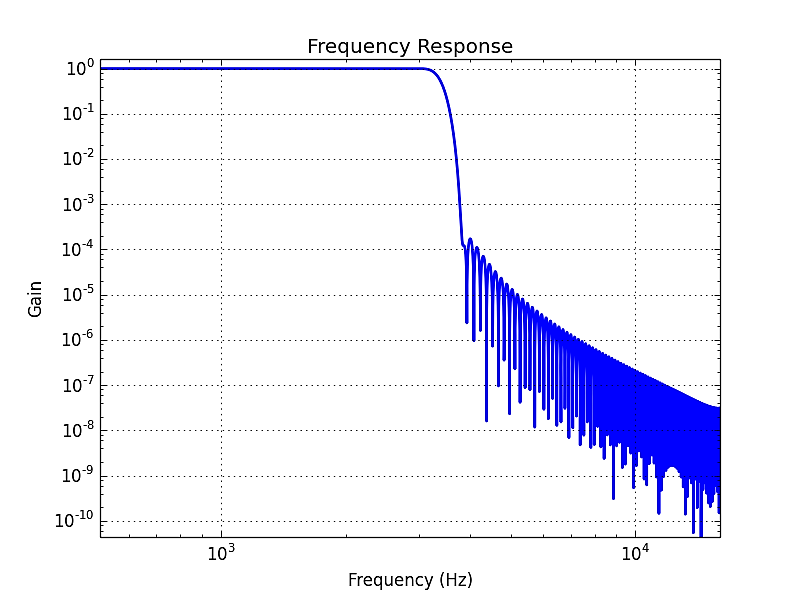
N = 215

taps = firwin(N, cutoff\_hz/nyq\_rate, window='blackman')

Kết quả được trình bày ở **Phụ lục I**.

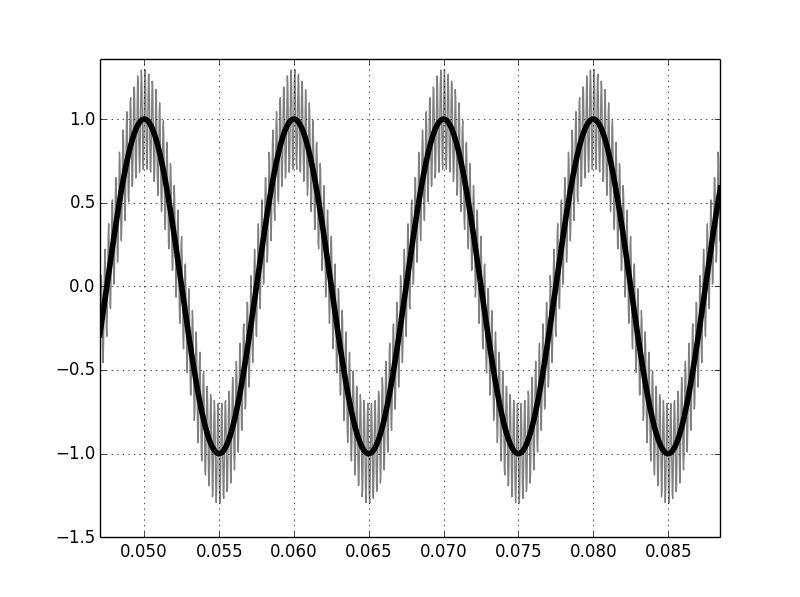
* + 1. Mô phỏng bộ lọc đã thiết kế trên Scipy

Đáp ứng tần số của bộ lọc trên hình Hình 3.13.



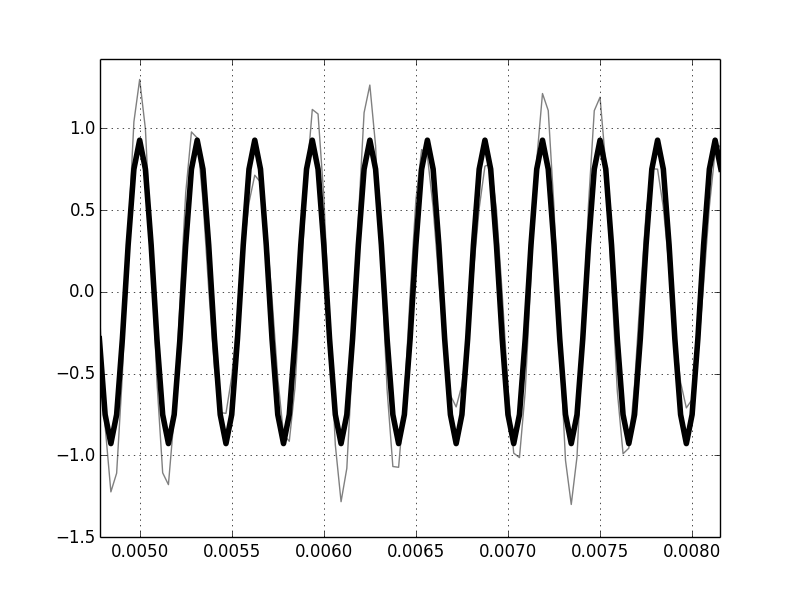
Hình 3.13 – Đáp ứng tần số của bộ lọc FIR 215 điểm

Kết quả mô phỏng bộ lọc với tín hiệu tổng hợp 100 Hz và 4100 Hz. Trên Hình 3.14, đường màu xám là tín hiệu ban đầu, đường màu đen là tín hiệu đã được lọc thành phần 4100 Hz.



Hình 3.14 – Bộ lọc với tín hiệu tổng hợp 100 Hz và 4100 Hz

Kết quả mô phỏng bộ lọc với tín hiệu tổng hợp 3200 Hz và 4050 Hz trên Hình 3.15 cũng cho thấy thành phần 4050 Hz đã được loại bỏ hoàn toàn.



Hình 3.15 – Bộ lọc với tín hiệu tổng hợp 3200 Hz và 4050 Hz

* + 1. Nhúng bộ lọc vào phần mềm điều khiển thông qua CMSIS

CMSIS hỗ trợ các thao tác xử lí tín hiệu số thông qua thư viện DSP Lib. Bộ lọc FIR có thể được thực hiện thông qua hàm **arm\_fir\_f32**.

void **arm\_fir\_f32** ( *const arm\_fir\_instance\_f32 \* S,*

*float32\_t \* pSrc,*

*float32\_t \* pDst,*

*uint32\_t blockSize*

)

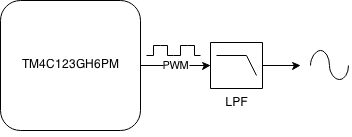
Trong đó:

* arm\_fir\_instance\_f32: trỏ tới mảng S chứa nhân của bộ lọc.
* pSrc: trỏ tới mảng chứa dữ liệu đầu vào
* pDst: trỏ tới mảng chứa dữ liệu đầu ra
* blockSize: số mẫu được xử lí trong một lần thực hiện hàm.

Các thao tác của **arm\_fir\_f32** là các thao tác trên dấu phẩy động kiểu độ chính xác đơn (single-precision floating point), trong khi đó giá trị lấy mẫu từ ADC của TM4C123GH6PM là kiểu nguyên. Do đó, để sử dụng hàm trên, việc cần thiết là phải chuyển đổi các giá trị dữ liệu tín hiệu vào từ số nguyên thành dấu phẩy động. Mặc dù DSP Lib cũng cung cấp các hàm xử lí tín hiệu số kiểu số nguyên, tuy nhiên sử dụng kiểu dấu phẩy động có các lợi thế cơ bản sau:

* Tỉ lệ Tín hiệu trên nhiễu (SNR) cao, nhiễu do lượng tử hóa thấp hơn 30000 lần so với việc dùng kiểu số nguyên [31].
* Tận dụng được khả năng hỗ trợ phần cứng của Đơn vị dấu phẩy động FPU để tăng tốc xử lí.
  1. Chuyển đổi số - tương tự D/A sử dụng kĩ thuật điều chế độ rộng xung PWM

Kĩ thuật điều chế độ rộng xung (PWM) là kĩ thuật điều chế tín hiệu theo phương pháp dữ liệu số được thể hiện dưới dạng tỉ lệ thời gian làm việc so với chu kì tín hiệu (còn gọi là duty cycle). Nếu duty cyle của tín hiệu PWM thay đổi theo thời gian, và tín hiệu đó được lọc thông qua bộ lọc thông thấp thì tín hiệu ở đầu ra sẽ là một tín hiệu tương tự. Đây chính là cơ sở để chuyển đổi số - tương tự D/A sử dụng PWM.



Hình 3.16 – Kĩ thuật PWM D/A

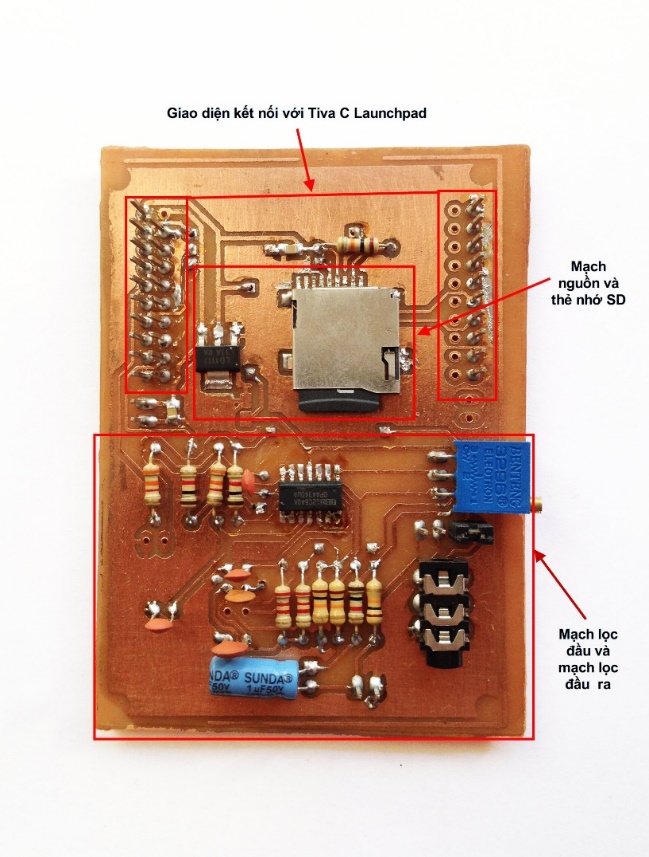
PWM trên TM4C123GH6PM được hỗ trợ bằng phần cứng, cho phép xuất đồng thời 16 kênh PWM ở đầu ra. Tín hiệu PWM cũng được xuất ra theo một chu kì nhất định là 8 KHz. Một bộ định thời đảm nhận công việc định thời gian xuất tín hiệu PWM ra, tương tự như ở mục 3.2.2. Với độ phân giải 10 bit, tần số của tín hiệu PWM khi tốc độ hệ thống ở 80 MHz là:

# MỘT SỐ KẾT QUẢ VÀ THẢO LUẬN

Chương này trình bày các kết quả từ đo đạc thực tế và đưa ra các thảo luận dựa trên những kết quả thu được.

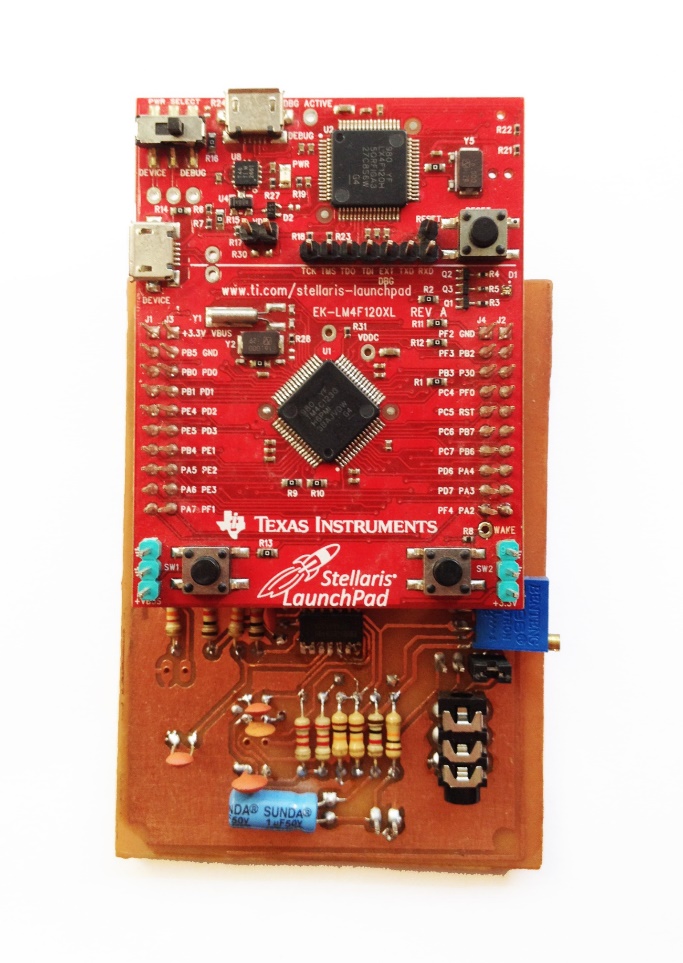
* 1. Phần cứng và giao diện phần mềm:

Phần mạch kết nối với Kit Tiva C Launchpad:



Hình 4.1 – Mạch kết nối với Kit Tiva C Launchpad

Kết nối hoàn chỉnh với Kit Tiva C Launchpad:



Hình 4.2 – Kết nối mạch với Kit Tiva C Launchpad

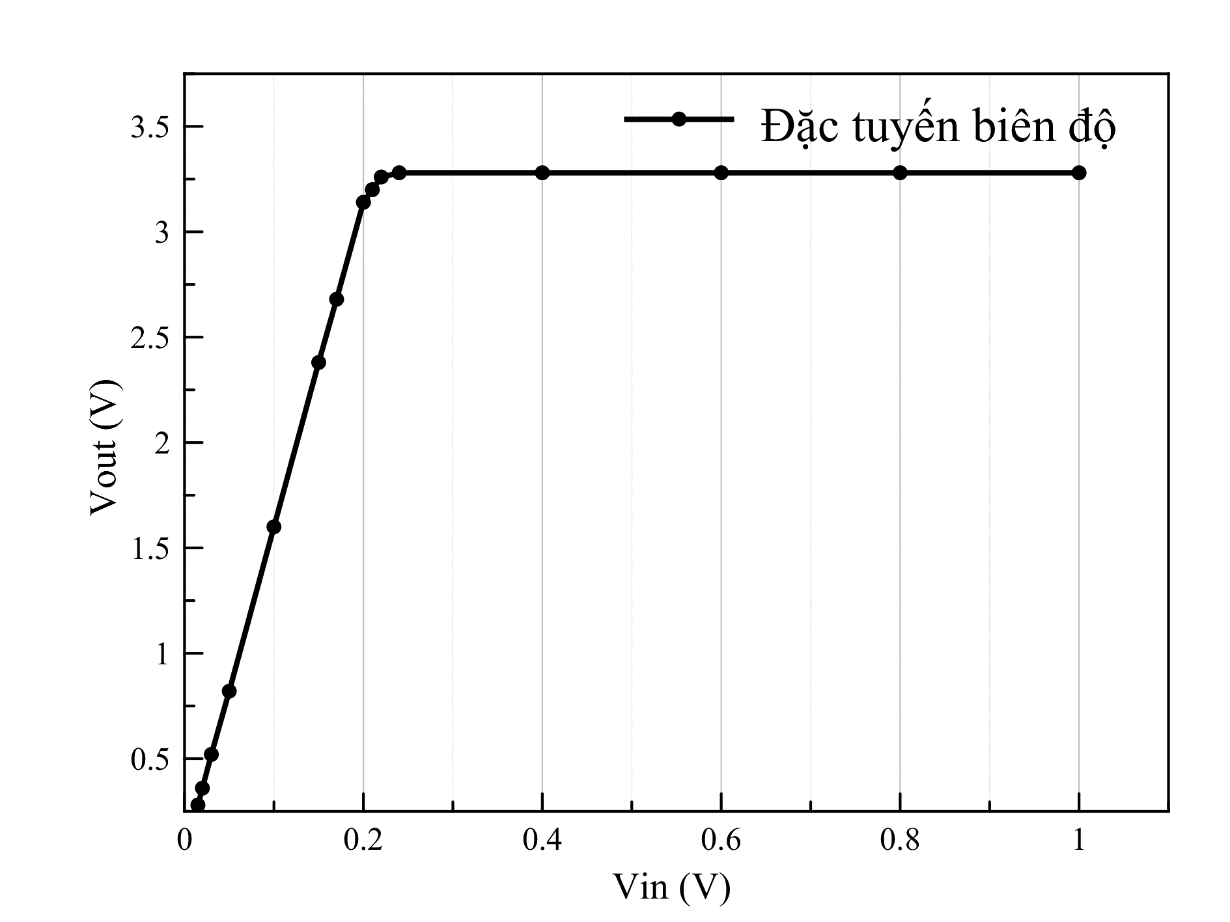
Giao diện điều khiển dòng lệnh thông qua truyền thông không đồng độ UART trên Hình 4.3.

Các lệnh được hỗ trợ hiện tại bao gồm:

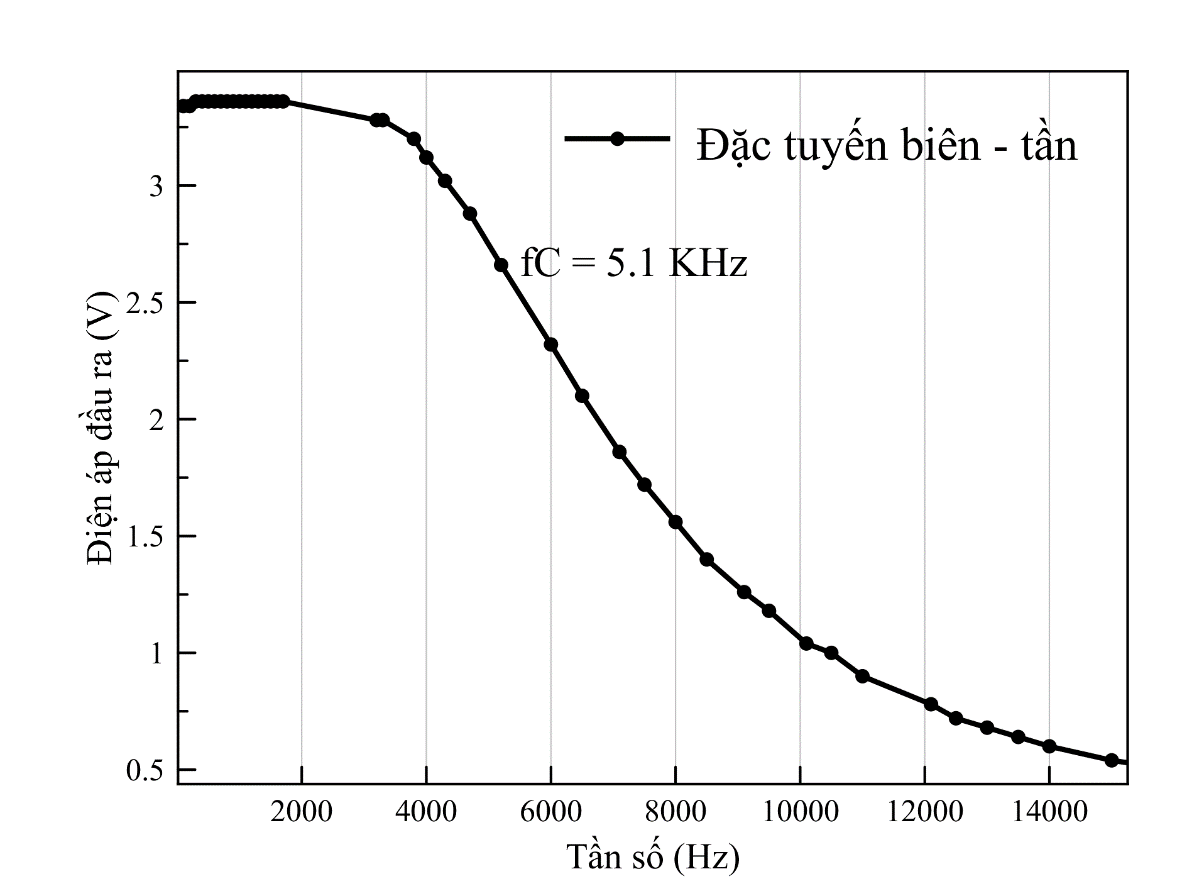
* help: Cung cấp thông tin về các câu lệnh
* ls: Liệt kê danh sách file và thư mục trên thẻ nhớ
* chdir: Thay đổi đến thư mục khác
* cd: Như chdir
* pwd: Hiển thị thư mục hiện tại
* play: Chơi một file dạng WAV
* rec: Ghi lại tín hiệu dưới dạng WAV
* 

Hình 4.3 – Giao diện dòng lệnh qua UART

* 1. Đặc tuyến của các mạch tích cực
     1. Đặc tuyến biên độ của mạch tiền khuếch đại



Hình 4.4 – Đặc tuyến biên độ mạch tiền khuếch đại



Hình 4.5 – Đặc tuyến biên – tần mạch khử chồng phổ AAF

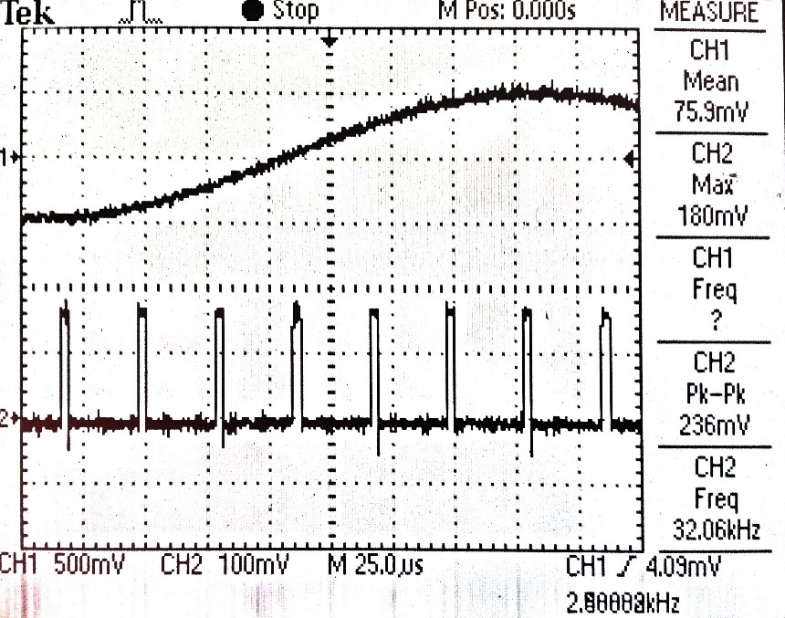
Hình 4.4 là đặc tuyến biên độ của mạch tiền khuếch đại. Giá trị biến trở R5 được thiết lập cố định là 18 kΩ, cho độ lợi là 18. Từ đồ thị trên thấy rằng:

* Vùng làm việc tuyến tính khi thiết lập giá trị độ lợi như trên là từ 0 đến 0.24 V. Giá trị này phù hợp với giá trị đầu ra của tín hiệu từ ECM, có biên độ không quá 200mV.
* Tín hiệu đạt trạng thái bảo hòa với biên độ là 3.28V, tiệm cận với giá trị điện áp của nguồn cấp.
  + 1. Đặc tuyến biên – tần của mạch khử chồng phổ AAF

Đặc tuyến biên tần của mạch khử chồng phổ AAF trên Hình 4.5. Kết quả đo đạc cho thấy tần số cắt của mạch, là 5.1 KHz. Điều này đảm bảo rằng các thành phần tần số trong khoảng DC đến 3.4 KHz đều được giữ lại.

* 1. Hoạt động thực tế của hệ thống
     1. Tốc độ lấy mẫu và tín hiệu PWM

Quá trình lẫy mẫu tín hiệu được thể hiện trên Hình 4.6. Tốc độ lấy mẫu ở đầu vào theo đúng yêu cầu là 32 KHz. Tốc độ lấy mẫu cho phép lấy mẫu tín hiệu với băng thông lớn nhất là 16 KHz mà không xảy ra hiện tượng chồng phổ tín hiệu.

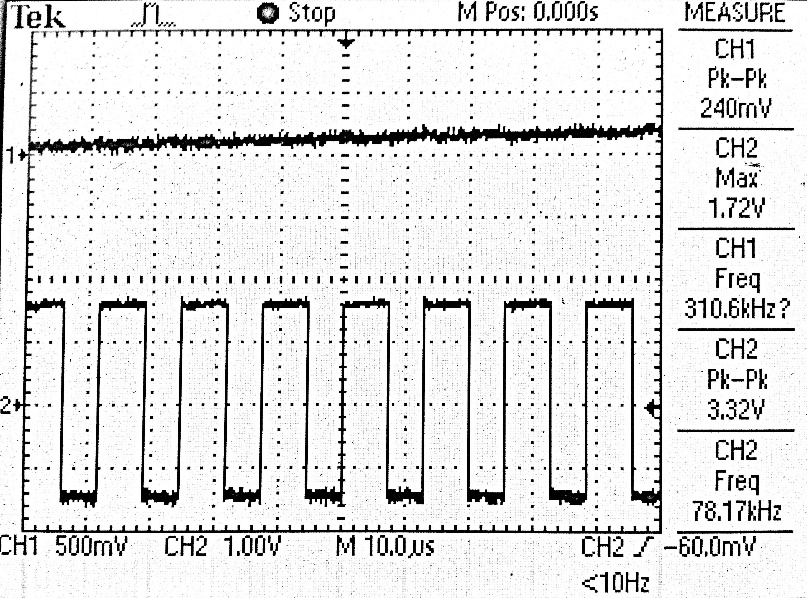


Hình 4.6 – Quá trình lấy mẫu tín hiệu

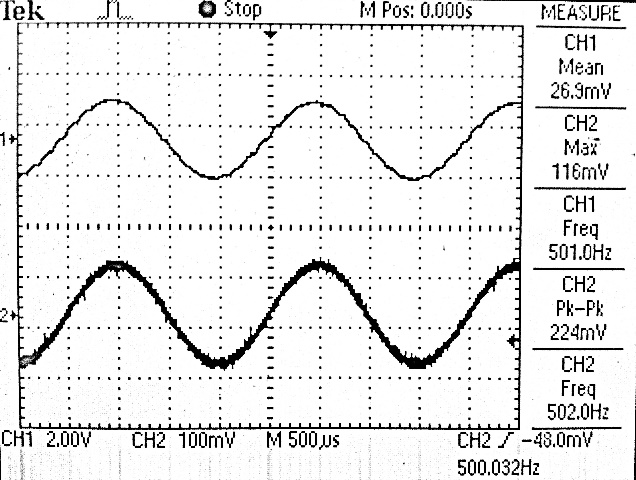
Ở Hình 4.7, kênh CH2 là tín hiệu PWM xuất ra từ vi điều khiển. Tần số của tín hiệu PWM là tương đương với tính toán ban đầu ở mục 3.7 là 78.2 KHz.

* + 1. Kiểm tra với các tín hiệu có tham số khác nhau:

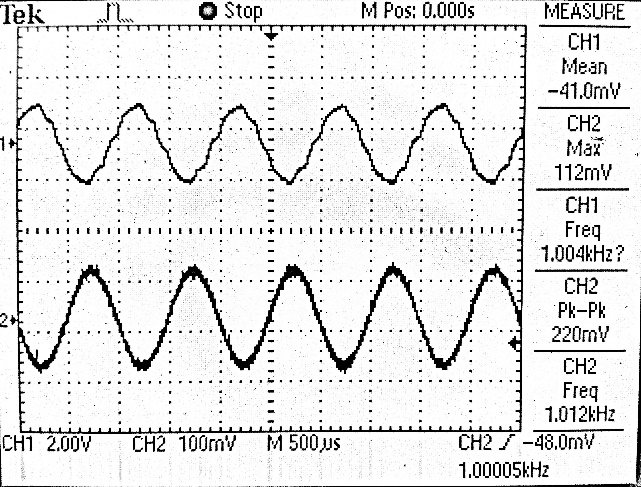
Từ Hình 4.8 đến Hình 4.11 là kết quả kiểm tra hệ thống với các tín hiệu đầu vào có tham số khác nhau. Tín hiệu đầu với được thể hiện ở kênh CH2, tín hiệu đầu ra ở kênh CH1. Các kết quả cho thấy đối với hệ thống làm việc tương đối tốt, tín hiệu đầu ra có dạng sóng tương tự tín hiệu đầu vào. Đối với tần số tín hiệu đầu vào 2000 Hz như trên hình Hình 4.11, tín hiệu đầu ra mặc dù vẫn có dạng sine nhưng đã bắt đầu có hiện tượng méo tín hiệu. Điều này xuất hiện do hạn chế của cơ chế PWM D/A và bộ lọc tái tạo.



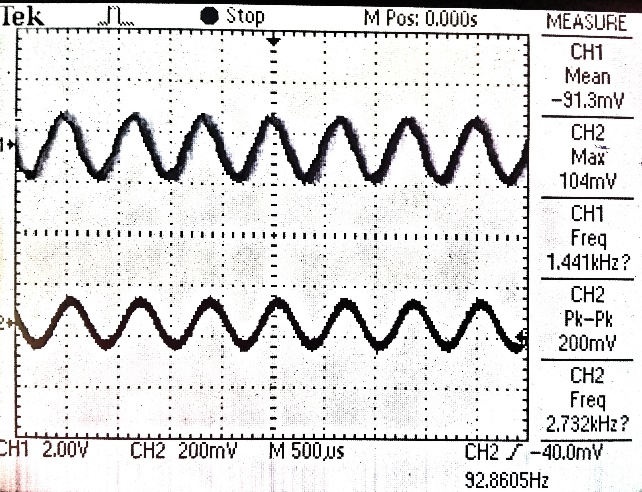
Hình 4.7 – Tín hiệu PWM



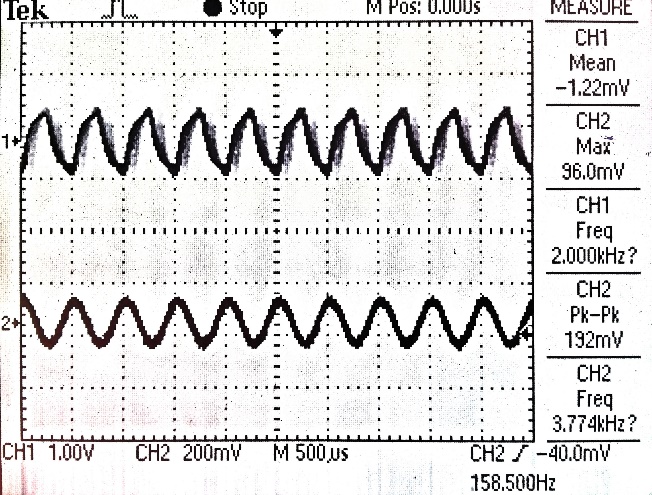
Hình 4.8 – Tín hiệu sine 500 Hz



Hình 4.9 – Tín hiệu sine 1000 Hz

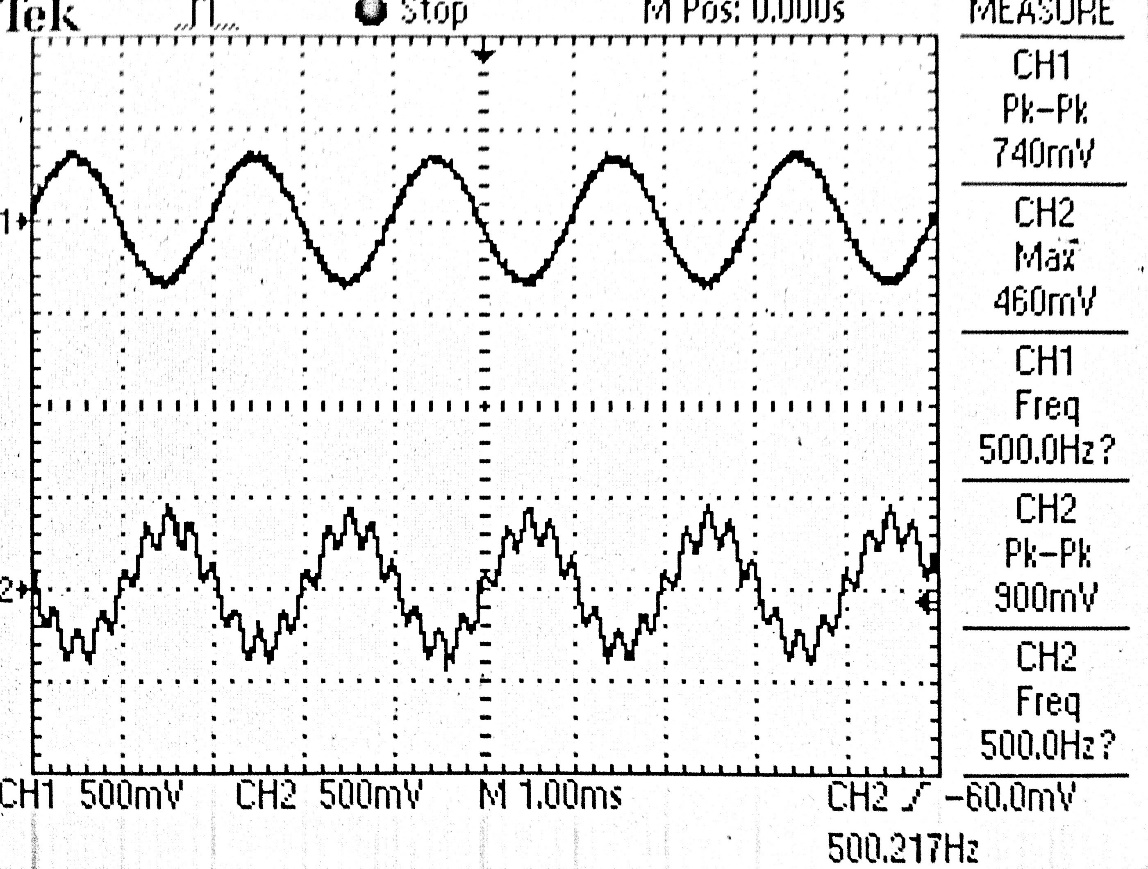


Hình 4.10 – Tín hiệu sine 1500 Hz



Hình 4.11 – Tín hiệu sine 2000 Hz

* + 1. Hoạt động của hệ thống với tín hiệu và tạp âm:



Hình 4.12 – Hoạt động của hệ thống với tín hiệu và tạp âm

Trên Hình 4.12, một máy phát tín hiệu phát tín hiệu tổng hợp gồm tín hiệu sine 500 Hz và tín hiệu sine 4050 Hz. Trường hợp này, tín hiệu 4050 Hz nằm ngoài băng tần 3400 Hz của thiết kế nên được xem như là tạp âm. Tín hiệu tổng hợp được thể hiện kênh CH2. Kênh CH1 là tín hiệu được tái tạo lại bởi hệ thống sau khi đã thu tín hiệu. Kết quả cho thấy, thành phần tạp âm đã được loại bỏ khỏi tín hiệu ban đầu.

KẾT LUẬN VÀ ĐỀ XUẤT

Quá trình bốn tháng nghiên cứu và thực hiện đồ án đã thu được một số kết quả như sau:

* Hoàn thành việc thiết kế và thi công mạch kết nối với kit Tiva C Launchpad.
* Viết phần mềm điều khiển của hệ thống với các module phần mềm thực hiện thao tác lấy mẫu tín hiệu, xử lí tín hiệu và quản lí dữ liệu trên thẻ nhớ SD.
* Thực hiện việc khảo sát một vài tín hiệu đặc trưng với các tham số khác nhau nhằm kiểm tra hiệu năng của hệ thống.

Các kết quả đạt được cho thấy mục đích ban đầu đề ra của đồ án đã được hoàn thành. Tuy nhiên, thời gian thực hiện đồ án ngắn không cho phép áp dụng các kĩ thuật thiết kế phức tạp hơn hay tinh chỉnh phần mềm để đạt được hiệu năng tối ưu nhất cho hệ thống. Dưới đây là những đề xuất có thể được triển khai để phát triển thêm thiết kế trong thời gian tới:

* Cấp phát bộ nhớ động ở bộ đệm FIFO nhằm tạo điều kiện thuận lợi cho việc nâng cấp phần mềm hệ thống để đáp ứng các yêu cầu xử lí với tốc độ cao hơn.
* Nâng tốc độ lấy mẫu đầu vào cùng với việc nâng cao hiệu năng của bộ lọc số với mục đích: tối ưu hóa hiệu năng hệ thống hoạt động với băng tần hiện tại là 3.4 KHz; phát triển hệ thống nhằm đáp ứng băng tần 7 KHz theo tiêu chuẩn âm thanh chất lượng cao HD voice.
* Thay thế phương pháp PWM D/A bằng việc sử dụng DAC chuyên biệt, cùng với đó thay thế mạch tái tạo tín hiệu với tính chất tốt hơn nhằm nâng cao chất lượng tín hiệu ở đầu ra.

Ngoài ra, việc hoàn thành đồ án cũng là cơ sở mở đường cho các bước nghiên cứu tiếp theo trong lĩnh vực xử lí tín hiệu băng tần tiếng nói trong tương lai.

Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | K. Wurm, "Skype and a New Audio Codec," 2012. [Online]. Available: http://blogs.skype.com/2012/09/12/skype-and-a-new-audio-codec/. |
| [2] | "Cortex-M4 Processor," [Online]. Available: http://www.arm.com/products/processors/cortex-m/cortex-m4-processor.php. |
| [3] | Hartmut Traunmüller, Anders Eriksson, "The frequency range of the voice fundamental in the speech of male and female adults," 1994. |
| [4] | ProAV, "proAV / data and information, lists, tables and links," [Online]. Available: http://www.bnoack.com/index.html. |
| [5] | B. P. Lathi, Zhi Ding, "Pulse Code Modulation (PCM)," in *Modern Digital and Analog Communication Systems (Oxford Series in Electrical and Computer Engineering)*, 2009, p. 263. |
| [6] | R. Nave, "Missing Fundamental Effect," [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/sound/subton.html#c2. |
| [7] | W. Kester, "Digital Filters," Analog Devices. |
| [8] | S. W. Smith, "Introduction to Digital Filters," in *The Scientist & Engineer's Guide to Digital Signal Processing*, California Technical Pub, 1997, pp. 261-262. |
| [9] | "FIR Filter Basics," [Online]. Available: http://www.dspguru.com/dsp/faqs/fir/basics. |
| [10] | S. W. Smith, "Designing the Filter," in *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997, pp. 288-290. |
| [11] | ARM, "About the Cortex-M4 processor and core peripherals," in *Cortex-M4 Devices Generic User Guide*, 2010, pp. 1-2. |
| [12] | ARM, "Floating Point Unit (FPU)," in *Cortex-M4 Devices Generic User Guide*, 2010, pp. 4-48. |
| [13] | "Tiva TM4C123GH6PM Microcontroller Datasheet," Inc., Texas Instruments, 2013. |
| [14] | ARM, "Cortex Microcontroller Software Interface Standard," ARM, [Online]. Available: http://www.keil.com/pack/doc/cmsis/General/html/index.html. |
| [15] | R. Mancini, "Single-supply op amp design," *Analog Applications Journal,* 1991. |
| [16] | "LD1117 Series," STMicroelectronics, 2005. |
| [17] | A. V. Rhijn, "Integrated Circuits for High Performance Electret Microphones," Inc., Texas Instruments, 2011. |
| [18] | "Insight - How Electret Condenser Microphone works," EngineersGarage, [Online]. Available: http://www.engineersgarage.com/insight/how-electret-condenser-microphone-works. |
| [19] | "What is an Electret Microphone?," [Online]. Available: http://www.learningaboutelectronics.com/Articles/Electret-microphones. |
| [20] | "OPA4340 Datasheet," Inc., Texas Instruments, 2007. |
| [21] | Bruce Carter and Ron Macini, Opamps for Everyone, Elsevier Inc., 2009. |
| [22] | "SPICE-Based Analog Simulation Program - TINA-TI - TI Software Folder," Inc., Texas Instruments, [Online]. Available: http://www.ti.com/tool/tina-ti. |
| [23] | "Tiva C Series TM4C123G LaunchPad (EK-TM4C123GXL)," Inc., Texas Instruments, 2014. [Online]. Available: http://www.ti.com/ww/en/launchpad/launchpads-connected-ek-tm4c123gxl.html#tabs. |
| [24] | "C Series for Connected MCUs," Inc., Texas Instruments, [Online]. Available: http://www.ti.com/tiva-c. |
| [25] | "Active Filter Design Application - FILTERPRO - TI Software Folder," Inc., Texas Instruments, 2014. [Online]. Available: http://www.ti.com/tool/filterpro. |
| [26] | M. Corporation, "Microsoft Extensible Firmware Initiative FAT32 File System Specification," Microsoft Corporation. |
| [27] | Peter Reen and Naveen Mohanswamy, "Implementing File I/O Functions Using Microchip’s Memory," Microchip Technology, 2008. |
| [28] | "FatFs - Generic FAT File System Module," [Online]. Available: http://elm-chan.org/fsw/ff/00index\_e.html. |
| [29] | "FatFs Module Application Note," [Online]. Available: http://elm-chan.org/fsw/ff/en/appnote.html. |
| [30] | SciPy developers, "SciPy.org — SciPy.org," [Online]. Available: http://scipy.org/. |
| [31] | S. W. Smith, "Fixed versus Floating Point," in *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997, pp. 514-520. |

PHỤ LỤC I – NHÂN BỘ LỌC THÔNG THẤP FIR TRONG THIẾT KẾ

-0.0000000000f, +0.0000002323f, +0.0000007832f, +0.0000006623f,

-0.0000013344f, -0.0000052010f, -0.0000088389f, -0.0000086861f,

-0.0000019135f, +0.0000109305f, +0.0000246171f, +0.0000306960f,

+0.0000217068f, -0.0000036617f, -0.0000374144f, -0.0000636747f,

-0.0000652397f, -0.0000329458f, +0.0000267322f, +0.0000906192f,

+0.0001266479f, +0.0001085528f, +0.0000312376f, -0.0000812237f,

-0.0001809153f, -0.0002150066f, -0.0001514953f, -0.0000000000f,

+0.0001840694f, +0.0003175232f, +0.0003249826f, +0.0001776711f,

-0.0000833346f, -0.0003538823f, -0.0005057655f, -0.0004446306f,

-0.0001617323f, +0.0002468301f, +0.0006083376f, +0.0007434267f,

+0.0005508558f, +0.0000685616f, -0.0005221625f, -0.0009603081f,

-0.0010166269f, -0.0006069718f, +0.0001462305f, +0.0009412564f,

+0.0014130646f, +0.0012957560f, +0.0005612871f, -0.0005343108f,

-0.0015316759f, -0.0019550690f, -0.0015320035f, -0.0003469578f,

+0.0011513379f, +0.0023113232f, +0.0025549538f, +0.0016547575f,

-0.0001171513f, -0.0020530995f, -0.0032832550f, -0.0031571081f,

-0.0015695511f, +0.0009256536f, +0.0032924294f, +0.0044315507f,

+0.0036777162f, +0.0011550680f, -0.0021882817f, -0.0049191898f,

-0.0057191265f, -0.0039998024f, -0.0002553880f, +0.0040390115f,

+0.0069873984f, +0.0070881679f, +0.0039629010f, -0.0013417470f,

-0.0066640387f, -0.0095781522f, -0.0084634882f, -0.0033352546f,

+0.0039649501f, +0.0103786814f, +0.0128614278f, +0.0097586557f,

+0.0017315711f, -0.0082336439f, -0.0158512626f, -0.0172755251f,

-0.0108842665f, +0.0016640160f, +0.0156537114f, +0.0248954297f,

+0.0241880976f, +0.0117573361f, -0.0093288647f, -0.0314689127f,

-0.0446569404f, -0.0398304505f, -0.0123105452f, +0.0359236943f,

+0.0960510962f, +0.1545391329f, +0.1969944767f, +0.2125004224f,

+0.1969944767f, +0.1545391329f, +0.0960510962f, +0.0359236943f,

-0.0123105452f, -0.0398304505f, -0.0446569404f, -0.0314689127f,

-0.0093288647f, +0.0117573361f, +0.0241880976f, +0.0248954297f,

+0.0156537114f, +0.0016640160f, -0.0108842665f, -0.0172755251f,

-0.0158512626f, -0.0082336439f, +0.0017315711f, +0.0097586557f,

+0.0128614278f, +0.0103786814f, +0.0039649501f, -0.0033352546f,

-0.0084634882f, -0.0095781522f, -0.0066640387f, -0.0013417470f,

+0.0039629010f, +0.0070881679f, +0.0069873984f, +0.0040390115f,

-0.0002553880f, -0.0039998024f, -0.0057191265f, -0.0049191898f,

-0.0021882817f, +0.0011550680f, +0.0036777162f, +0.0044315507f,

+0.0032924294f, +0.0009256536f, -0.0015695511f, -0.0031571081f,

-0.0032832550f, -0.0020530995f, -0.0001171513f, +0.0016547575f,

+0.0025549538f, +0.0023113232f, +0.0011513379f, -0.0003469578f,

-0.0015320035f, -0.0019550690f, -0.0015316759f, -0.0005343108f,

+0.0005612871f, +0.0012957560f, +0.0014130646f, +0.0009412564f,

+0.0001462305f, -0.0006069718f, -0.0010166269f, -0.0009603081f,

-0.0005221625f, +0.0000685616f, +0.0005508558f, +0.0007434267f,

+0.0006083376f, +0.0002468301f, -0.0001617323f, -0.0004446306f,

-0.0005057655f, -0.0003538823f, -0.0000833346f, +0.0001776711f,

+0.0003249826f, +0.0003175232f, +0.0001840694f, -0.0000000000f,

-0.0001514953f, -0.0002150066f, -0.0001809153f, -0.0000812237f,

+0.0000312376f, +0.0001085528f, +0.0001266479f, +0.0000906192f,

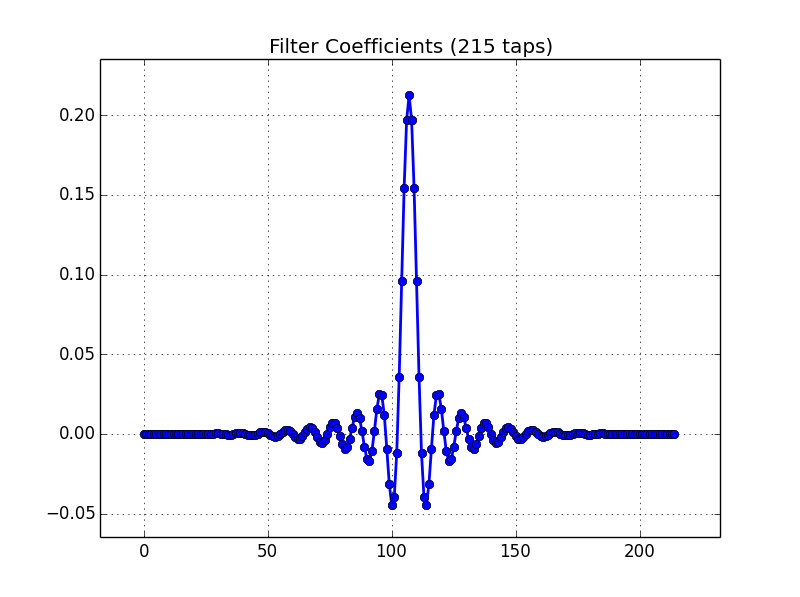
+0.0000267322f, -0.0000329458f, -0.0000652397f, -0.0000636747f,

-0.0000374144f, -0.0000036617f, +0.0000217068f, +0.0000306960f,

+0.0000246171f, +0.0000109305f, -0.0000019135f, -0.0000086861f,

-0.0000088389f, -0.0000052010f, -0.0000013344f, +0.0000006623f,

+0.0000007832f, +0.0000002323f, -0.0000000000f



Hình 4.13 – Nhân bộ lọc thông thấp FIR trong thiết kế