ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ
----------o0o----------

BÁO CÁO BÀI TẬP LỚN
XỬ LÝ TÍN HIỆU SỐ VỚI FPGA

GVHD: ThS. Nguyễn Tuấn Hùng

TP. HỒ CHÍ MINH, THÁNG NĂM 2024

ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ
----------o0o----------

THIẾT KẾ BỘ LỌC FIR

SVTH:

Tên thành viên	MSSV

TP. HỒ CHÍ MINH, THÁNG NĂM 2024

1. Giới thiệu:

1.1 Giới thiệu về bộ lọc:

Lọc tín hiệu là một trong những hoạt động quan trọng trong xử lý tín hiệu, đóng vai trò thiết yếu trong việc tách, giảm nhiễu và tăng cường các tín hiệu mong muốn. Có hai loại bộ lọc chính: bộ lọc tương tự và bộ lọc số. Bộ lọc tương tự xử lý tín hiệu liên tục và thường được xây dựng từ các linh kiện như khuếch đại thuật toán, điện trở và tụ điện. Ngược lại, bộ lọc số xử lý tín hiệu thời gian rời rạc, cho phép thực hiện các phép toán phức tạp thông qua các thuật toán số.

Quá trình lọc số bắt đầu bằng việc chuyển đổi tín hiệu tương tự thành tín hiệu số bằng bộ chuyển đổi A/D (Analog to Digital). Sau đó, các mẫu số được xử lý qua bộ lọc và kết quả được chuyển đổi lại thành tín hiệu tương tự thông qua bộ chuyển đổi D/A (Digital to Analog). Bộ lọc số có nhiều ưu điểm vượt trội so với bộ lọc tương tự như độ tin cậy cao, độ chính xác cao hơn và ít bị ảnh hưởng bởi các yếu tố môi trường như nhiệt độ hoặc sự lão hóa linh kiện. Hơn nữa, các tham số như tần số cắt, băng thông và loại bộ lọc có thể dễ dàng điều chỉnh theo yêu cầu mà không cần thay đổi phần cứng.

1.2 Bộ lọc FIR:

Bộ lọc FIR (Finite Impulse Response) là một loại bộ lọc số được sử dụng rộng rãi trong xử lý tín hiệu số. Bộ lọc FIR có đặc điểm nổi bật là đáp ứng xung hữu hạn, nghĩa là đầu ra của bộ lọc sẽ trở về 0 sau một khoảng thời gian nhất định khi không còn tín hiệu đầu vào. Điều này làm cho bộ lọc FIR luôn ổn định và không bị ảnh hưởng bởi các vấn đề như độ trễ pha hoặc phản hồi ngược như trong các bộ lọc IIR (Infinite Impulse Response).

Một trong những ưu điểm chính của bộ lọc FIR là khả năng đảm bảo tính tuyến tính pha. Điều này có nghĩa là tất cả các tần số trong tín hiệu được lọc sẽ trải qua cùng một độ trễ, giúp bảo toàn được hình dạng ban đầu của tín hiệu mà không gây ra sự biến dạng. Do đó, FIR thường được sử dụng trong các ứng dụng yêu cầu độ chính xác cao về pha như xử lý âm thanh, viễn thông và radar.

Thiết kế bộ lọc FIR cũng rất linh hoạt, cho phép thay đổi các tham số như tần số cắt và băng thông một cách dễ dàng mà không cần phải điều chỉnh phần cứng. Các phương pháp thiết kế bộ lọc FIR phổ biến bao gồm sử dụng các hàm cửa sổ (window functions) như Hamming, Hanning, hoặc Blackman để kiểm soát đáp ứng tần số của bộ lọc.

Bộ lọc FIR có thể được triển khai trên nhiều nền tảng phần cứng và phần mềm, từ các bộ vi xử lý tín hiệu số (DSP) đến các FPGA, hay ngay cả trong các phần mềm xử lý tín hiệu như MATLAB hoặc Python. Với khả năng tùy biến cao và hiệu suất vượt trội, bộ lọc FIR đóng vai trò quan trọng trong nhiều hệ thống xử lý tín hiệu hiện đại.

1.3 Phương pháp Hamming Window:

Phương pháp Hamming Window là một trong những kỹ thuật phổ biến được sử dụng trong thiết kế bộ lọc FIR để giảm thiểu hiện tượng rò rỉ tần số (spectral leakage). Khi một tín hiệu được xử lý bằng bộ lọc số, việc cắt ngắn hoặc giới hạn chuỗi mẫu sẽ tạo ra các hiệu ứng không mong muốn trong miền tần số, gây ra sự lan truyền năng lượng từ các tần số này sang các tần số khác. Để giảm thiểu vấn đề này, các hàm cửa sổ (window functions) như Hamming được sử dụng để làm trơn biên của chuỗi mẫu.

Hàm Hamming Window được định nghĩa bằng công thức:

$$W(n) = 0.54 - 0.46.\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n \le N-1$$

Trong đó:

- $W(n)$ là hệ số cửa sổ tại vị trí n.
- N là số lượng mẫu của cửa sổ.

Phương pháp này tạo ra một dạng biên mịn hơn cho chuỗi tín hiệu, làm giảm các tác động tiêu cực của các dao động phụ bên ngoài băng thông mong muốn. Hamming Window có ưu điểm là tạo ra độ gợn nhỏ trong băng dừng (stopband), tức là khả năng loại bỏ tín hiệu ngoài băng thông của bộ lọc được cải thiện mà không làm giảm đáng kể hiệu quả lọc trong băng thông (passband).

Đặc điểm chính của Hamming Window:

- **Giảm rò rỉ tần số**: Làm giảm năng lượng rò rỉ vào các tần số không mong muốn.
- **Hiệu quả lọc tốt trong cả hai miền**: Hamming Window cung cấp sự cân bằng giữa độ rộng băng chuyển tiếp (transition band) và độ gợn trong băng thông (ripple).
- **Ứng dụng**: Hamming Window thường được sử dụng trong các ứng dụng như thiết kế bộ lọc số, phân tích Fourier, và các ứng dụng xử lý tín hiệu liên quan đến việc cắt ngắn tín hiệu.

Mặc dù không hoàn hảo về mọi mặt, Hamming Window là một lựa chọn tối ưu trong các tình huống yêu cầu sự cân bằng giữa độ chính xác tần số và hiệu suất lọc, thường được sử dụng khi cần loại bỏ hiện tượng rò rỉ tần số nhưng vẫn duy trì chất lượng tín hiệu ở mức chấp nhận được.

2. Nguyên lý hoạt động của bộ lọc FIR:

Bộ lọc FIR (Finite Impulse Response) là một trong hai loại bộ lọc số phổ biến trong xử lý tín hiệu, bên cạnh bộ lọc IIR (Infinite Impulse Response). Bộ lọc FIR hoạt động trên cơ sở xử lý một chuỗi tín hiệu rời rạc thông qua một hàm phản hồi xung hữu hạn. Nguyên lý hoạt động của bộ lọc FIR dựa trên tính toán tích chập (convolution) giữa tín hiệu đầu vào và các hệ số của bộ lọc, còn được gọi là hằng số lọc hay trọng số lọc.

2.1. Cấu trúc toán học của bộ lọc FIR:

Công thức cơ bản của bộ lọc FIR được biểu diễn dưới dạng:

$$y[n] = \sum_{k=0}^{N-1} h[k] \,.\, x[n-k]$$

Trong đó :

- $y[n]$là đầu ra của bộ lọc tại thời điểm n

- $x[n-k]$ là tín hiệu đầu vào tại thời điểm $n-k$
- $h[k]$ là các hệ số của bộ lọc FIR (hay còn gọi là hằng số lọc)
- N là số bậc của bộ lọc (hay số lượng hệ số của bộ lọc)

Bộ lọc FIR là một bộ lọc có đáp ứng xung hữu hạn, nghĩa là đầu ra của bộ lọc chỉ phụ thuộc vào một số lượng hữu hạn các mẫu đầu vào trước đó. Đây là điểm khác biệt so với bộ lọc IIR, trong đó đầu ra còn phụ thuộc vào các giá trị đầu ra trước đó (phản hồi ngược). Tính chất này làm cho bộ lọc FIR luôn ổn định, không bị dao động dài hạn như các bộ lọc IIR.

2.2. Hoạt động của bộ lọc FIR

Quá trình hoạt động của bộ lọc FIR bao gồm ba bước chính: lấy mẫu tín hiệu đầu vào, thực hiện tích chập giữa tín hiệu và các hệ số của bộ lọc, và đưa ra tín hiệu đầu ra.

Bước 1: Lấy mẫu tín hiệu đầu vào:

Tín hiệu đầu vào $x[n]$ thường là một tín hiệu rời rạc, có thể là kết quả của quá trình chuyển đổi từ tín hiệu tương tự sang tín hiệu số thông qua bộ biến đổi A/D. Các mẫu tín hiệu này sẽ được lưu trữ trong một bộ đệm để chuẩn bị cho quá trình tính toán tích chập.

Bước 2: Tích chập tín hiệu với hệ số bộ lọc:

Mỗi mẫu đầu vào $x[n-k]$ sẽ được nhân với hệ số tương ứng $h[k]$ của bộ lọc. Kết quả của các phép nhân này sẽ được cộng lại để tạo ra một mẫu đầu ra duy nhất $y[n]$. Quá trình này là một dạng tính toán tích chập, trong đó tín hiệu đầu vào được lọc qua các trọng số của bộ lọc.

Bước 3: Xuất tín hiệu đầu ra:

Sau khi thực hiện tích chập, tín hiệu đầu ra $y[n]$ sẽ được tính toán và xuất ra dưới dạng tín hiệu số. Tín hiệu này có thể tiếp tục được xử lý hoặc chuyển đổi trở lại dạng tín hiệu tương tự thông qua bộ chuyển đổi D/A để sử dụng trong các hệ thống tương tự.

2.3. Ưu điểm và tính chất của bộ lọc FIR

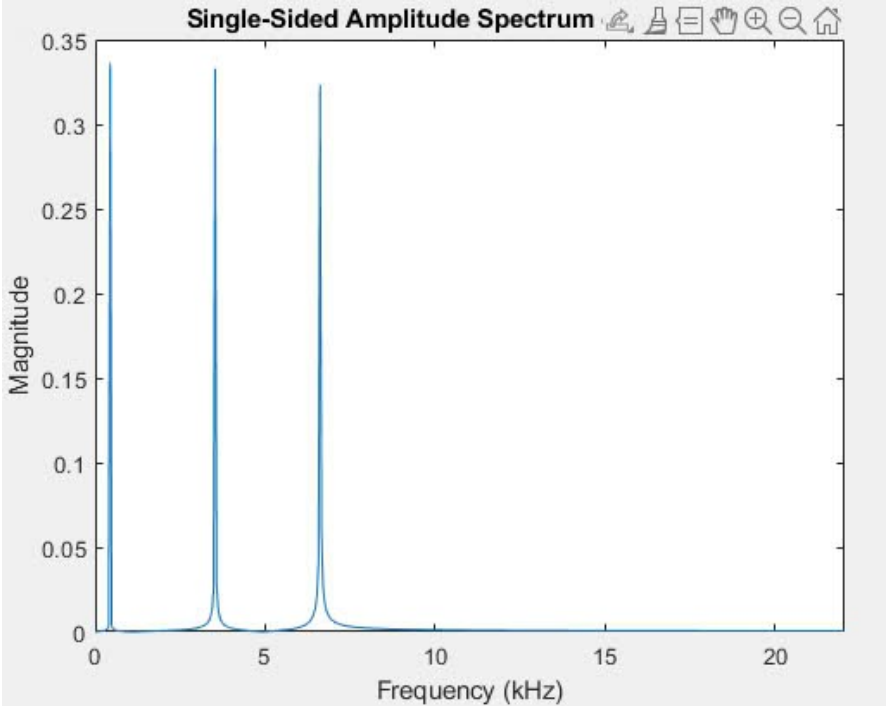
Bộ lọc FIR có một số ưu điểm và tính chất nổi bật, khiến nó trở thành lựa chọn phổ biến trong nhiều ứng dụng xử lý tín hiệu số:

- **Pha tuyến tính**: Một trong những ưu điểm quan trọng nhất của bộ lọc FIR là khả năng đảm bảo pha tuyến tính. Điều này có nghĩa là mọi tần số trong tín hiệu đều trải qua cùng một độ trễ, giúp bảo toàn được hình dạng tín hiệu mà không gây ra hiện tượng méo pha.
- **Tính ổn định tuyệt đối**: Do bộ lọc FIR không có phản hồi ngược, đầu ra của nó chỉ phụ thuộc vào tín hiệu đầu vào hiện tại và các mẫu đầu vào trước đó trong một khoảng thời gian hữu hạn. Điều này giúp bộ lọc FIR luôn ổn định, không xảy ra hiện tượng dao động dài hạn như trong bộ lọc IIR.
- **Thiết kế dễ dàng**: Quá trình thiết kế bộ lọc FIR rất linh hoạt và đơn giản. Các hệ số lọc có thể được tính toán dễ dàng bằng nhiều phương pháp khác nhau, chẳng hạn như phương pháp cửa sổ (windowing method), phương pháp tối ưu hóa hoặc sử dụng các phần mềm chuyên dụng như MATLAB.
- **Đáp ứng tần số điều chỉnh dễ dàng**: Bằng cách điều chỉnh các hệ số lọc, người dùng có thể dễ dàng kiểm soát các thông số của bộ lọc FIR như tần số cắt, băng thông hoặc hình dạng của đáp ứng tần số.

3. Thiết kế bộ lọc:

3.1 Xác định đặc tính và đáp ứng tần số của bộ lọc bằng Matlab:

- **Sample *sine.hex:*** Dùng các hàm cơ bản trong Matlab để phân tích phổ từ 1000 samples với tốc độ lấy mẫu $F_s = 44kHz$

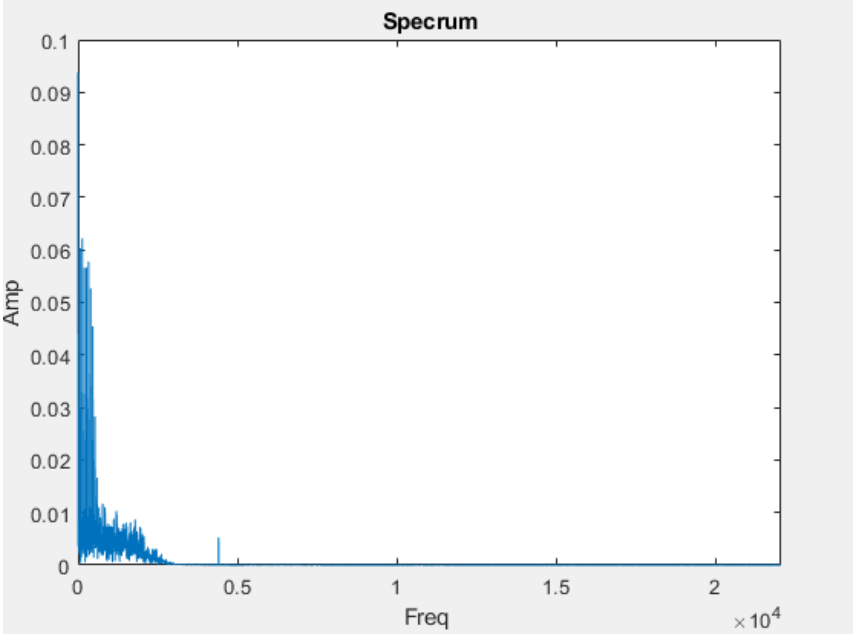


Hình 1: Phổ tần số tín hiệu sample sine.hex

Phân tích phổ cho thấy, có 3 tần số đang trộn lẫn vào tín hiệu, sau khi lọc, cần giữ lại tín hiệu tần số thấp và có biên độ cao nhất, nên ta sẽ thiết kế bộ lọc FIR thông thấp với tần số lấy mẫu $F_s = 44kHz$, tần số cắt $F_c = 1kHz$

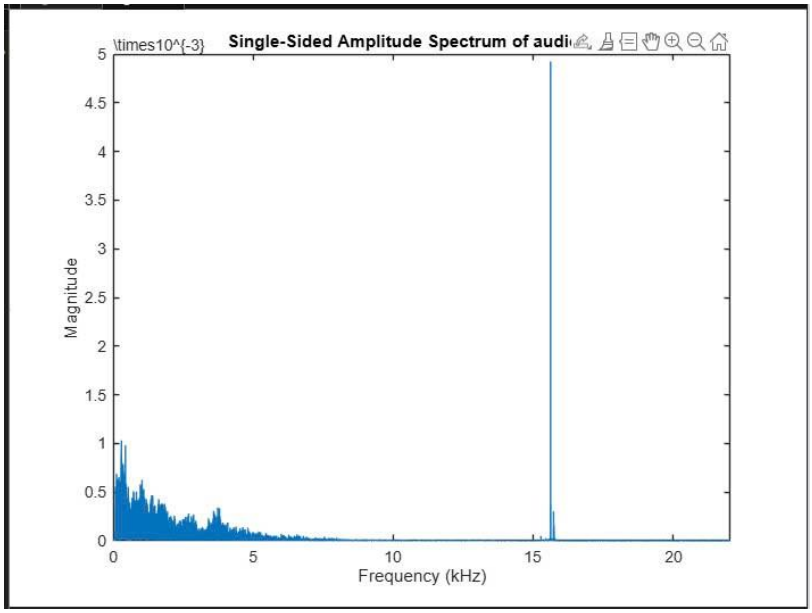
- **Sample *ecg.hex:***

Với tần số lấy mẫu là 40kHz thì phổ tín hiệu khá bé, vì là tín hiệu ECG nên sẽ thiết kế bộ lọc FIR thông thấp với tần số lấy mẫu $F_s = 44kHz$, tần số cắt $F_c = 500Hz$



Hình : Phổ tần số tín hiệu ecg.hex

- **Sample *audio.hex:***



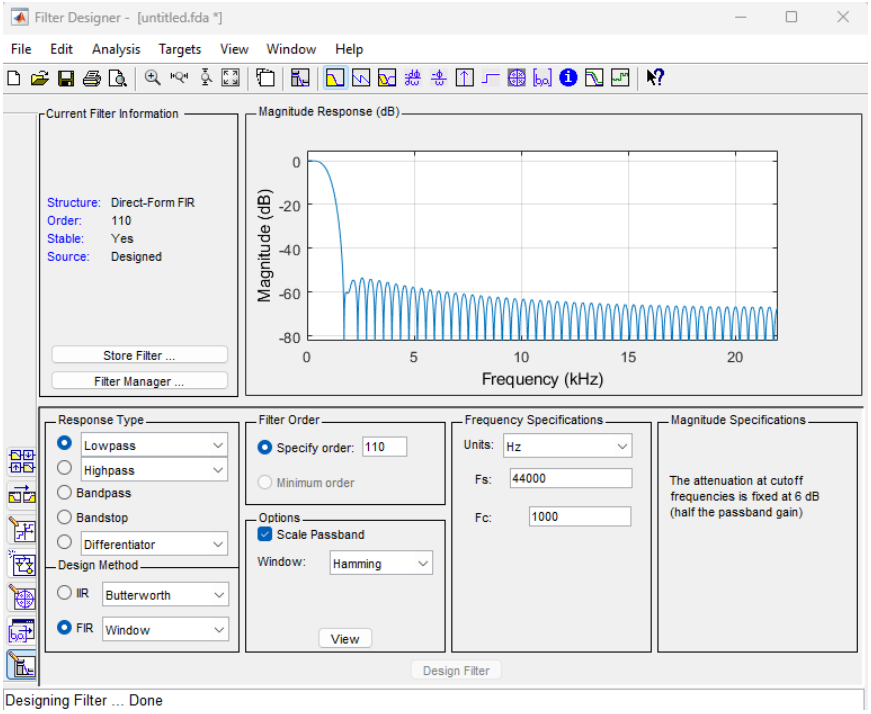
Hình : Phổ tần số tín hiệu audio.hex

Phổ tần số cho thấy âm thanh từ giọng nói nằm trong khoảng dưới 5kHz, tín hiệu nhiễu cần lọc nằm khoảng tần số trên 5kHz, ta sẽ thiết kế bộ lọc FIR thông thấp với $F_s = 44.1kHz$, tần số cắt $F_c = 1kHz$.

3.2 Thiết kế bộ lọc và hệ số bằng Matlab:

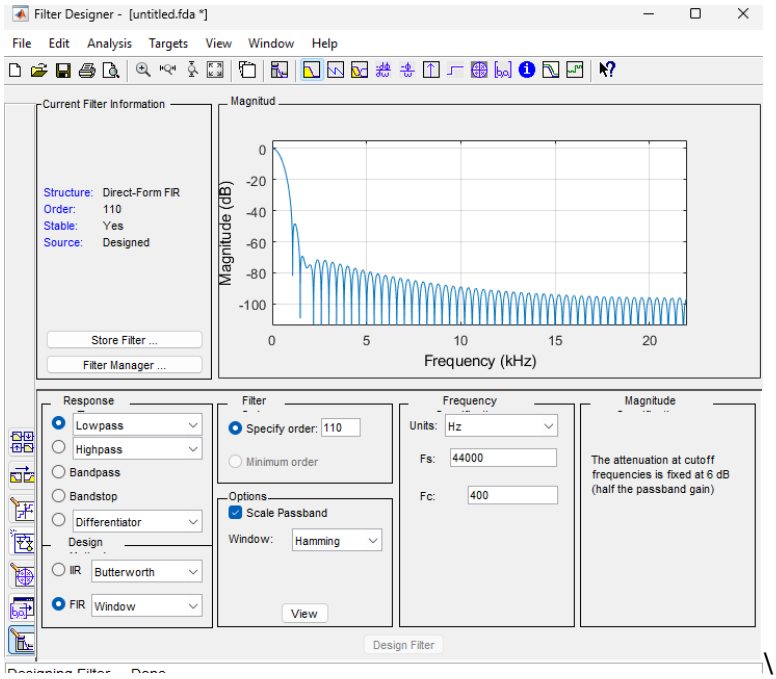
Sử dụng *filterDesigner* để tạo bộ lọc FIR thông thấp 110 order theo thiết kế đã phân tích và export các hệ số từ bộ lọc qua hexadecimal bằng các hàm trong Matlab.

- **Sample sine.hex:**



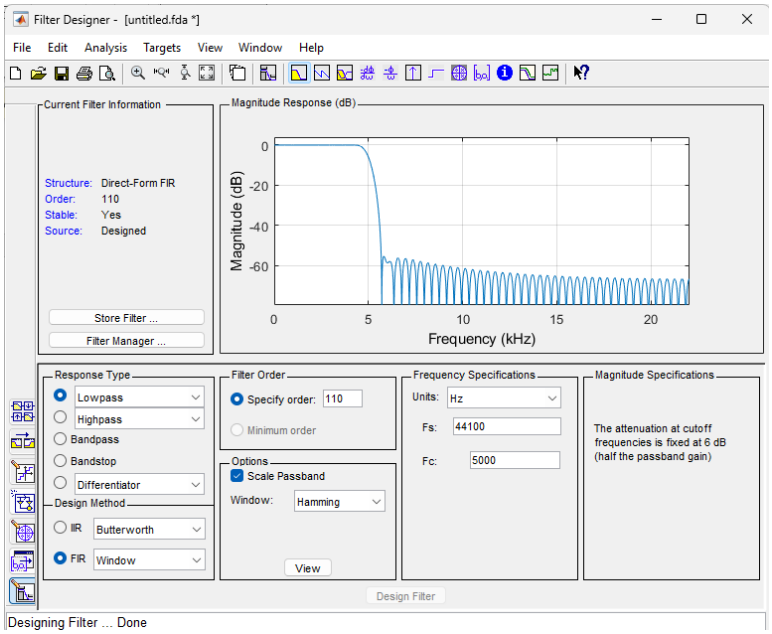
Hình : Filter Designer Sine on Matlab

- **Sample ecg.hex:**



Hình : Filter Designer ECG on Matlab

- **Sample audio.hex:**

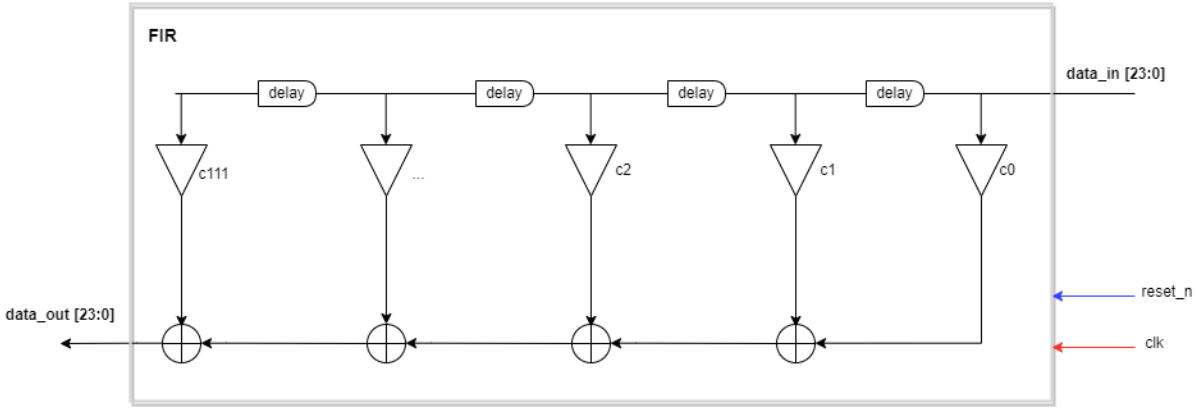


Hình : Filter Designer Audio on Matlab

4. Giải pháp thiết kế:

4.1 Direct Form FIR

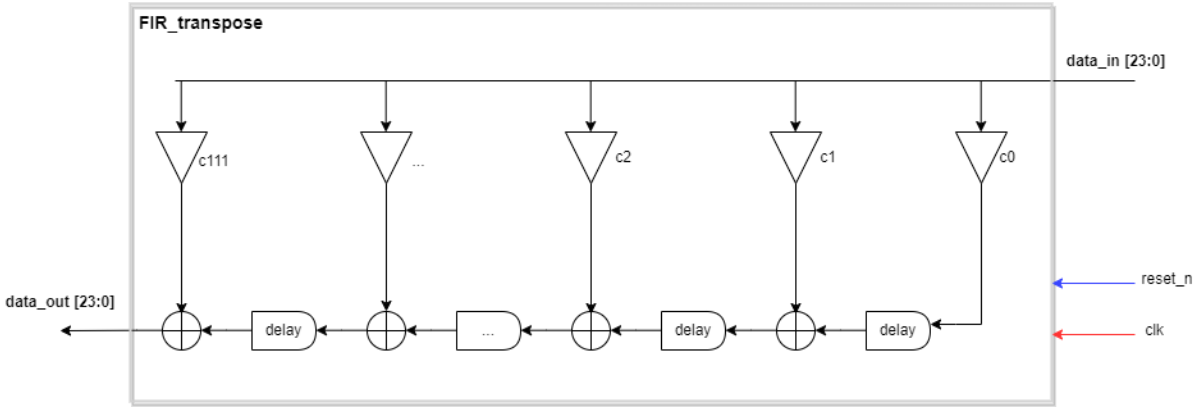
Bộ lọc Dicrect form FIR được tạo bằng cách đưa ngõ vào qua các phần tử tạo trễ (delay) và dùng các phép nhân cho các phần tử, kết quả sẽ là phép cộng tất cả sau khi nhân.



Hình : Dicrect Form FIR Block Diagram

4.2 Transposed FIR

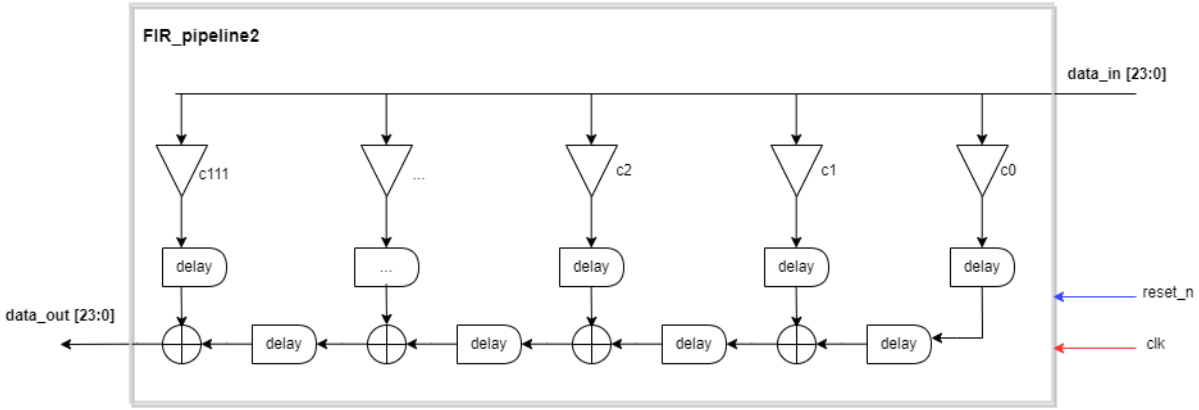
Transposed FIR là một cấu trúc thay thế cho bộ lọc FIR tiêu chuẩn, trong đó cách sắp xếp các khối tính toán của bộ lọc được thay đổi so với cấu trúc trực tiếp (direct form) của FIR.



Hình : Transposed FIR Block Diagram

4.3 Pipelined Transposed FIR

Pipelined Transposed FIR là một biến thể của cấu trúc Transposed FIR, trong đó các khối tính toán (phép nhân và phép cộng) được tổ chức theo một kiến trúc pipelining (đường ống). Điều này giúp tăng tốc độ xử lý tín hiệu bằng cách cho phép các phép tính được thực hiện song song qua nhiều giai đoạn.

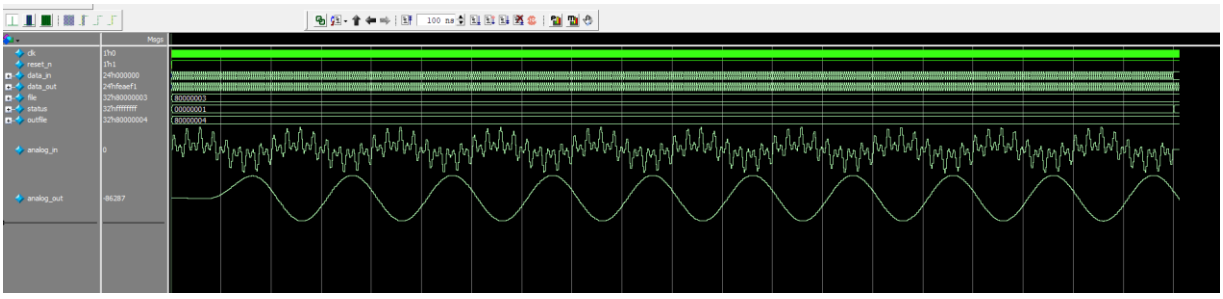


Hình : Pipelined Transposed FIR Block Diagram 2-level

5. Mô phỏng thiết kế

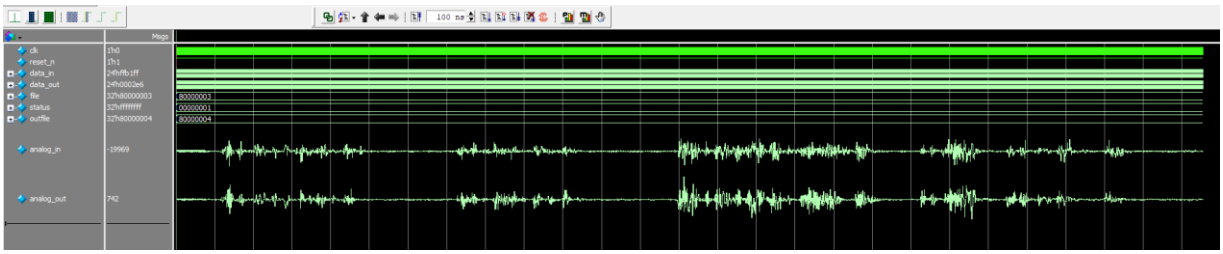
Cả 3 cấu trúc thiết kế sẽ cho kết quả mô phỏng dạng sóng tương đương nhau

5.1 Mẫu tín hiệu Sine



Hình : Dạng sáng mô phỏng tín hiệu sine

5.2 Mẫu tín hiệu Audio

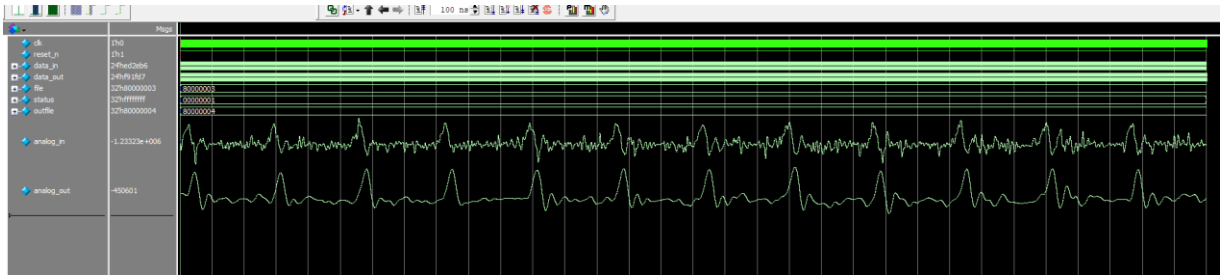


Hình : Dạng sáng mô phỏng tín hiệu audio

Âm thanh trước khi lọc: This link

Âm thanh sau khi qua bộ lọc: This link

5.3 Mẫu tín hiệu ECG



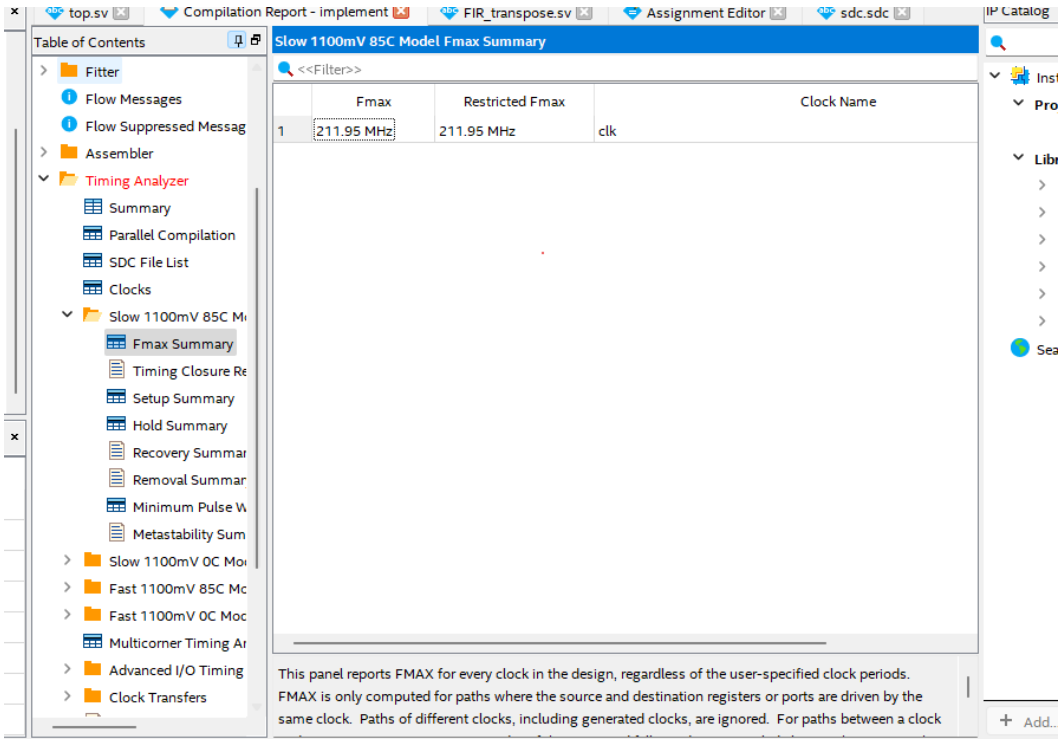
Hình : Dạng sáng mô phỏng tín hiệu ecg

→ Bộ lọc FIR cơ bản đã thực hiện đúng chức năng đề ra.

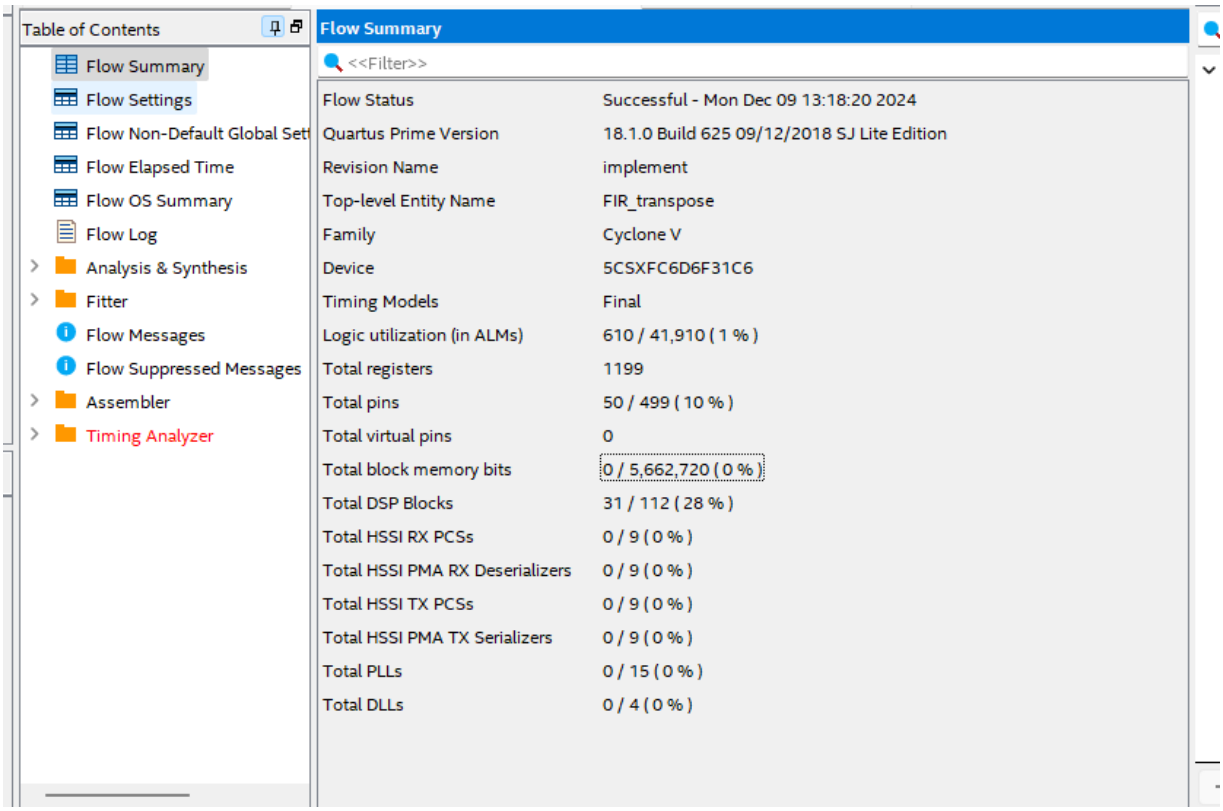
6. Synthesis on FPGA DE10-Standard in Quartus Prime.

Vì bộ lọc được thiết kế 31 tap nên sẽ tiêu hao 31 khối DSP block. Constraint *clk* đầu vào là 50MHz, quan sát F_{max} và tài nguyên đã sử dụng trên FPGA.

6.1 Transposed FIR

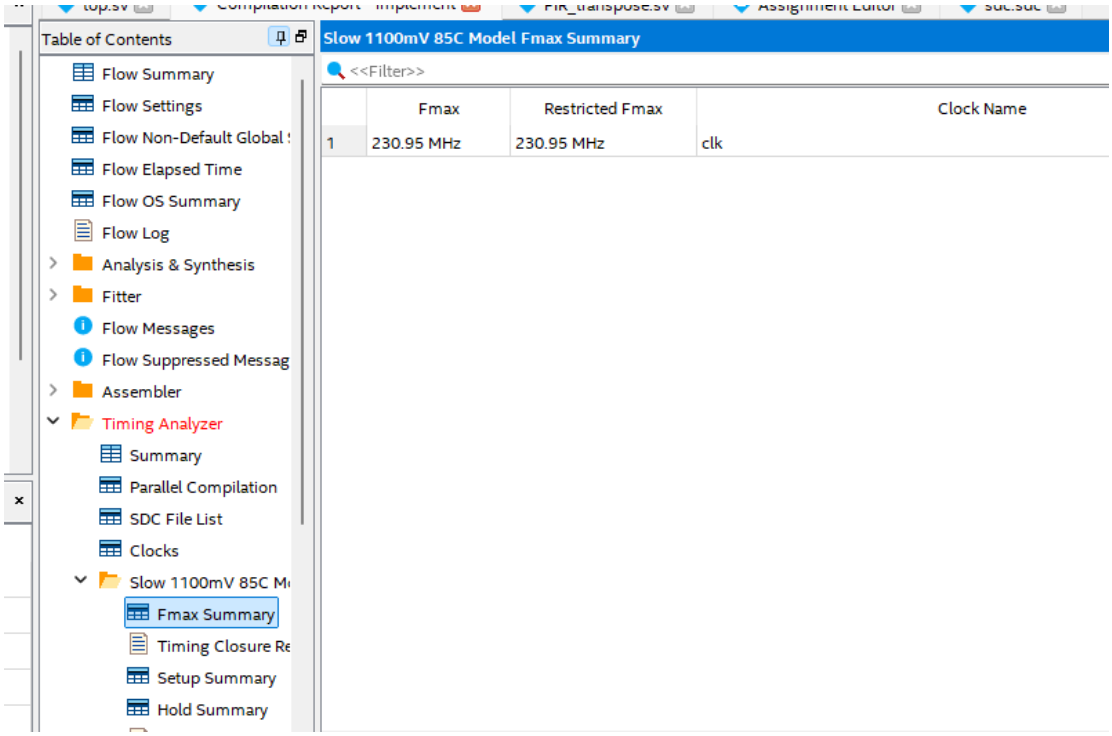


Hình : Fmax Transposed FIR

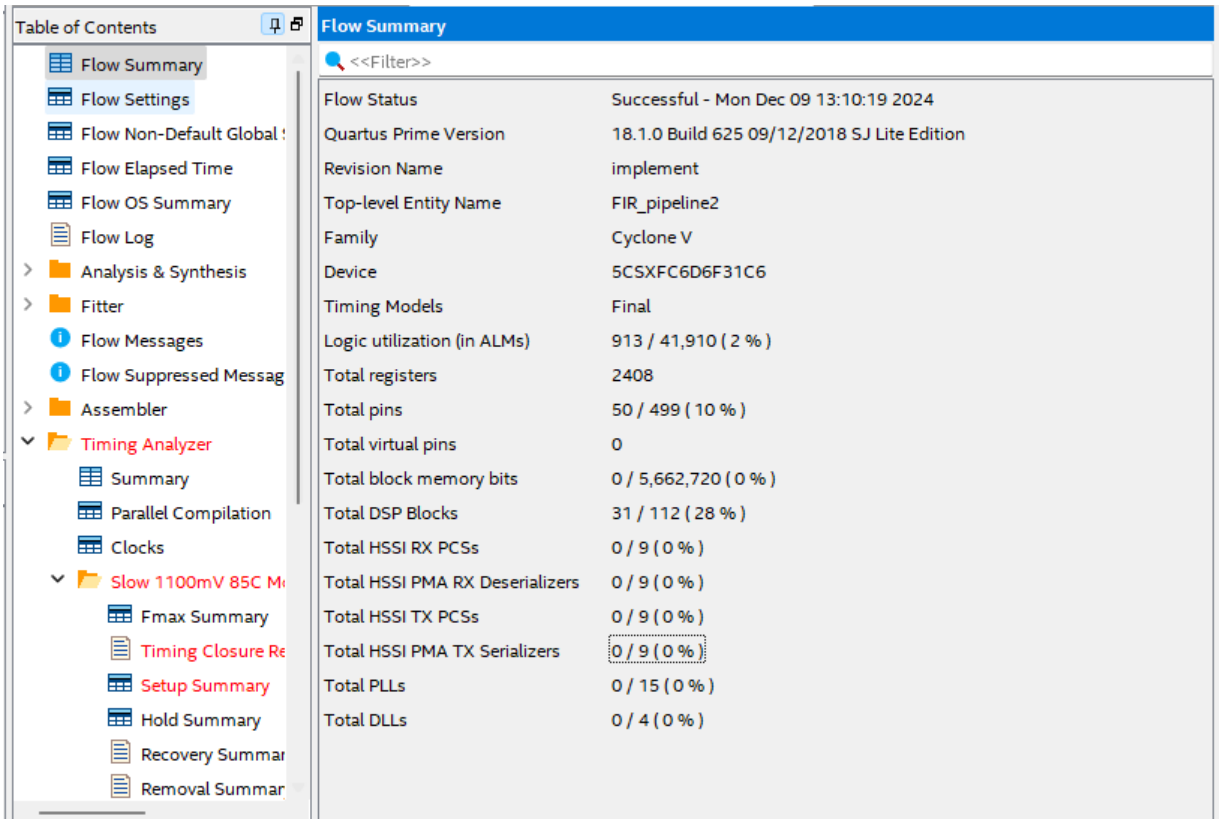


Hình : Tài nguyên sử dụng FIR transpose

6.2 Pipelined Transposed FIR 2-level



Hình : Fmax Pipelined Transposed FIR 2-level



Hình : Tài nguyên sử dụng FIR pipeline 2-level

7. Implement on FPGA

7.1 Mục tiêu và cách triển khai

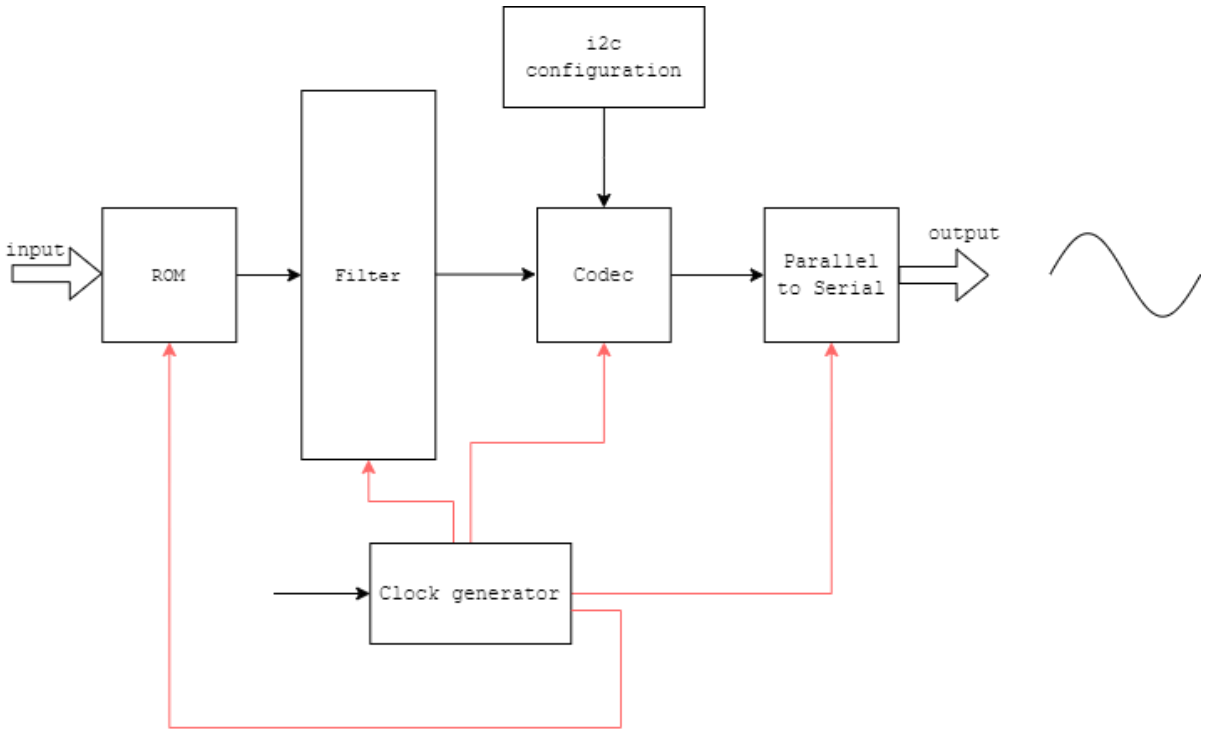
Quá trình thiết kế và triển khai một bộ lọc FIR/IIR trên nền tảng FPGA DE10, từ cấu trúc phần cứng cho đến các bước kiểm tra tín hiệu đầu ra. Trong đó, các khối chức năng chính của hệ thống sẽ bao gồm:

- **Khối ROM lưu trữ mẫu đầu vào:** ROM này chứa các mẫu đầu vào cố định cho bộ lọc FIR/IIR, giúp phân tích đáp ứng của bộ lọc.
- **Khối PLL (Phase-Locked Loop):** được dung để tạo tín hiệu clock chuẩn với tần số phù hợp cho codec âm thanh WM8731.

- **Khối I2C giao tiếp với Codec WM8731:** Codec WM8731 cần được cấu hình qua giao thức I2C để hoạt động theo mong muốn.
- **Khối khởi tạo Codec WM8731:** Thiết lập các tham số âm thanh và điều khiển giao tiếp với bộ chuyển đổi tín hiệu số sang tín hiệu tương tự (DAC).
- **Khối chuyển đổi tín hiệu song sang nối tiếp (P2S):** Nhận dữ liệu song song từ Codec và chuyển qua tín hiệu nối tiếp.
- **Khối bộ lọc FIR/IIR:** đưa tín hiệu dạng song song qua bộ lọc để kiểm chứng đáp ứng tần số.

7.2 Quy trình thiết kế

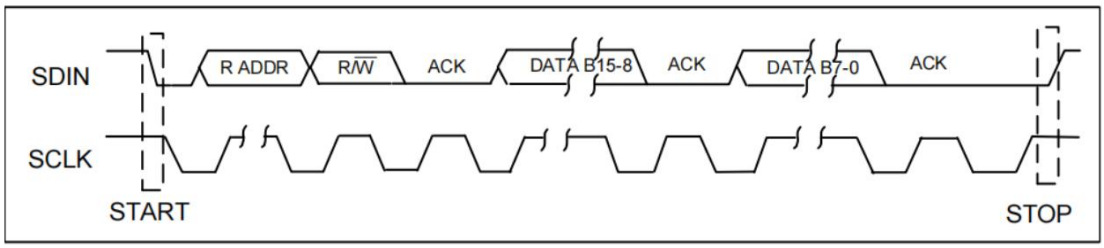
Các khối được thiết kế và phân tích để hiểu cách vận hành của toàn bộ hệ thống



Hình : Overall System

7.2.1 Khởi tạo Codec

Khởi tạo Codec là bước quan trọng để cấu hình Codec hoạt động theo đúng các đặc điểm nhất định được đề ra. Để khởi tạo Codec, giao thức I2C được sử dụng. Trong thực tế, nguyên lý của giao thức này là xác định điều kiện bắt đầu và kết thúc của tín hiệu SDIN và sau đó điều khiển dữ liệu truyền đi chứa các cấu hình. Chẳng hạn, sau khi gửi mỗi tám bit, Codec phải nhận được một bit xác nhận (ACK) từ FPGA để xác nhận rằng dữ liệu đã được nhận chính xác, như minh họa trong Hình dưới.



Hình : Giao thức I2C

Các bit address và data được truyền tới Codec WM8731 để khởi tạo hoạt động theo mong muốn. Để tìm hiểu rõ về các bit cách chọn bit, tham khảo datasheet của CODEC WM8731.

7.2.2 Khối chuyển đổi song song sang nối tiếp (P2S)

Khối bộ chuyển đổi P2S chuyển đổi dữ liệu số song song từ Codec sau khi qua bộ lọc và chuyển nó thành tín hiệu nối tiếp để truyền nó đến output (máy phát song), nó đóng vai trò như một bộ DAC. Sau đó, nó nhận dữ liệu đã được xử lý từ khối FIR sau khi chuyển đổi từ dạng song song sang dạng nối tiếp, cách thức chuyển đổi dựa trên datasheet WM8731 được thể hiện ở hình dưới.

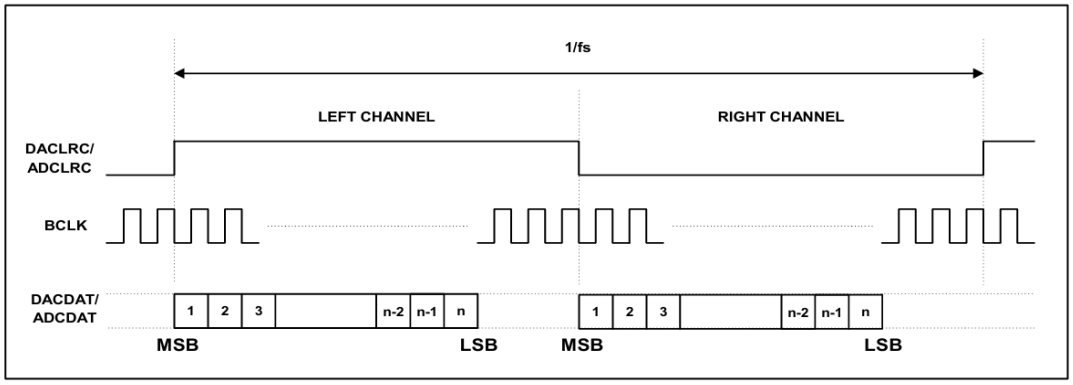


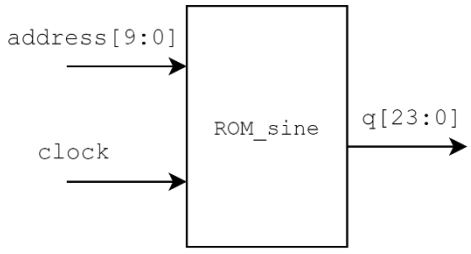
Figure 26 Left Justified Mode

Hình : Giao diện chuyển đổi tín hiệu số

Bclk và *xck* có cùng giá trị (12 MHz trong chế độ USB). Dữ liệu chỉ được gửi trong chu kỳ tiếp theo khi *bclk* ở mức thấp và *lrclk* ở mức cao. Một điều quan trọng khác là tần số lấy mẫu, chúng tôi đặt là 48 kHz, vì vậy, trong 48.000 chu kỳ, chúng tôi chỉ gửi một mẫu (24 bit cho một kênh). Trong 48.000 chu kỳ, *lrclk* chỉ ở mức cao một lần, điều này có nghĩa là chúng tôi kích hoạt *lrclk* ở mức cao trong 1 chu kỳ của *bclk* và sau 48 kHz.

7.2.3 Khối ROM

ROM được tạo bằng IP trong Quartus Prime, dùng để lưu trữ dữ liệu mẫu đầu vào 16 bit address.



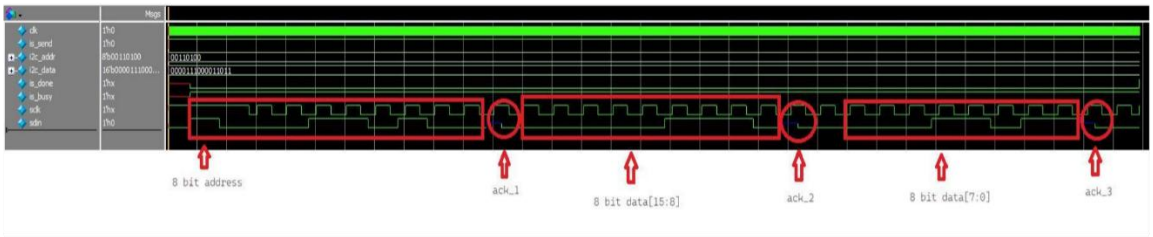
Hình : ROM memory

7.2.4 Khối Clock Generator

Chia 2 bộ tần số cần sử dụng cho *i2c_protocol* và clock *xck* để cấu hình audio codec. Bộ chia tần 400kHz và 12MHz.

7.3 Kiểm tra kết quả

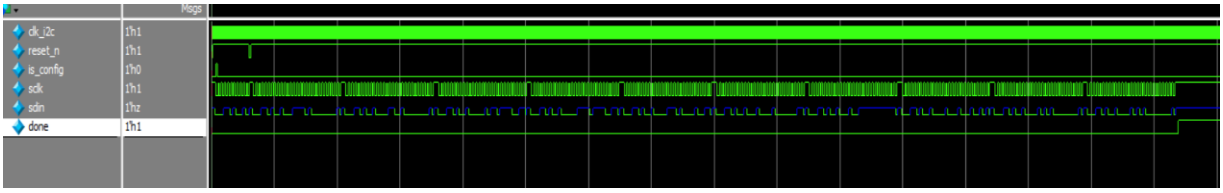
7.3.1 Khởi tạo Codec qua giao thức i2c



Hình : Kết quả test giao thức i2c

Từ hình trên, với 7 bit địa chỉ và 9 bit dữ liệu được truyền tới Codec qua giao thức I2C, cứ mỗi 8 bit tín hiệu *ack/nack* sẽ được *slave* (WM8731) trả về *master* (FPGA).

7.3.2 Kết quả khối Audio codec



Hình : Kết quả test config audio

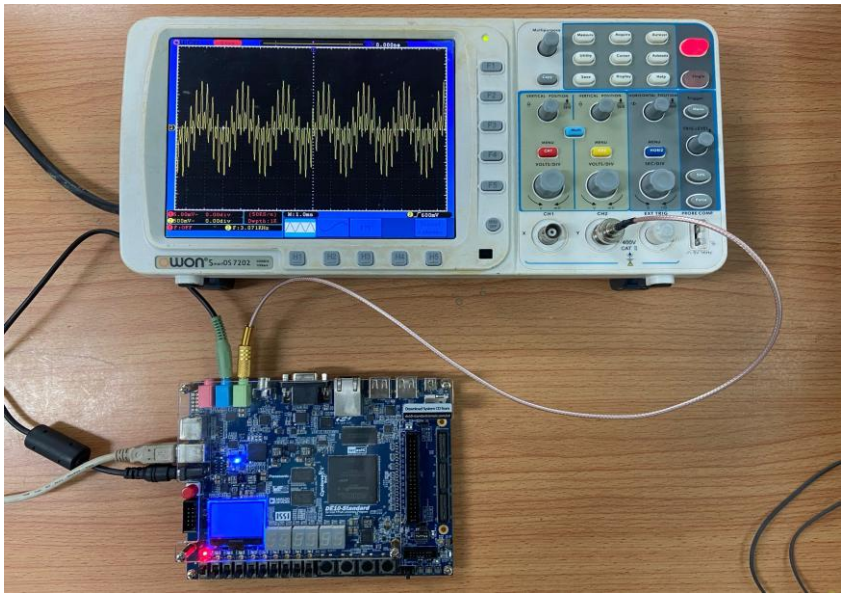
Kết quả cho thấy khi có xung *dalrc* thì tín hiệu song song sẽ được chuyển qua tín hiệu nối tiếp, bắt đầu từ MSB.

7.3.2 Kết quả thực hiện trên FPGA

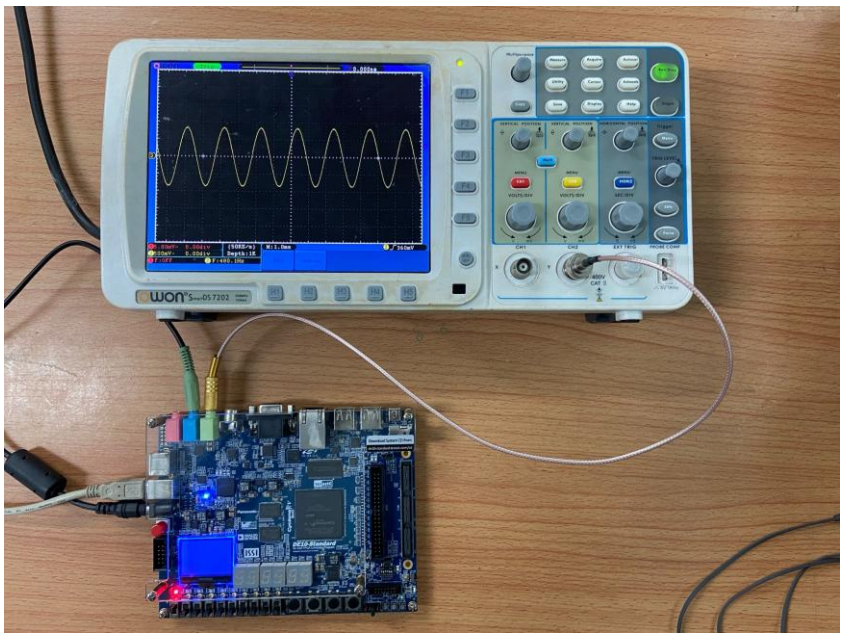
Sử dụng KEY[1] trên FPGA dùng để khởi tạo cho Codec.

SW[0] dung để chuyển đổi dữ liệu data gốc và data sau khi lọc.

Quan sát kết quả tín hiệu trên máy phát sóng. Ở đây chúng mình test tín hiệu sóng sine từ sample *sine.hex*

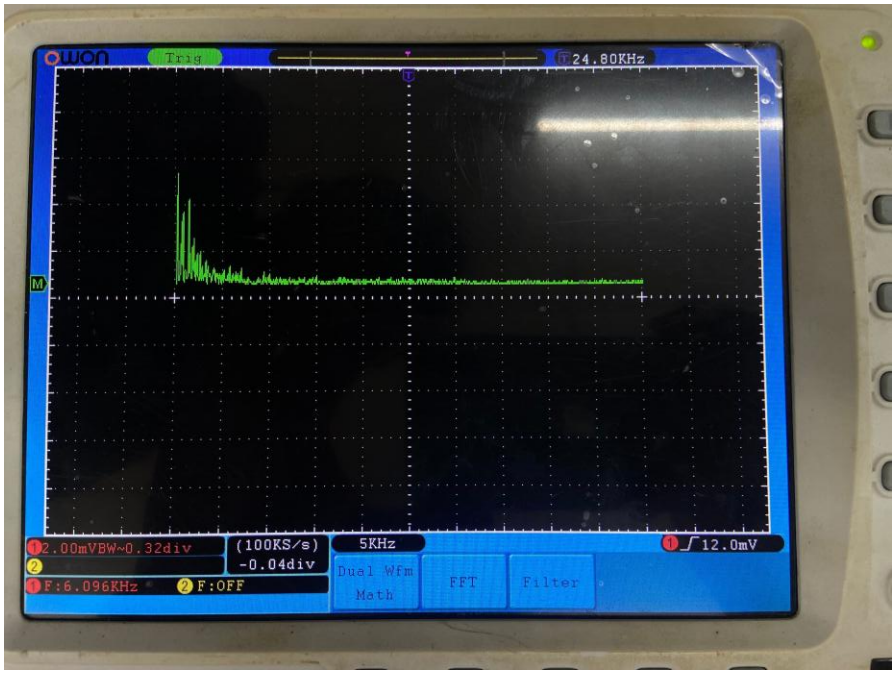


Hình : Kết quả mẫu dữ liệu đầu vào

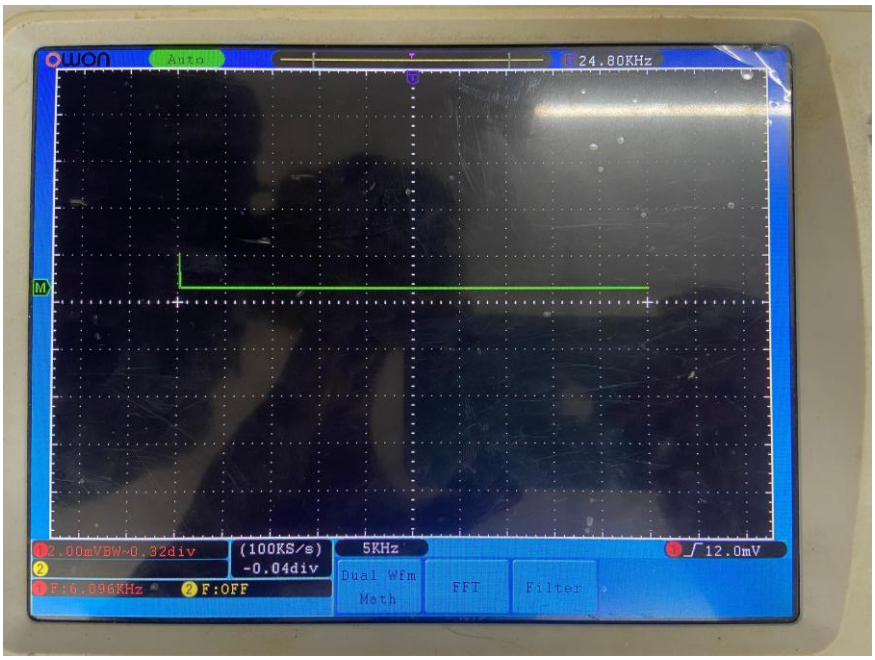


Hình : Kết quả sóng sau khi lọc

Quan sát tín hiệu phổ (FFT) trên máy phát sóng



Hình : Phân tích FFT tín hiệu mẫu đầu vào



Hình : Phân tích FFT tín hiệu sau lọc