Hiện thực và so sánh các thiết kế FFT 2048 điểm xây dựng trên nền tảng FPGA

Đàm Quang Linh*, Hoàng Trọng Thức† và Bùi Trọng Tú‡
Phòng Thí Nghiệm DESLab, Khoa Điện Tử - Viễn Thông
Trường Đại Học Khoa Học Tự Nhiên
Thành phố Hồ Chí Minh, Việt Nam
Email: †‡{htthuc, bttu}@fetel.hcmus.edu.vn

Đinh Đức Anh Vũ
Trường Đại Học Công Nghệ Thông Tin
Thành phố Hồ Chí Minh, Việt Nam
Email: anhvu@uit.edu.vn

***Tóm tắt*—FFT là một thuật toán đóng vai trò quan trọng trong lĩnh vực xử lý tín hiệu số. Ứng dụng của FFT trở nên rất rộng rãi và thiết yếu cho rất nhiều hệ thống xử lý tiếng nói, cũng như truyền thông vệ tinh và các hệ thống viễn thông. Do yêu cầu gắt gao khi thiết kế hệ thống tính FFT, việc chọn lựa mô hình thiết kế là một yếu tố rất quan trọng. Tùy theo cách chọn mô hình thiết kế cũng như cơ số Radix sẽ ảnh hưởng đến những hiệu suất khác nhau của hệ thống. Do đó, để có được sự so sánh cho việc lựa chọn giữa các mô hình, trong bài báo này các tác giả sẽ thực hiện và so sánh chúng với cùng một điều kiện thiết kế là 2048-điểm FFT và xây dựng chung trên một nền tảng FPGA. Có tất cả bốn mô hình FFT khác nhau được xây dựng và so sánh bao gồm: Radix-2 SDF, Radix-2 MDC, Mix-Radix SDF và Mix-Radix MDC. Chúng sẽ được đánh giá kỹ lưỡng về mặt tiêu tốn tài nguyên, tốc độ thực thi và độ chính xác.**

Từ khóa*—*FFT, FPGA, Hardware, Multi-path Delay Commutator, Single-path Delay Feedback, So Sánh, Radix-2, Mix-Radix

I. GIỚI THIỆU

Trong lĩnh vực xử lý tín hiệu số và phân tích tín hiệu, DFT đóng một vai trò không thể thiếu khi nó là là một trong những công cụ phân tích phổ mạnh mẽ nhất. Tuy nhiên, việc tính toán trực tiếp từ DFT mang đến những bất lợi khi khối lượng phép tính lớn dẫn đến độ phức tạp cao. Từ đó, FFT đã được đề xuất bởi Cooley và Turkey [1], một phương pháp cho phép tính toán nhanh DFT bằng cách giảm thiểu số phép tính nhưng vẫn đảm bảo tính đúng đắn của thuật toán. Từ khi được giới thiệu cho đến nay, FFT đã có rất nhiều thay đổi trong phương pháp thực thi, nhiều mô hình cài đặt đã được đề xuất nhắm đến tối ưu tài nguyên, công suất, tốc độ, cũng như độ chính xác.

Thuật toán FFT đã được xây dựng trên phần mềm như trong [2], [3]. Tuy nhiên, do bất lợi về mặt tốc độ, phần mềm chỉ thực thi được các kiến trúc ít điểm FFT (8, 16 điểm) và với cơ số căn bản Radix-2. Vì vậy, giải pháp phần mềm sẽ không thỏa được các hệ thống cần tính tốc độ cao và tính nhiều điểm FFT như DVB-T/H và DVB-T2 (Digital Video Broadcasting) [4], DAB (Digital Audio Broadcasting) [5], MB-OFDM Ultra Wide Band [6], vân vân. Do đó, giải pháp phần cứng là thiết yếu trong thiết kế hệ thống FFT tốc độ cao.

Căn cứ theo bài khảo sát của Chauhan và Karwal [7], các phương pháp và kỹ thuật để xây dựng hệ thống FFT hướng đến thiết kế VLSI đã được trình bày và phân tích. Các kiến trúc phần cứng FFT được chia làm hai loại chính là kiến trúc song song [6] và kiến trúc pipelined [8]. Kiến trúc song song cho thấy ưu thế vượt trội trong điều kiện dữ liệu ngõ vào được truyền đến đồng thời. Khi đó, dữ liệu vào được chia tách ra nhiều lần và xử lý song song giữa các luồng tín hiệu. Thiết kế song song cho tốc độ cao với cái giá tài nguyên tương ứng. Ngược lại, thiết kế pipelined phù hợp với kiểu dữ liệu ngõ vào được đưa đến một cách tuần tự. Thiết kế pipelined cho phép tầng sau có thể bắt đầu thực hiện khi tầng trước chưa hoàn thành xong công việc. Như vậy, tốc độ của hệ thống sẽ được tăng lên đáng kể. Trong các thiết kế pipelined FFT, có hai kỹ thuật chính được đề xuất đó là Single-path Delay Feedback (SDF) [9] và Multi-path Delay Commutator (MDC) [10]. SDF được xây dựng trên ý tưởng dùng một buffer để lưu một nửa dữ liệu ngõ ra được hồi tiếp ngược về, trong khi nửa dữ liệu còn lại được đưa đến tầng sau. Khi nửa sau được truyền đi hết sang tầng sau, phần được lưu trong buffer lúc này mới được đẩy ra bên ngoài. Kiến trúc MDC cũng tương tự, nhưng dùng đến hai buffer cho mỗi tầng và dữ liệu ngõ ra không hồi tiếp. MDC cho tốc độ cao hơn, bù lại sẽ tốn tài nguyên hơn.

Ngoài ra, các kỹ thuật để thực hiện việc nhân số phức cũng là một yếu tố cần phải được cân nhắc khi xây dựng hệ thống. Có hai kỹ thuật chính là: Memory-based [4] và CORDIC-based [11]. Với Memory-based, kết quả phép tính được lưu trữ sẵn trong ROM theo cấu trúc bảng tra, còn với CORDIC-based là sử dụng thuật toán Co-

Ordinate Rotation Digital Computer (CORDIC) để tính toán trực tiếp từ dữ liệu ngõ vào. Vì Memory-based là bảng tra được lưu sẵn nên độ chính xác sẽ cao hơn so với phương pháp CORDIC-based. Tuy nhiên, việc sử dụng memory trong thiết kế phần cứng sẽ gặp một số bất lợi nhất định, như tốc độ thực thi chậm cũng như khó thực hiện ASIC. Ngược lại, phương pháp CORDIC-based sử dụng cổng logic nên sẽ cho lợi thế về tốc độ tính toán cũng như tiết kiệm tài nguyên bộ nhớ.

Ngoài chọn mô hình thiết kế, việc chọn cơ số Radix và số điểm FFT cũng là một yếu tố quan trọng khi xây dựng hệ thống. Radix-2 và Radix-4 là hai cơ số thông dụng nhất, nó cho người thiết kế dễ dàng xây dựng hệ thống nhưng lại có nhiều số phép tính. Các phương pháp trộn Split-Radix [12] như Radix-2/4, Radix-2/8, hoặc cao hơn, sẽ giảm số phép tính nhưng bù lại sẽ làm phức tạp cho hệ thống điều khiển dữ liệu. Ngược lại với việc chọn cơ số thường phụ thuộc vào người thiết kế, thì việc chọn số điểm FFT phụ thuộc vào yêu cầu của hệ thống. Ví dụ như trong các hệ thống xử lý tiếng nói thì lại không cần tính nhiều điểm FFT [13]. Nhưng trong các hệ thống truyền thông như 3GPP-LTE [14] yêu cầu tính FFT trong khoảng 128 đến 2048 điểm, còn DVB-T/H [15] thì cần đến 2048/4096/8192 điểm tính.

Như vậy, khi nhà thiết kế cần xây dựng một hệ thống phần cứng FFT phải cân nhắc giữa nhiều lựa chọn cho mô hình thiết kế. Như MDC hay SDF sẽ cho sự cân bằng giữa tốc độ và tài nguyên. Hay Radix-2 và Mix-Radix sẽ cho sự trao đổi giữa độ chính xác và tài nguyên hệ thống. Vì những lý do đó, trong bài báo này các tác giả sẽ hiện thực và so sánh bốn mô hình khác nhau của kiến trúc FFT. Để có được sự so sánh công bình giữa các mô hình, chúng sẽ được hiện thực trong cùng một điều kiện yêu cầu là 2048 điểm FFT, sử dụng kỹ thuật CORDIC cho phép nhân số phức và cùng xây dựng trên một nền tảng FPGA. Bốn mô hình được hiện thực và so sánh bao gồm: Radix-2 SDF (R2SDF), Radix-2 MDC (R2MDC), Mix-Radix SDF (RMixSDF) và Mix-Radix MDC (RMixMDC). Các mô hình sẽ được đánh giá về tài nguyên, tốc độ và độ chính xác.

Phần còn lại của bài báo được chia như sau. Phần 2 sẽ nêu lý thuyết FFT và nền tảng kiến trúc cơ bản. Phần 3 sẽ trình bày chi tiết các mô hình thiết kế được xây dựng. Phần 4 cho biết kết quả thu được. Cuối cùng, phần 5 cho kết luận toàn văn.

II. THUẬT TOÁN FFT VÀ CÁC NỀN TẢNG KIẾN TRÚC

A. Thuật toán FFT

Thuận toán DFT tính cho N-điểm được định nghĩa như trong công thức (1).

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \quad k = 0, 1, ..., N-1 \tag{1}$$

Trong đó, $x(n)$ là dữ liệu ngõ vào ở miền liên tục thời gian, $X(k)$ là giá trị ngõ ra ở miền tần số, W_N^{kn} là trọng số nhân. W_N^{kn} được tính theo công thức (2).

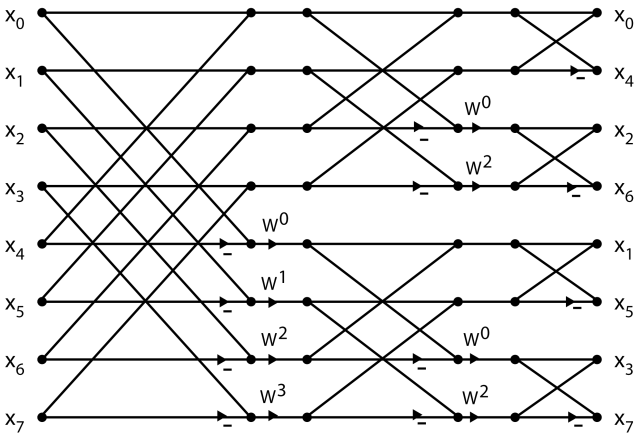
$$W_N^{nk} = exp(-j2\pi kn/N) \tag{2}$$

Việc thực hiện DFT trực tiếp sẽ có độ phức tạp là $O(N^2)$. Thay vào đó, thuật toán FFT giảm độ phức tạp xuống còn $O(N.\log_2 N)$

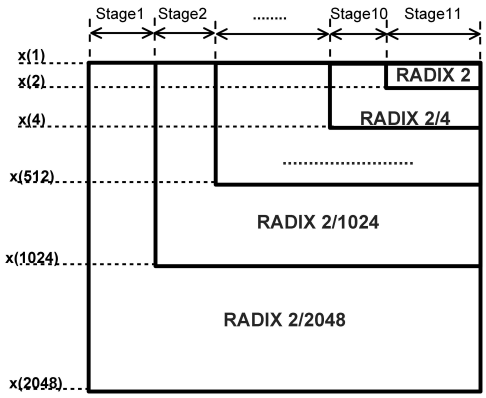
1) Radix-2 FFT: Radix-2 là cơ số căn bản nhất trong thiết kế FFT. Thiết kế Radix-2 có phần tiết kiệm tài nguyên do chính sự đơn giản của nó. Nhưng bù lại sẽ không có độ chính xác và tốc độ bằng các thiết kế trộn Radix. Ý tưởng thiết kế Radix-2 FFT dựa trên việc chia chuỗi N-điểm $X(k)$ ra thành hai chuỗi $\frac{N}{2}$-điểm chẵn, $X(2r)$, và $\frac{N}{2}$-điểm lẻ, $X(2r+1)$, được tính độc lập. Như vậy với phương pháp tiếp cận Radix-2, từ công thức (1) sẽ được chuyển thành công thức (3). Hình 1 cho thấy ví dụ tính 8-điểm FFT sử dụng cơ số Radix-2.

$$\begin{aligned} X(2k) &= \sum_{n=0}^{N/2-1} [x(n) + x(n+\tfrac{N}{2})] W_N^{kn} \\ X(2k+1) &= \sum_{n=0}^{N/2-1} \{[x(n) - x(n+\tfrac{N}{2})] W_N^n\} W_{N/2}^{kn} \\ & (k = 0, 1, ..., N/2-1) \end{aligned} \tag{3}$$

2) Mix-Radix FFT: Như đã trình bày ở trên, kỹ thuật trộn cơ số có thể cho phép mạch thiết kế đạt kết quả tốt hơn, có phần tiết kiệm và giảm số phép tính cũng như độ phức tạp của thuật toán. Tuy nhiên, mạch Mix-Radix sẽ làm phức tạp cơ chế điều khiển của hệ thống. Mix-Radix dựa trên việc chia đôi số điểm tính FFT ra theo từng tầng, mỗi lần chia đôi sẽ áp dụng một cơ số radix khác nhau.



Hình 1. Ví dụ cho mô hình 8-điểm FFT Radix-2.



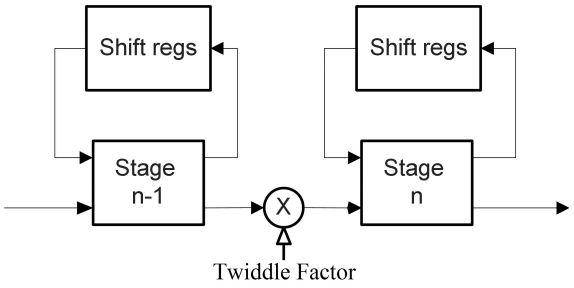
Hình 2. Ví dụ xây dựng 2048-điểm FFT Mix-Radix.

Hình 2 cho thấy ví dụ thiết kế Mix-Radix cho 2048-điểm FFT. Radix-2/2048 sẽ chia N điểm ban đầu theo chẵn và lẻ, sau đó $N/2$ điểm chẵn sẽ được tính theo Radix-2, còn $N/2$ điểm lẻ sẽ được tính theo Radix-2048. Theo hình 2 ta thấy toàn bộ tầng 1 và nửa dưới các tầng tiếp theo được tính theo Radix-2/2048. Điều đó có nghĩa là điểm từ 1 đến 1024 của tầng 1 được tính theo Radix-2, còn điểm từ 1025 đến 2048 của tất cả các tầng được tính theo Radix-2048. Tương tự trong vùng tính của Radix-2/1024 sẽ bao gồm: điểm từ 1 đến 512 của tầng 2 được tính theo Radix-2, còn điểm từ 513 đến 1024 của các tầng từ tầng 2 đến tầng cuối cùng được tính theo Radix-1024. Cứ như vậy cho đến cuối cùng sẽ còn Radix-2 thuần túy áp dụng cho 2 điểm đầu tiên của tầng cuối cùng.

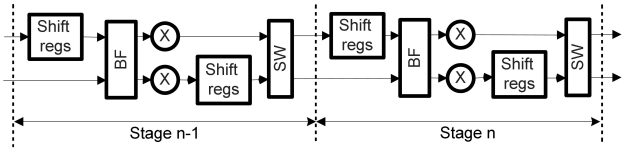
B. Các nền tảng kiến trúc

1) Kiến trúc SDF: Hình 3 thể hiện ý tưởng của kiến trúc SDF. Mỗi tầng SDF sẽ có một bộ đệm dữ liệu hồi tiếp. Dung lượng bộ đệm của tầng sau sẽ bằng nửa dung lượng bộ đệm của tầng trước. Bộ nhân số phức W sẽ nằm trên đường đi dữ liệu giữa hai tầng kế tiếp như có thể thấy trong hình 3. Việc thay đổi cơ số Radix chỉ làm thay đổi cấu trúc và cách điều khiển bộ nhân W, chứ không làm thay đổi cấu trúc của một tầng trong SDF.

2) Kiến trúc MDC: Ý tưởng thiết kế MDC có thể thấy trong hình 4. Tại mỗi tầng sẽ có hai bộ đệm dữ liệu. Trong một tầng, bộ đệm thứ hai có dung lượng bằng nửa dung lượng bộ đệm đầu tiên. Và dung lượng bộ đệm đầu tiên của một tầng sẽ bằng dung lượng bộ đệm thứ hai của tầng ngay trước nó. Kiến trúc R2MDC bao gồm $log_2(N-2)$ phép nhân, $2log_2(N)$ phép công và $1.5N-2$ thanh ghi. Trong thiết kế MDC, bộ nhân số phức W nằm trên đường dữ liệu đi giữa hai bộ đệm trong một tầng, chứ không nằm giữa hai tầng như trong thiết kế SDF. Việc thay đổi cơ số Radix trong thiết kế MDC sẽ làm thay đổi cách điều khiển và kiến trúc bộ nhân W.



Hình 3. Ý tưởng thiết kế SDF.



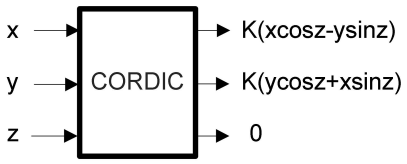
Hình 4. Ý tưởng thiết kế MDC.

III. HIỆN THỰC CÁC KIẾN TRÚC FFT

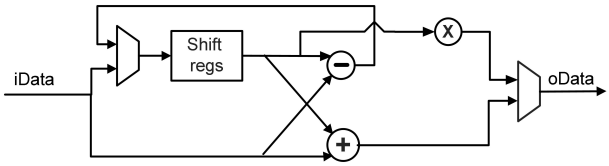
A. Bộ nhân số phức dùng CORDIC-based

CORDIC là một thuật toán mạnh mẽ được sử dụng trong các thiết kế phần cứng dùng để tính toán các chức năng toán học siêu việt. Ứng dụng của CORDIC để tính các phép toán lượng giác, hyberbolic, hàm mũ, hàm logarit và tổ hợp của các hàm trên. Độ chính xác của CORDIC-based sẽ không bằng phương pháp memory-based truyền thống. Tuy nhiên, CORDIC phổ biến vì sử dụng chuỗi các bộ cộng để tính ra kết quả, dẫn đến ít hao tốn tài nguyên và đạt tốc độ cao hơn. Trong các thiết kế FFT được đề xuất, CORDIC được sử dụng ở chế độ Rotation Circular để tính phép nhân số phức. Công thức trình bày trong công thức (4), và mô hình thiết kế của nó được thể hiện trong hình 5.

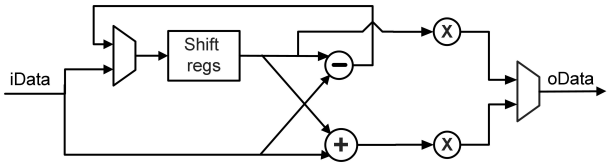
$$\begin{aligned} x_{i+1} &= x_i - d_i y_i z^{-i} \\ y_{i+1} &= y_i + d_i x_i z^{-i} \\ z_{i+1} &= z_i - d_i \alpha_i \\ d_i &= sign(z_i) \\ \alpha &= tan^{-1} z^{-i} \end{aligned} \tag{4}$$



Hình 5. Top-view của khối CORDIC ở chế độ Rotation Circular.



Hình 6. Mô hình kiến trúc R2SDF.



Hình 7. Mô hình kiến trúc RMixSDF.

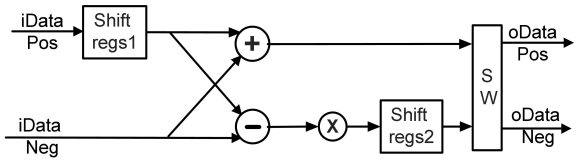
Kết quả của phép tính CORDIC sẽ trội thêm một hệ số K không mong muốn. Hệ số này được khử bằng cách nhân với số nghịch đảo của nó. Tuy nhiên, bởi vì K là hằng số, nên trong các thiết kế sẽ sử dụng thuật toán *Canonical signed digit* (CSD) để giảm tài nguyên và tăng thời gian đáp ứng. Thuật toán CSD có tác dụng chuyển đổi phép nhân hằng số thành chuỗi các phép cộng và dịch.

B. Kiến trúc SDF

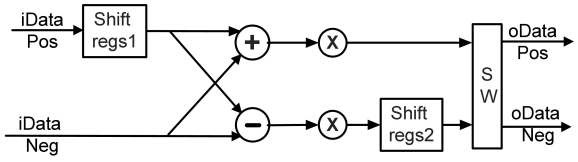
Hình 6 và hình 7 lần lượt thể hiện kiến trúc của một tầng R2SDF và RMixSDF. Ta có thể thấy kiến trúc của chúng là tương đồng nhau với các bộ multiplexers, thanh ghi dịch, butterfly (bộ cộng, trừ) và bộ nhân CORDIC. Một cách tổng quát, quy trình hoạt động một tầng SDF được chia làm ba pha:

- *Pha 1:* $\frac{N}{2}$ dữ liệu đầu được lưu vào trong thanh ghi dịch. Lúc này không có dữ liệu ngõ ra.
- *Pha 2:* $\frac{N}{2}$ dữ liệu sau sẽ thực hiện tính với dữ liệu đệm trong thanh ghi dịch. Kết quả phần (+) sẽ đi thẳng đến tầng tiếp theo, còn kết quả phần (-) sẽ được hồi tiếp lưu trở lại vào thanh ghi.
- *Pha 3:* Sau pha 2 sẽ có $\frac{N}{2}$ kết quả phần (-) đang nằm trong bộ đệm. Lượng dữ liệu này sẽ được chuyển ra bên ngoài để kết thúc quá trình hoạt động.

Khác với Radix-2, cơ số trộn Mix-Radix sẽ có hai bộ nhân CORDIC nằm trên cả hai đường (+) và (-) như có thể thấy trong hình 7. Một cách tổng quát, tốc độ thực hiện 2 mô hình R2SDF và RMixSDF là như nhau do tương đồng về mặt cấu trúc. Tuy nhiên, Mix-Radix sẽ tiêu tốn tài nguyên hơn do trội thêm một bộ nhân, đồng nghĩa với việc cho kết quả chính xác hơn Radix-2.



Hình 8. Mô hình kiến trúc R2MDC.



Hình 9. Mô hình kiến trúc RMixMDC.

C. Kiến trúc MDC

Mô hình thiết kế của R2MDC và RMixMDC lần lượt được trình bày trong hình 8 và hình 9. Trong hình, mỗi tầng MDC sẽ bao gồm hai bộ đệm thanh ghi dịch Shift_regs1 và Shift_regs2, butterfly (bộ cộng, trừ), bộ nhân CORDIC và các multiplexers. Nguyên lí hoạt động chung của một tầng MDC bao gồm bốn pha:

- *Pha 1:* $\frac{N}{2}$ dữ liệu đầu được lưu vào trong Shift_reg1. Lúc này không có dữ liệu ngõ ra.
- *Pha 2:* $\frac{N}{4}$ dữ liệu tiếp theo tính toán với dữ liệu đệm từ Shift_reg1 cho ra kết quả phần (+) sẽ đi thẳng đến tầng tiếp theo, trong khi kết quả phần (-) được lưu vào Shift_reg2. Lúc này chỉ có ngõ $oData\_pos$ cho ra dữ liệu.
- *Pha 3:* $\frac{N}{4}$ dữ liệu còn lại tiếp tục được tính toán với dữ liệu ra từ Shift_reg1. Lúc này kết quả phần (+) đi thẳng đến ngõ ra $oData\_neg$, trong khi kết quả phần (-) sẽ cất vào Shift_reg2 làm cho phần dữ liệu cũ có trong thanh ghi dịch này đẩy ra bên ngoài thông qua ngõ $oData\_pos$. Tức là lúc này cả hai ngõ ra đều chuyển dữ liệu đến tầng sau.
- *Pha 4:* Lúc này trong thanh ghi dịch Shift_reg2 còn chứa $\frac{N}{4}$ dữ liệu của kết quả phần (-) được tính ra trong pha 3. Lượng dữ liệu này được chuyển hết sang tầng sau thông qua ngõ ra $oData\_pos$ để kết thúc quy trình.

Một cách tương tự, khi tính cơ số trộn Mix-Radix sẽ có hai bộ nhân CORDIC nằm trên cả hai đường (+) và trừ (-), thay vì chỉ nằm trên đường (-) như trong thiết kế Radix-2. Và như vậy, RMixMDC và R2MDC sẽ có cùng tốc độ thực thi do tương đồng trong cấu trúc thiết kế, nhưng RMixMDC sẽ cho kết quả chính xác hơn và tiêu tốn nhiều tài nguyên hơn. So với thiết kế SDF, do

MDC không hồi tiếp dữ liệu về nên sẽ cho tốc độ nhanh hơn, nhưng bù lại tiêu tốn nhiều tài nguyên hơn một bộ thanh ghi dịch cho mỗi tầng.

D. Các tối ưu

1) Tối ưu bộ nhân tầng đầu tiên: do dữ liệu ngõ vào là chỉ có phần thực và là số nguyên, nên phần ảo và phần thập phân được gán bằng không. Từ đó, bộ nhân CORDIC ở tầng đầu sẽ được tối ưu dựa vào lợi thế này.

2) Tối giản bộ nhân tầng cuối cùng: do tính chất của cả Radix-2 và Mix-Radix ở tầng cuối cùng không nhân với W. Vì thế, tầng cuối cùng của cả bốn mô hình dữ liệu sẽ đi thẳng mà không qua bộ nhân CORDIC. Kết quả thiết kế tầng cuối cùng ở các mô hình sẽ tiết kiệm được tài nguyên.

3) Tối ưu bộ nhân $-j$: đối với Radix-2, tầng kế cuối chỉ nhân với $W_N^{N/2} = -j$. Tương tự, thiết kế Mix-Radix cũng tồn tại rất nhiều phép nhân $-j$ trong mô hình. Mà khi nhân với $-j$ chính là đổi chỗ và đảo dấu giữa phần thực và phần áo, như có thể thấy trong công thức (5). Do đó, thiết kế đặc biệt bộ nhân $-j$ sẽ chỉ còn lại các multiplexer và bộ đảo dấu, dẫn đến tiết kiệm được đáng kể tài nguyên hệ thống.

$$(a + jb) * -j = b - ja \tag{5}$$

IV. KẾT QUẢ THỰC NGHIỆM

Bốn mô hình thiết kế đều được xây dựng trên board Altera StratixIV với FPGA chip là EP4SGX230KF40C2. Kết quả thực nghiệm được trình bày và so sánh trong bảng I.

A. Đáp ứng thời gian

Các mô hình được kiểm tra dựa trên các thông số về tần số đáp ứng tối đa F_{Max}, đường delay dài nhất trong mạch *Critial Path*, tổng số chu kỳ clock cần thiết để tính ra kết quả (*Latency*) và tổng số thời gian tính theo μs (*Timing*), số triệu bit mỗi giây hệ thống tính toán được (Mega bit per second - *MBps*).

Cả bốn mô hình R2SDF, RMixSDF, R2MDC và RMixMDC đều cho đáp ứng tần số tối đa F_{Max} xấp xỉ như nhau, $\approx$ 42 MHz. Tuy nhiên, thiết kế MDC cho Latency ít hơn khoảng 50% so kiến trúc SDF, dẫn đến đáp ứng thời gian MDC nhanh hơn gần gấp đôi.

Dựa vào bảng I, ta có thể thấy Radix-2 hay Mix-Radix không ảnh hưởng nhiều đến tốc độ tính toán của hệ thống. Tuy nhiên, thời gian thực thi của Radix-2 có phần nhỉnh hơn đôi chút khi so sánh với Mix-Radix có cùng một cấu trúc.

Bảng I. KẾT QUẢ THỰC NGHIỆM BỐN MÔ HÌNH THIẾT KẾ FFT KHÁC NHAU.

		RMixMDC	RMixSDF	R2MDC	R2SDF
ALUTs		24,719	14,387	12,866	13,121
Registers		111,995	67,312	132,462	77,570
F_{Max}	85^0C	42.09	40.19	44.34	41.86
	0^0C	43.57	41.56	45.92	43.44
Critical Path (ns)		23.732	24.832	22.534	23.849
Latency (Clock)		2057	3081	2057	3081
Timing (μs)		48.817	76.507	46.352	73.479
MBps		671.245	428.298	706.932	445.952
MSE		3.780e-2	3.780e-2	4.674e-2	4.674e-2

B. Tài nguyên hệ thống

Do Mix-Radix tiêu tốn đến hai bộ nhân CORDIC so với chỉ một bộ nhân của thiết kế Radix-2, dẫn đến tài nguyên logic (ALUTs) của Mix-Radix nhiều hơn. Về tài nguyên thanh ghi (Registers) thì Mix-Radix cho thấy tiêu tốn ít hơn đôi chút so với mô hình Radix-2 có cùng cấu trúc.

Cấu trúc MDC tiêu tốn đến hai bộ đệm thanh ghi dịch trong một tầng, so với chỉ một bộ đệm của SDF. Điều này dẫn đến số thanh ghi tiêu tốn trong các thiết kế MDC gần gấp đôi so với mô hình SDF, như ta có thể thấy rõ ràng trong bảng I.

C. Độ chính xác

Kết quả của hệ thống phần cứng được so sánh với kết quả lý tưởng lấy từ phần mềm MATLab. Công cụ bình phương tối thiểu (mean square error - *MSE*) được sử dụng để đánh giá sự sai số này. Như có thể thấy trong mục MSE của bảng I, việc chọn cơ số trộn Mix-Radix cho độ chính xác cao hơn khoảng 25% so với phương pháp tiếp cận dùng Radix-2. Như có thể thấy, việc lựa chọn kiến trúc MDC hay SDF không ảnh hưởng đến độ chính xác của hệ thống.

D. Tổng kết so sánh

Lựa chọn giữa kiến trúc SDF và MDC sẽ đánh đổi giữa tốc độ thực thi và tài nguyên thanh ghi, trong khi Mix-Radix và Radix-2 sẽ lựa chọn giữa độ chính xác và tài nguyên logic. Thiết kế MDC cho tốc độ nhanh gần gấp đôi so với SDF, với tiêu tốn khoảng hai lần tổng số thanh ghi. Mix-Radix sẽ cho kết quả chính xác hơn khoảng 25% đổi lại tốn nhiều tài nguyên logic hơn so với mô hình Radix-2.

V. KẾT LUẬN

Bài báo này vừa trình bày việc so sánh giữa các tùy chỉnh khi thiết kế hệ thống phần cứng FFT. Từ đó rút ra kết luận cho các nhà thiết kế phần cứng FFT có thể cân nhắc giữa việc chọn MDC hay SDF, Mix-Radix

hay Radix-2 sao cho phù hợp với từng yêu cầu cụ thể của hệ thống. Kiến trúc MDC sẽ cho thiết kế có tốc độ nhanh hơn gần gấp đôi, nhưng bù lại tiêu tốn gấp đôi tài nguyên thanh ghi. Trong khi đó, thiết kế Mix-Radix cho độ chính xác cao hơn khoảng 25% nhưng sẽ tiêu tốn tài nguyên logic hơn đôi chút.

Trong tương lai gần, các tác giả sẽ tiếp tục thực hiện thêm các mô hình FFT với nhiều tùy chọn khác nhau về số điểm, từ 128-điểm đến 8192-điểm FFT, cũng như thay đổi giữa việc dùng memory-based hay CORDIC-based để thiết kế bộ nhân số phức. Ngoài ra, các mô hình sau khi được thử nghiệm trên FPGA sẽ được tiến hành ASIC để có được thêm kết quả đo về công suất tiêu thụ mạch, cũng như rút ra được kết quả đo chính xác hơn về đáp ứng thời gian và tài nguyên hệ thống tiêu tốn. Như vậy, bài báo sẽ cho cái nhìn toàn diện và xuyên suốt từ ý tưởng đến sản phẩm trong quá trình xây dựng hệ thống phần cứng FFT.

LỜI CẢM ƠN

Bài báo được tài trợ bởi đề tài 39/2013/HĐ-SKHCN của sở khoa học công nghệ thành phố Hồ Chí Minh.

TÀI LIỆU THAM KHẢO

[1] James W. Cooley and John W. Turkey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Mathematics of Computation*, Vol. 19, no. 90, pp. 297-301, April 1965.

[2] Xuan Guan, Hai Lin, and Yunsi Fei, "Design of an application-specific instruction set processor for high-throughput and scalable FFT," *Design, Automation & Test in Europe Conference & Exhibition (DATE '09)*, pp. 1302-1307, April 2009.

[3] T. Cupaiuolo and D. Lo Iacono, "A flexible and fast software implementation of the FFT on the BPE platform," *Design, Automation & Test in Europe Conference & Exhibition (DATE '12)*, pp.1467-1470, Mar. 2012.

[4] Kisun Jung and Hanho Lee, "Low-Complexity Multi-Mode Memory-Based FFT Processor for DVB-T2 Applications," *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E94-A, No. 11, pp. 2376-2383, Nov. 2011.

[5] Jun-Rim Choi, Soo-Bok Park, Dong-Seok Han, and Se-Ho Park, "A 2048 complex point FFT architecture for digital audio broadcasting system," *Proc. of IEEE Int. Symposium on Circuits and Systems (ISCAS 2000)*, Vol.5, pp. 693-696, 2000.

[6] Hanho Lee and Minhyeok Shin, "A high-speed low-complexity two-parallel radix-2^4 FFT/IFFT processor for UWB applications," *Proc. of IEEE Asian Solid-State Circuits Conf. (ASSCC '07)*, pp. 284-287, Nov. 2007.

[7] T. Chauhan and V. Karwal, "DFT implementation aspects and techniques suitable for VLSI implementation: A survey," *Int. Conf. on Signal Processing and Communication (ICSC)*, pp. 330-334, Dec. 2013.

[8] He, Shousheng, and Mats Torkelson, "A New Approach to Pipeline FFT Processor," *Proc. of IEEE Int. Conf. on Parallel Processing Symposium (IPPS '96)*, pp. 766-770, 1996.

[9] N. Polychronakis, D. Reisis, E. Tsilis, "A continuous-flow, Variable-Length FFT SDF architecture," *Proc. of IEEE Int. Conf. on Electronics, Circuits, and Systems (ICECS '10)*, pp. 730-733, Dec. 2010.

[10] K. J. Yang, S. H. Tsai, and G. C. H. Chuang, "MDC FFT/IFFT Processor With Variable Length for MIMO-OFDM Systems," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol.21, No.4, pp. 720-731, April 2013.

[11] L. Dongpei, L. Hengzhu, Z. Botao, Z. Jianfeng, W. Shixian, and L. Zhengfa, "Low Cost CORDIC-Based Configurable FFT/IFFT Processor for OFDM Systems," *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E95-A, No. 10, pp. 1683-1691, Oct. 2012.

[12] T. Lenart and V. Öwall, "A 2048 complex point FFT processor using a novel data scaling approach," *Proc. of Int. Symposium on Circuits and Systems (ISCAS '03)*, Vol.4, pp. IV-45 - IV-48, May 2003.

[13] Chao Wang, Woon-Seng Gan, Ching-Chuen Jong, and Jianwen Luo, "A Low-Cost 256-Point FFT Processor for Portable Speech and Audio Applications," *Int. Symposium on Integrated Circuits (ISIC '07)*, pp. 81-84, Sept. 2007.

[14] 3GPP TR 25.814 v7.1.00, "Physical Layer Aspects for Evolved UTRA," 2006.

[15] ETSI, "Digital Video Broadcasting (DVB); Framing Structure, Channel Coding and Modulation for Digital Terrestrial Television," ETSI EN 300 744 v1.4.1, 2001.