**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH**

**KHOA ĐIỆN – ĐIỆN TỬ**

**NGÀNH CÔNG NGHỆ - KỸ THUẬT MÁY TÍNH**



**BÁO CÁO MÔN HỌC**

**KIẾN TRÚC VÀ TỔ CHỨC MÁY TÍNH**

**ĐỀ TÀI**

**THIẾT KẾ MẠCH NHÂN 8 BIT**

**SỬ DỤNG VERILOG HDL**

**GVHD:** TS**.** Phạm Văn Khoa

**NHÓM SINH VIÊN THỰC HIỆN:**

|  |  |
| --- | --- |
| La Công Lộc | 22119098 |
| Trần Đoàn Trung | 22119144 |
| Nguyễn Anh Tuấn | 22119148 |
| Nguyễn Võ Hoàng | 22119076 |
| Hà Minh Hoàng | 22119075 |
| Đỗ Thị Ngọc Yến | 22119159 |

***Thành phố Hồ Chí Minh, Tháng 6 năm 2024***

**MỤC LỤC**

[1. Giới thiệu 1](#_Toc169377247)

[2. Tổng quan 1](#_Toc169377248)

[2.1. Đối tượng nghiên cứu 1](#_Toc169377249)

[2.2. Công cụ hỗ trợ 1](#_Toc169377250)

[2.3. Phân chia công việc 2](#_Toc169377251)

[3. Thực hiện 3](#_Toc169377252)

[3.1. Mạch cộng 3](#_Toc169377253)

[3.1.1. Bộ cộng toàn phần FA 3](#_Toc169377254)

[3.1.2. Bộ cộng bán phần HA 3](#_Toc169377255)

[3.1.3. Mạch cộng Ripple Carry Adder 4](#_Toc169377256)

[3.1.4. Mạch cộng Carry Look Ahead 5](#_Toc169377257)

[3.2. Mạch nhân 6](#_Toc169377258)

[3.2.1. Mạch nhân cột sử dụng mạch cộng RCA và CLA 6](#_Toc169377259)

[3.2.1.1. Tổng quan 6](#_Toc169377260)

[3.2.1.2. Nguyên lý hoạt động 7](#_Toc169377261)

[3.2.2. Mạch nhân mảng. 9](#_Toc169377262)

[3.2.2.1. Tổng quan 9](#_Toc169377263)

[3.2.2.2. Nguyên lý hoạt động 9](#_Toc169377264)

[3.2.3. Mạch nhân cây Wallace 11](#_Toc169377265)

[3.2.3.1. Tổng quan 11](#_Toc169377266)

[3.2.3.2. Nguyên lý hoạt động 11](#_Toc169377267)

[3.2.4. Mạch nhân Vedic 14](#_Toc169377268)

[3.2.4.1. Tổng quan 14](#_Toc169377269)

[3.2.4.2. Mạch nhân Vedic 2x2 14](#_Toc169377270)

[3.2.4.3. Mạch nhân Vedic 4x4 15](#_Toc169377271)

[3.2.4.4. Mạch nhân Vedic 8x8 17](#_Toc169377272)

[4. Kết quả thực hiện 18](#_Toc169377273)

[4.1. Mạch cộng 18](#_Toc169377274)

[4.1.1. Mạch cộng RCA 18](#_Toc169377275)

[4.1.2. Mạch cộng CLA 19](#_Toc169377276)

[4.2. Mạch nhân 20](#_Toc169377277)

[4.2.1. Mạch nhân cột sử dụng mạch cộng RCA 20](#_Toc169377278)

[4.2.2. Mạch nhân cột sử dụng mạch cộng CLA 21](#_Toc169377279)

[4.2.3. Mạch nhân mảng 22](#_Toc169377280)

[4.2.4. Mạch nhân cây Wallace 23](#_Toc169377281)

[4.2.5. Mạch nhân Vedic 8x8 24](#_Toc169377282)

[4.3. Đánh giá – So sánh 25](#_Toc169377283)

[5. Kết luận và phương hướng phát triển 27](#_Toc169377284)

[5.1. Kết luận 27](#_Toc169377285)

[5.2. Phương hướng phát triển 27](#_Toc169377286)

[6. Tài liệu tham khảo 28](#_Toc169377287)

**DANH MỤC CÁC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| HA | Half Adder |
| FA | Full Adder |
| RCA | Ripple Carry Adder |
| CLA | Carry Look - Ahead |
| LUT | Look – up table |
| HDL | Hardware Description Language |

1. **Giới thiệu**

Trong bối cảnh công nghệ ngày càng phát triển, việc nghiên cứu và phát triển các hệ thống vi mạch có hiệu năng cao, tiết kiệm năng lượng và chi phí thấp trở thành một trong những mục tiêu quan trọng của ngành điện tử và máy tính. Một trong những thành phần cơ bản và quan trọng của các hệ thống số là mạch nhân. Mạch nhân được sử dụng rộng rãi trong nhiều ứng dụng khác nhau, từ các bộ vi xử lý đơn giản trong các thiết bị gia dụng, hệ thống điều khiển phức tạp trong công nghiệp cho đến hệ thống các siêu máy tính (các dòng vi xử lý, vi điều khiển: Atmel AVR series, Microchip PIC series, Intel 8051 series, MSP430,…)

Mục tiêu của đề tài Thiết kế mạch nhân 8 bit sử dụng Verilog HDL là thiết kế và mô phỏng bằng ngôn ngữ mô tả phần cứng một mạch nhân 8 bit hiệu quả bằng nhiều phương pháp nhân, đáp ứng các tiêu chí về tốc độ xử lý, độ chính xác, diện tích. Để đạt được mục tiêu này, nhóm tác giả sẽ tiến hành các bước nghiên cứu bao gồm phân tích các thuật toán nhân số, lựa chọn mô tả cấu trúc bằng ngôn ngữ mô tả phần cứng, mô phỏng và kiểm tra tính đúng đắn của mạch trên các phần mềm chuyên dụng, cũng như đánh giá hiệu năng thông qua các thông số kỹ thuật đã đề ra.

1. **Tổng quan**
   1. **Đối tượng nghiên cứu**

* Mạch cộng: Mạch cộng là phần tử con bắt buộc phải có để thực thi các phép nhân số học (mạch cộng RCA, CLA,…). Mạch cộng sử dụng các bộ cộng toàn phần (FA) và bộ cộng bán phần (HA) cấu tạo bởi các cổng logic cơ bản,
* Mạch nhân: Mạch nhân được thiết lập dựa trên việc kết nối các mạch cộng theo một thuật toán, phương pháp nhất định. (Array, Wallace Tree,…).
* Ngôn ngữ mô tả phần cứng Verilog: sử dụng để thiết kế và mô phỏng các hệ thống điện tử, từ các mạch tích hợp đơn giản đến các vi xử lý phức tạp. Nó cho phép các nhà thiết kế mô tả cấu trúc và hành vi của mạch ở các mức độ trừu tượng khác nhau.
* Các thao tác trên phần mềm để phân tích, thiết kế, tổng hợp.
  1. **Công cụ hỗ trợ**

Vivado 2022: là một phần mềm thiết kế và mô phỏng mạch tích hợp do Xilinx phát triển, hỗ trợ toàn diện từ lập trình FPGA. Vivado cung cấp môi trường mạnh mẽ và trực quan cho việc thiết kế các hệ thống số phức tạp.

* 1. **Phân chia công việc**

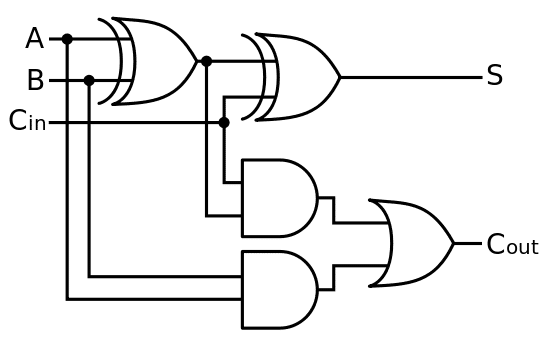
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tên  Nội dung | La Công Lộc | Trần Đoàn Trung | Nguyễn Anh Tuấn | Nguyễn Võ Hoàng | Hà Minh Hoàng | Đỗ Thị Ngọc Yến |
| MSSV | 22119098 | 22119144 | 22119148 | 22119076 | 22119075 | 22119159 |
| Mạch cộng RCA |  |  |  |  |  | **x** |
| Mạch cộng CLA | **x** |  |  |  | **x** |  |
| Nhân cột | **x** |  |  | **x** |  |  |
| Mạch nhân mảng | **x** |  |  |  |  |  |
| Mạch nhân cây Wallace | **x** |  | **x** |  |  |  |
| Mạch nhân Vedic |  | **x** | **x** |  |  |  |
| Thiết kế - mô phỏng | **x** | **x** | **x** | **x** | **x** | **x** |
| Tổng hợp (Synthesis) | **x** |  |  |  |  |  |
| Thuyết trình (PPT) | **x** | **x** | **x** | **x** | **x** | **x** |
| Báo cáo | **x** | **x** |  |  |  |  |

**Bảng I. Bảng phân chia công việc**

1. **Thực hiện**
   1. **Mạch cộng**
      1. **Bộ cộng toàn phần FA**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **Cin** | **S** | **Cout** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**Bảng II. Bảng trạng thái mạch FA**



**Hình 3.1. Sơ đồ mạch FA**

Bộ cộng toàn phần hai số nhị phân 1 bit có ba đầu vào A, B, Cin và hai đầu ra S, Cout, Cin mặc định là 0. Dựa vào bảng trạng thái và sơ đồ ta có được các hàm:

|  |  |
| --- | --- |
| S = A⊕B⊕Cin  Cout= AB + (A⊕B)Cin | Trong đó  A,B,C là các đầu vào  S: kết quả  C số nhớ |

* + 1. **Bộ cộng bán phần HA**

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **S** | **Cout** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

A black background with a black square

Description automatically generated with medium confidence

**Bảng III. Bảng trạng thái HA**

**Hình 3.2. Sơ đồ bộ cộng HA**

Bộ cộng bán phần hai số nhị phân 1 bit có hai ngõ vào A,B và hai ngõ ra S, C. Dựa vào bảng trạng thái và sơ đồ khối ta có được các hàm:

|  |  |
| --- | --- |
| S = A⊕B  C= AB | Trong đó:  A,B là hai số đầu vào 1 bit  S là kết quả  C là số nhớ |

* + 1. **Mạch cộng Ripple Carry Adder**

Mạch cộng RCA là mạch cộng hai số nhị phân n bit. Ripple Carry có nghĩa là việc cộng được thực hiện theo một cách nối dây liên tiếp từ bit thấp đến bit cao, và các bit nhớ truyền dẫn từ bit thấp đến bit cao như một "ripple" (sóng truyền dẫn)

A diagram of a machine

Description automatically generated

**Hình 3.3. Sơ đồ mạch cộng RCA 8 bit**

Mạch cộng RCA 8 bit sử dụng 8 bộ FA kết nối nối tiếp với nhau bằng cách nối ngõ ra Cout của bộ FA thứ k với ngõ vào Cin của bộ FA k+1. Mạch cộng RCA 8 bit có 16 ngõ vào tương đương với 8 cặp A[i]B[i] của 8 bộ FA. Kết quả là số nhị phân 8 bit, trong đó carry (7) là MSB, S0 là LSB.

A diagram of a block diagram

Description automatically generated

**Hình 3.4. Sơ đồ nguyên lý mạch cộng RCA**

* **Nguyên lý hoạt động**
* Các đầu vào A,B được đưa vào mạch ở từng bộ FA (đầu vào của bộ FA đầu tiên thì Cin mặc định là 0
* Đầu ra tại mỗi bộ FA thứ i là S[i], Cout[i], tiếp theo đầu vào tại bộ FA thứ i+1 sẽ thực hiện cộng 3 số đầu vào A[i+1] + B[i+1] + Cout[i]. Lưu ý, các bộ cộng thực hiện tuần tự vì phải chờ đầu ra Cout của bộ cộng phía trước nó.
* Sau khi thực hiện phép cộng lần lượt ở 8 bộ FA thì sẽ được kết quả là số nhị phân 8 bit, với Cout[7] là LSB, S[0] là MSB.

**Ưu điểm:**

* Dễ triển khai, đơn giản: Sử dụng các bộ cộng toàn phần mắc nối tiếp.
* Thích hợp dùng trong các bộ nhân đơn giản, không yêu cầu tốc độ cao.

**Hạn chế:**

* Độ trễ cao: Các bộ cộng phải chở đầu ra Cout của các bộ cộng các bit trước đó
* Khi số lượng bit tăng thì độ trễ cũng tăng, hiệu suất giảm.
  + 1. **Mạch cộng Carry Look Ahead**

Mạch cộng RCA có độ trễ cao do việc thực hiện phép cộng các số đầu vào ở các bộ FA diễn ra tuần tự. Mạch cộng CLA sẽ giải quyết được vấn đề về độ trễ, giúp các bộ FA thực hiện đồng thời.

A diagram of a machine

Description automatically generated

**Hình 3.5. Sơ đồ mạch cộng CLA 8 bit**

Mạch cộng CLA gồm 8 bộ cộng FA và một mạch tổ hợp. Mạch tổ hợp có chức năng tính toán trước các số nhớ Cout ở từng cặp bit trước khi đưa vào từng bộ cộng FA. Tương tự như RCA, mạch cộng CLA 8 bit có 16 ngõ vào tương đương với 8 cặp A[i]B[i] của 8 bộ FA. Kết quả là số nhị phân 8 bit, trong đó carry (7) là MSB, S0 là LSB.

**A diagram of a mathematical equation

Description automatically generated**

**Hình 3.6. Sơ đồ nguyên lý mạch cộng CLA**

* **Nguyên lý hoạt động**
* Các đầu vào A,B được đưa vào mạch ở từng bộ FA (đầu vào của bộ FA đầu tiên thì Cin mặc định là 0
* Đồng thời, các đầu vào A,B cũng được đưa vào mạch tổ hợp. Mạch tổ hợp tính toán trước các số Cout[i] của từng bộ cộng thứ i.
* Tiếp theo, tất cả các bộ cộng sẽ đồng thời thực hiện phép cộng A[i+1] + B[i+1] + Cout[i].
* Kết quả là số nhị phân 8 bit, trong đó carry (7) là MSB, S0 là LSB.

**Ưu điểm:**

* Độ trễ thấp: Giảm đáng kể độ trễ so với RCA, vì ta đã tính trước các Carry nên việc thực hiện ở các bộ cộng diễn ra đồng thời
* Hiệu suất cao hơn mạch cộng RCA
* Hiệu quả với số lượng bit lớn

**Hạn chế:**

* Phức tạp và tốn nhiều tài nguyên hơn mạch cộng RCA
  1. **Mạch nhân**
     1. **Mạch nhân cột sử dụng mạch cộng RCA và CLA**
        1. **Tổng quan**

Phương pháp nhân theo cột truyền thống thực hiện bằng cách tính tích từng phần của hai số nhị phân, sau đó cộng dồn các kết quả tạm thời này theo từng cột tương ứng. Quá trình này bao gồm việc dịch bit và cộng dồn các kết quả trung gian tuần tự, tạo ra một phép nhân với độ trễ tăng dần theo số lượng bit. Phương pháp này dễ hiểu và thực hiện đơn giản nhưng có hiệu suất kém do thực hiện tuần tự, độ trễ cao.

* + - 1. **Nguyên lý hoạt động**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **B7** | **B6** | **B5** | **B4** | **B3** | **B2** | **B1** | **B0** |
| **A7** | **A6** | **A5** | **A4** | **A3** | **A2** | **A1** | **A0** |

**Bảng IV. Hai số 8 bits**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **P0** |  |  |  |  |  |  |  | **A0B7** | **A0B6** | **A0B5** | **A0B4** | **A0B3** | **A0B2** | **A0B1** | **A0B0** |
| **P1** |  |  |  |  |  |  | **A1B7** | **A1B6** | **A1B5** | **A1B4** | **A1B3** | **A1B2** | **A1B1** | **A1B0** |  |
| **N1**  **[8:0]** |  |  |  |  |  | **N1**  **[8]** | **N1**  **[7]** | **N1**  **[6]** | **N1**  **[5]** | **N1**  **[4]** | **N1**  **[3]** | **N1**  **[2]** | **N1**  **[1]** | **N1**  **[0]** |  |
| **P3** |  |  |  |  |  | **A2B7** | **A2B6** | **A2B5** | **A2B4** | **A2B3** | **A2B2** | **A2B1** | **A2B0** |  |  |
| **N2**  **[8:0]** |  |  |  |  | **N2**  **[8]** | **N2**  **[7]** | **N2**  **[6]** | **N2**  **[5]** | **N2**  **[4]** | **N2**  **[3]** | **N2**  **[2]** | **N2**  **[1]** | **N2**  **[0]** |  |  |
|  |  |  |  |  | **A3B7** | **A3B6** | **A3B5** | **A3B4** | **A3B3** | **A3B2** | **A3B1** | **A3B0** |  |  |  |
|  |  |  |  | **A4B7** | **A4B6** | **A4B5** | **A4B4** | **A4B3** | **A4B2** | **A4B1** | **A4B0** |  |  |  |  |
|  |  |  | **A5B7** | **A5B6** | **A5B5** | **A5B4** | **A5B3** | **A5B2** | **A5B1** | **A5B0** |  |  |  |  |  |
|  |  | **A6B7** | **A6B6** | **A6B5** | **A6B4** | **A6B3** | **A6B2** | **A6B1** | **A6B0** |  |  |  |  |  |  |
|  | **A7B7** | **A7B6** | **A7B5** | **A7B4** | **A7B3** | **A7B2** | **A7B1** | **A7B0** |  |  |  |  |  |  |  |
| **N7**  **[8]** | **N7**  **[7]** | **N7**  **[6]** | **N7**  **[5]** | **N7**  **[4]** | **N7**  **[3]** | **N7**  **[2]** | **N7**  **[1]** | **N7**  **[0]** | **N6**  **[0]** | **N5**  **[0]** | **N4**  **[0]** | **N3**  **[0]** | **N2**  **[0]** | **N1**  **[0]** | **A0B0** |
| **R**  **15** | **R**  **14** | **R**  **13** | **R**  **12** | **R**  **11** | **R**  **10** | **R**  **9** | **R**  **8** | **R**  **7** | **R**  **6** | **R**  **5** | **R**  **4** | **R**  **3** | **R**  **2** | **R**  **1** | **R**  **0** |

**Bảng V. Tích từng phần khi nhân hai số 8 bit A,B**

* **Nguyên lý hoạt động**
* Đầu tiên, tính tích từng phần rồi đặt tính như thông thường.
* Ta sẽ tính lần lượt cộng dồn từng hàng của các tích từng phần.
* Bảng trên minh hoạ, sau khi đã có được các tính từng phần, A0B0 chính là R[0] (LSB), sau đó ta đưa hai dòng đầu tiên P[0], P[1] vào mạch cộng RCA 8 bit (hoặc CLA) ta sẽ được kết quả là một số N1[8:0] 9 bit, N1[0] chính là R[1].
* Sau đó ta đưa N1[8:1] và dòng tích P3 vào mạch cộng RCA 8 bit (hoặc CLA) ta sẽ được kết quả là một số N2[8:0] 9 bit, N2[0] chính là R[2].
* Tương tự như hai bước trên, sau khi thực hiện cộng xong dòng tích từng phần P[7] ta sẽ thu được kết quả là một số R[15:0] 16 bit.

So sánh theo lý thuyết, mạch nhân sử dụng bộ cộng CLA sẽ có độ trễ thấp hơn, nhưng phức tạp hơn (vì yêu cầu tính toán trước các số nhớ) mạch cộng RCA.

* + 1. **Mạch nhân mảng.**
       1. **Tổng quan**

Phương pháp nhân mảng là một kỹ thuật nhân nhị phân sử dụng cấu trúc mảng để sắp xếp các phần tử nhân và cộng trong một lưới hình chữ nhật. Cấu trúc giúp đơn giản hóa thiết kế phần cứng và dễ dàng mở rộng quy mô. Phép nhân mảng thường được sử dụng trong các ứng dụng yêu cầu thiết kế phần cứng đơn giản và dễ thực hiện, tốc độ thấp.

* + - 1. **Nguyên lý hoạt động**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | **A0B7** | **A0B6** | **A0B5** | **A0B4** | **A0B3** | **A0B2** | **A0B1** | **A0B0** |
|  |  |  |  |  |  | **A1B7** | **A1B6** | **A1B5** | **A1B4** | **A1B3** | **A1B2** | **A1B1** | **A1B0** |  |
|  |  |  |  |  | **A2B7** | **A2B6** | **A2B5** | **A2B4** | **A2B3** | **A2B2** | **A2B1** | **A2B0** |  |  |
|  |  |  |  | **A3B7** | **A3B6** | **A3B5** | **A3B4** | **A3B3** | **A3B2** | **A3B1** | **A3B0** |  |  |  |
|  |  |  | **A4B7** | **A4B6** | **A4B5** | **A4B4** | **A4B3** | **A4B2** | **A4B1** | **A4B0** |  |  |  |  |
|  |  | **A5B7** | **A5B6** | **A5B5** | **A5B4** | **A5B3** | **A5B2** | **A5B1** | **A5B0** |  |  |  |  |  |
|  | **A6B7** | **A6B6** | **A6B5** | **A6B4** | **A6B3** | **A6B2** | **A6B1** | **A6B0** |  |  |  |  |  |  |
| **A7B7** | **A7B6** | **A7B5** | **A7B4** | **A7B3** | **A7B2** | **A7B1** | **A7B0** |  |  |  |  |  |  |  |

**Bảng VI. Tích từng phần hai số nhị phân 8 bit**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A7B7** | **A6B7** | **A5B7** | **A4B7** | **A3B7** | **A2B7** | **A1B7** | **A0B7** | **A0B6** | **A0B5** | **A0B4** | **A0B3** | **A0B2** | **A0B1** | **A0B0** |
|  | **A7B6** | **A6B6** | **A5B6** | **A4B6** | **A3B6** | **A2B6** | **A1B6** | **A1B5** | **A1B4** | **A1B3** | **A1B2** | **A1B1** | **A1B0** |  |
|  |  | **A7B5** | **A6B5** | **A5B5** | **A4B5** | **A3B5** | **A2B5** | **A2B4** | **A2B3** | **A2B2** | **A2B1** | **A2B0** |  |  |
|  |  |  | **A7B4** | **A6B4** | **A5B4** | **A4B4** | **A3B4** | **A3B3** | **A3B2** | **A3B1** | **A3B0** |  |  |  |
|  |  |  |  | **A7B3** | **A6B3** | **A5B3** | **A4B3** | **A4B2** | **A4B1** | **A4B0** |  |  |  |  |
|  |  |  |  |  | **A7B2** | **A6B2** | **A5B2** | **A5B1** | **A5B0** |  |  |  |  |  |
|  |  |  |  |  |  | **A7B1** | **A6B1** | **A6B0** |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **A7B0** |  |  |  |  |  |  |  |

**Bảng VII. Tích từng phần sau khi tịnh tiến**

A group of black and white symbols

Description automatically generated

**Hình 3.7. Sơ đồ mạch nhân mảng**

Mạch nhân mảng sử dụng 8 bộ cộng HA và 48 bộ cộng HA, được kết nối hình tháp ngược như sơ đồ trên, các đầu vào và đầu ra của từng bộ cộng là các cặp tích từng phần A,B và số nhớ.

* **Nguyên lý hoạt động**

- Đầu tiên, tính các tích từng phần A[i]B[i], như **Bảng VI**.

- Tiếp theo, ta tiến hành tịnh tiến như **Bảng VII**, dựa vào **Bảng VII** ta tiến hành kết nối các bộ cộng với các ngõ vào – ra tương ứng như **Hình 3.7**

- Các bộ cộng thực thi tuần tự lan truyền các ngõ ra Cout, S từ trái qua phải từ trên xuống dưới.

- Minh họa cột P[2], cột P[2] có 1 bộ cộng FA và 1 bộ cộng HA. Bộ cộng FA có 3 đầu vào A0B2, A1B1 và Cout[0] lấy từ đầu ra bộ FA của cột P[1], cho ra S và Cout của nó.

- Giá trị S sẽ được dùng cho bộ HA ngay phía dưới bộ FA, giá trị Cout sẽ được dùng cho bộ FA phái bên phải.

- Tương tự, các bộ FA và HA còn lại cũng thực hiện tuần tự bằng cách sử dụng đầu ra Cout của bộ cộng phía bên phải và S của bộ cộng ở phía trên theo sơ đồ **Hình 3.7**.

- Kết quả sẽ là một số P[15:0] 16 bit.

* + 1. **Mạch nhân cây Wallace**
       1. **Tổng quan**

Phương pháp nhân Wallace Tree Multipliers là một kỹ thuật nhân nhị phân tốc độ cao, sử dụng cấu trúc cây Wallace để giảm số lượng tầng cộng trong quá trình tính toán tích. Cấu trúc này giúp giảm đáng kể độ trễ và tăng hiệu suất so với các phương pháp truyền thống như nhân hàng loạt hoặc cây Booth. Wallace Tree Multipliers thường được ứng dụng trong các hệ thống số học hiệu năng cao như bộ vi xử lý và các bộ xử lý tín hiệu số (DSP).

* + - 1. **Nguyên lý hoạt động**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | **A0B7** | **A0B6** | **A0B5** | **A0B4** | **A0B3** | **A0B2** | **A0B1** | **A0B0** |
|  |  |  |  |  |  | **A1B7** | **A1B6** | **A1B5** | **A1B4** | **A1B3** | **A1B2** | **A1B1** | **A1B0** |  |
|  |  |  |  |  | **A2B7** | **A2B6** | **A2B5** | **A2B4** | **A2B3** | **A2B2** | **A2B1** | **A2B0** |  |  |
|  |  |  |  | **A3B7** | **A3B6** | **A3B5** | **A3B4** | **A3B3** | **A3B2** | **A3B1** | **A3B0** |  |  |  |
|  |  |  | **A4B7** | **A4B6** | **A4B5** | **A4B4** | **A4B3** | **A4B2** | **A4B1** | **A4B0** |  |  |  |  |
|  |  | **A5B7** | **A5B6** | **A5B5** | **A5B4** | **A5B3** | **A5B2** | **A5B1** | **A5B0** |  |  |  |  |  |
|  | **A6B7** | **A6B6** | **A6B5** | **A6B4** | **A6B3** | **A6B2** | **A6B1** | **A6B0** |  |  |  |  |  |  |
| **A7B7** | **A7B6** | **A7B5** | **A7B4** | **A7B3** | **A7B2** | **A7B1** | **A7B0** |  |  |  |  |  |  |  |

**Bảng VIII. Tích từng phần nhân hai số 8 bit**

Sau khi có tích từng phần, sẽ thực hiện nhóm thành nhóm 3 hàng, các hàng còn lại sẽ giữ nguyên thứ tự để thực hiện các bước rút gọn tiếp theo. **Lưu ý, cột có hai số sẽ sử dụng bộ cộng HA, cột có ba số sẽ sử dụng bộ cộng FA**. **Các bộ cộng ở từng bước sẽ được thực hiện đồng thời, hay nói cách khác phép cộng ở từng nhóm sẽ được thực hiện đồng thời**.

|  |  |
| --- | --- |
| **Bước 1: Rút gọn từ 8 xuống 6 hàng**  - Có 8 hàng tích từng phần, ta sẽ nhóm các hàng thành các nhóm 3, những nhóm còn lại sẽ được nhóm ở bước rút gọn tiếp theo.  - Ta nhóm được hai nhóm 3, Nhóm I và Nhóm II.  - Đưa các số ở từng cột của hai nhóm I, II vào bộ cộng để thực hiện phép cộng.  - Thu được Output I và Output II lần lượt là ngõ ra của các bộ cộng của Nhóm I và Nhóm II, chúng được xếp ở các cột và các hàng tương ứng như, hai hàng còn lại giữ nguyên vị trí như **Hình 3.8** để thực hiện bước rút gọn tiếp theo. | A yellow and blue squares with black text  Description automatically generated  **Hình 3.8. Rút gọn từ 8 xuống 6** |
| **Bước 2: Rút gọn từ 6 xuống 4**  - Sau **bước 1**, còn lại 6 hàng, tiến hành nhóm được hai nhóm 3, là Nhóm III và Nhóm IV.  - Thực hiện phép cộng ở Nhóm III và Nhóm IV tương tự như Nhóm I,II ở B1.  - Thu được Output III và Output IV lần lượt là các ngõ ra của các bộ cộng ở từng cột của Nhóm I và Nhóm II, chúng được sắp xếp ở các vị trí tương ứng như **Hình 3.9** để thực hiện bước rút gọn tiếp theo. | **Hình 3.9. Rút gọn từ 6 xuống 4** |
| **Bước 3: Rút gọn từ 4 xuống 3**  - Sau **bước 2,** còn lại 4 hàng, tiến hành nhóm được một nhóm 3, là Nhóm V.  - Thu được Output V là đầu ra của Nhóm V và một hàng còn lại được sắp xếp như Hình 11 để thực hiện bước rút gọn tiếp theo. | A table with numbers and letters  Description automatically generated  **Hình 3.10. Rút gọn từ 4 xuống 3** |
| **Bước 4: Rút gọn từ 3 xuống 2**  - Sau bước 3, còn lại 3 hàng, tiến hành nhóm được một nhóm 3, là Nhóm VI.  - Thu được Output VI là đầu ra của Nhóm VI như Hình 12.  **Bước 5:** Đưa hai hàng của Output VI vào mạch nhân CLA mở rộng.  Thu được kết quả là số nhị phân 16 bit. | A screenshot of a computer  Description automatically generated  **Hình 3.11. Rút gọn từ 3 xuống 2** |

* + 1. **Mạch nhân Vedic**
       1. **Tổng quan**

Phương pháp này được xây dựng dựa trên các nguyên tắc trong Veda, các văn bản cổ của Ấn Độ, và nó giúp thực hiện các phép nhân (và nhiều phép tính khác) một cách nhanh chóng và dễ dàng. Phương pháp này được biết đến với sự tối giản, linh hoạt và tính trực quan trong cách thực hiện các phép tính. Ví dụ, để thiết lập mạch nhân 8x8 sẽ đi từ mạch nhân 2x2 rồi đến 4x4.

* + - 1. **Mạch nhân Vedic 2x2**
* **Nguyên lý hoạt động và sơ đồ khối**

|  |  |
| --- | --- |
| **B1 :** Đầu tiên ta lấy A0 nhân với B0 ta thu được giá trị nhân S0  **B2:** Ta lấy A1 nhân với B0 cộng với A0 nhân với B1 ta thu được giá trị nhân S1 và bit nhớ C1.  **B3 :** Ta lấy A1 nhân với B1 và cộng với bit nhớ C1 trước đó ta thu được giá trị nhân S2 và giá trị nhớ C2  => Kết quả của nhân hai số 2 bit là C2S2S1S0.  - Đầu tiên chúng ta AND a0 và b0 thu được giá trị s0, sau đó ta tiếp tục AND a0 và b1 với a1 và b0 đưa vào khối cộng bán phần ta thu được giá trị s1 và c1, cuối cùng ta cộng giá trị AND a1 và b1 với c1 chúng ta thu được giá trị s2 và c2. | **Hình 3.12. Sơ đồ khối Vedic 2x2**    **Hình 3.13. Sơ đồ nguyên lý 2x2** |

* **Lưu đồ - thuật toán**

|  |  |
| --- | --- |
| - Đầu tiên ta khai báo module có tên là Vedic\_2x2 có 2 ngõ vào “a,b” 2 bit và 1 ngõ ra “s” 4 bit.  - Tiếp theo ta khai báo các dây lần lượt là w1 tới w4 sẽ có vai trò là lưu kết quả AND của các tích từng phần hai số “a,b” theo nguyên lý của Vedic2x2 và dây c1 sẽ lưu giá trị bit nhớ sau khi cộng bán phần.  - Dây w1 sẽ lưu giá trị của a0 và b0, dây w2 sẽ lưu giá trị của a1 và b0, dây w3 sẽ lưu giá trị của a0 và b1, dây w4 sẽ lưu giá trị của a1 và b1. Do dây w1 không còn thực hiện phép tính nên ta gắn dây w1 nào trực tiếp kết quả nơi có trọng số thấp nhất là s0. Tiếp theo ta sẽ sử dụng 2 khối cộng bán phần-half adder trong đó khối cộng bán phần đầu tiên ta dùng để cộng giá trị của w2 và w3, ta thu được giá trị s1 và c1. Khối cộng bán phần thứ 2 sẽ cộng giá trị w4 và c1, ta thu được s2 và s3. | **A diagram of a number of numbers  Description automatically generated**A diagram of a graph  Description automatically generated with medium confidence  **Hình 3.15. Sơ đồ khối**  **Hình 3.14. Lưu đồ** |

* + - 1. **Mạch nhân Vedic 4x4**
* **Nguyên lý hoạt động**

Tương tự với cách nhân của 2x2, ta áp dụng cho 4x4 như sau

Ta có ví dụ:  Cho a và b là các số 4 bit: a = 1001, b = 1011.

A screenshot of a step 2

Description automatically generated with medium confidence

**Hình 3.16. Các bước nhân Vedic 4x4**

A close-up of a math equation

Description automatically generated

**Hình 3.17. Ví dụ nhân hai số 4 bit a,b**

Dựa trên nguyên lý hoạt động, mạch Vedic 4x4 sẽ được thiết lập dựa trên mạch Vedic 2x2. Bộ cộng FA 6 bit được sử để cộng các cặp đầu ra của từng cặp Vedic 2x2. Lưu ý, các bộ FA ở từng lớp thực hiện đồng thời. **Hình 3.18** mô tả cụ thể cách thiết lập mạch nhân Vedic 4x4.

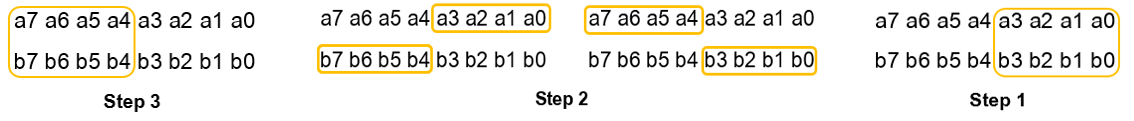
|  |  |
| --- | --- |
| **Hình 3.18. Sơ đồ khối Vedic 4x4** | **A diagram of a graph  Description automatically generated with medium confidence**  **Hình 3.19. Lưu đồ thuật toán Vedic 4x4** |

* + - 1. **Mạch nhân Vedic 8x8**
* **Nguyên lý hoạt động**

Ta có ví dụ sau:

Cho a và b là các số 8 bit:

a = 10011100, b = 10001101.



**Hình 3.20. Các bước nhân Vedic 8x8**

A close-up of a number

Description automatically generated

**Hình 3.21. Ví dụ mạch nhân hai số 8 bit Vedic 8x8**

Dựa trên nguyên lý thực thi, mạch nhân Vedic 8x8 sẽ được thiết lập từ mạch nhân Vedic 4x4 (thiết lập từ Vedic 2x2). Bộ cộng FA 6 bit được sử để cộng các cặp đầu ra của từng cặp Vedic 2x2. Lưu ý, các bộ FA ở từng lớp thực hiện đồng thời. **Hình 3.22** mô tả cụ thể cách thiết lập mạch nhân Vedic 4x4.

|  |  |
| --- | --- |
| **Hình 3.22. Sơ đồ khối Vedic 8x8** | A diagram of a graph  Description automatically generated with medium confidence  **Hình 3.23. Lưu đồ thuật toán Vedic 8x8** |

1. **Kết quả thực hiện**
   1. **Mạch cộng**

Sau các bước phân tích, thiết kế, RTL coding, mô phỏng (Simulation) và tổng hợp (Synthesis) ta thu được các kết quả ở từng mạch cộng và mạch nhân, tất cả các mạch đều chạy trên kit **Spartan – 7 series , model xc7s6cpga196 – 1.**

* + 1. **Mạch cộng RCA**

A screenshot of a computer

Description automatically generated

**Hình 4.1. Kết quả chạy mô phỏng mạch cộng RCA**

**Hình 4.1** biểu thị các phép tính và kết quả khi cộng 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tổng S[8:0] 9 bit ( 200 + 190 = 390, 144 + 89 = 233,…).

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated  **Hình 4.2. Mạch cộng RCA - kit** | A white and black text  Description automatically generated with medium confidence  **Hình 4.3. RCA - LUTs** |

A screenshot of a computer

Description automatically generated

**Hình 4.4. RCA - Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.3, Hình 4.4.**

|  |  |
| --- | --- |
| **LUTs** | **Delay (ns)** |
| 11 | 6.398 |

**Bảng IX. Thông số sau khi Synthesis - Ultilization**

* + 1. **Mạch cộng CLA**

A screenshot of a computer

Description automatically generated

**Hình 4.5.. Kết quả chạy mô phỏng mạch cộng CLA**

**Hình 4.5** biểu thị các phép tính và kết quả khi cộng 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tổng S[8:0] 9 bit ( 200 + 190 = 390, 144 + 89 = 233,…).

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated  **Hình 4.6. CLA - kit** | A white rectangular box with black text  Description automatically generated  **Hình 4.7. CLA - LUTs** |

A screenshot of a computer

Description automatically generated

**Hình 4.8. CLA - Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.7, Hình 4.8**.

|  |  |
| --- | --- |
| **LUTs** | **Delay (ns)** |
| 17 | 6.260 |

**Bảng X. Thông số sau khi Synthesis - Ultilization**

* 1. **Mạch nhân**
     1. **Mạch nhân cột sử dụng mạch cộng RCA**

**A screenshot of a computer

Description automatically generated**

**Hình 4.9. Kết quả sau khi chạy mô phỏng mạch nhân RCA**

**Hình 4.9** biểu thị các phép tính và kết quả khi nhân 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tích S[15:0] 9 bit ( 200 x 190 = 38000, 144 + 89 = 12816,…).

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated  **Hình 4.10. Nhân RCA - kit** | A white and black text  Description automatically generated with medium confidence  **Hình 4.11. Nhân RCA - LUTs** |

A close up of a sign

Description automatically generated

**Hình 4.12. Nhân RCA – Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.11, Hình 4.12**.

|  |  |  |  |
| --- | --- | --- | --- |
| **LUTs** | **Delay (ns)** | **FA** | **HA** |
| 128 | 14.864 | 56 | 0 |

**Bảng XI. Thông số sau khi Synthesis - Ultilization**

* + 1. **Mạch nhân cột sử dụng mạch cộng CLA**

A screenshot of a computer

Description automatically generated

**Hình 4.13. Kết quả sau khi chạy mô phỏng mạch nhân CLA**

**Hình 4.13** biểu thị các phép tính và kết quả khi nhân 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tích S[15:0] 9 bit ( 200 x 190 = 38000, 144 + 89 = 12816,…).

|  |  |
| --- | --- |
| A screenshot of a computer program  Description automatically generated  **Hình 4.14. Nhân CLA - kit** | A screenshot of a computer  Description automatically generated  **Hình 4.15. Nhân CLA - LUTs** |

A close up of a sign

Description automatically generated

**Hình 4.16. Nhân CLA – Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.15, Hình 4.16**.

|  |  |  |  |
| --- | --- | --- | --- |
| **LUTs** | **Delay (ns)** | **FA** | **HA** |
| 134 | 14.624 | 56 | 0 |

**Bảng XII. Thông số sau khi Synthesis - Ultilization**

* + 1. **Mạch nhân mảng**

A screenshot of a computer

Description automatically generated

**Hình 4.17. Kết quả sau khi chạy mô phỏng mạch nhân mảng**

**Hình 4.17** biểu thị các phép tính và kết quả khi nhân 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tích S[15:0] 9 bit ( 200 x 190 = 38000, 144 + 89 = 12816,…).

|  |  |
| --- | --- |
| A screenshot of a computer program  Description automatically generated  **Hình 4.18. Nhân mảng - kit** | A white rectangular box with black text  Description automatically generated  **Hình 4.19. Nhân mảng - LUTs** |

A close-up of a sign

Description automatically generated

**Hình 4.20. Nhân mảng - Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.19, Hình 4.20**.

|  |  |  |  |
| --- | --- | --- | --- |
| **LUTs** | **Delay (ns)** | **FA** | **HA** |
| 126 | 13.301 | 48 | 8 |

**Bảng XIII. Thông số sau khi Synthesis - Ultilization**

* + 1. **Mạch nhân cây Wallace**

A screenshot of a computer

Description automatically generated

**Hình 4.21.Kết quả sau khi chạy mô phỏng mạch nhân cây Wallace**

**Hình 4.21** biểu thị các phép tính và kết quả khi nhân 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tích S[15:0] 9 bit ( 200 x 190 = 38000, 144 + 89 = 12816,…).

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated  **Hình 4.22. Wallace - kit** | A screenshot of a computer  Description automatically generated  **Hình 4.23. Wallace - LUTs** |

A close up of a sign

Description automatically generated

**Hình 4.24. Wallace - Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.23, Hình 4.24**.

|  |  |  |  |
| --- | --- | --- | --- |
| **LUTs** | **Delay (ns)** | **FA** | **HA** |
| 109 | 9.933 | 49  (11 của mạch cộng CLA) | 15 |

**Bảng XIV. Thông số sau khi Synthesis - Ultilization**

* + 1. **Mạch nhân Vedic 8x8**

A screenshot of a computer

Description automatically generated

**Hình 4.25. Kết quả sau khi chạy mô phỏng nạch nhân Vedic 8x8**

**Hình 4.25** biểu thị các phép tính và kết quả khi nhân 2 số 8 bit là đúng với lý thuyết, hai số 8 bit A[7:0], B[7:0] cho ra tích S[15:0] 9 bit ( 200 x 190 = 38000, 144 + 89 = 12816,…).

|  |  |
| --- | --- |
| A screenshot of a computer program  Description automatically generated  **Hình 4.26. Vedic 8x8 - kit** | A white and black text  Description automatically generated with medium confidence  **Hình 4.27. Vedic - LUTs** |

A close up of a sign

Description automatically generated

**Hình 4.28. Vedic 8x8 - Delay**

Sau khi mô phỏng và tổng hợp ta thu được các thông số LUT, Delay time được biểu thị ở các **Hình 4.27, Hình 4.28**.

|  |  |  |  |
| --- | --- | --- | --- |
| **LUTs** | **Delay (ns)** | **FA** | **HA** |
| 103 | 11.481 | 96 | 32 |

**Bảng XV. Thông số sau khi Synthesis – Ultilization**

* 1. **Đánh giá – So sánh**
* **Mạch cộng**

|  |  |  |
| --- | --- | --- |
| Mạch cộng  Thông số | **RCA** | **CLA** |
| **LUTs** | 17 | 11 |
| **Delay (ns)** | 6.398 | 6.260 |

**Bảng XVI. Thông số mạch cộng sau khi Synthesis - Ultilization**

Ở quy mô mạch cộng hai số 8 bit thì ta thấy mạch cộng CLA có số LUT và độ trễ nhỏ hơn của mạch cộng RCA. Như đã phân tích ở lý thuyết thì mạch cộng CLA có độ trễ thấp hơn RCA vì nó được trang bị một mạch tính toán trước các số nhớ, sau đó tất cả các bộ FA được thực hiện đồng thời. Mạch cộng CLA phức tạp và khó triển khai hơn RCA. Thông số LUT cho thấy sau khi Synthesis -Ultilization thì mạch cộng CLA tốn ít diện tích hơn ( thông số LUT cũng phụ thuộc một phần vào khâu coding). Nếu mở rộng quy mô lên n bit thì các thông số trên sẽ thay đổi và còn phụ thuộc vào cách triển khai thiết kế của người thiết kế

Tóm lại, mạch cộng CLA sẽ có tốc độ thực thi nhanh hơn, phức tạp và khó triển khai hơn mạch cộng RCA. Mạch cộng CLA được dùng trong các ứng dụng yêu cầu tốc độ cao và số bit lớn. Căn cứ vào nhu cầu và thông số của hệ thống thì ta sẽ chọn mạch cộng phù hợp.

* **Mạch nhân**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mạch nhân  8x8  Thông số | **RCA** | **CLA** | **ARRAY** | **WALLACE** | **VEDIC** |
| **LUTs** | 128 | 134 | 126 | 109 | 103 |
| **Delay (ns)** | 14.864 | 14.624 | 13.301 | 9.933 | 11.481 |
| **FA** | 56 | 56 | 48 | 49 | 96 |
| **HA** | 0 | 0 | 8 | 15 | 32 |

**Bảng XVII. Thông số mạch nhân sau khi Synthesis – Ultilization**

Dựa vào **Bảng XVII**, ta thấy trong các mạch nhân trên, mạch nhân sử dụng RCA có độ trễ lớn nhất (14.864ns), mạch nhân cây Wallace có độ trễ thấp nhất (9.933ns), tốc độ của mạch nhân cây Wallace nhanh hơn mạch nhân RCA (14.864ns), CLA (14.624ns), Array(13.301ns), Vedic(11.481ns) lần lượt là 33,2%, 32,1%, 25,3%, 13,5%.

Mạch nhân CLA (sử dụng mạch cộng CLA) có độ trễ thấp hơn mạch nhân RCA (sử dụng mạch cộng RCA) và có số LUT lớn hơn, điều này là hệ quả như ta đã phân tích ở phần mạch cộng.

Căn cứ tiếp vào thông số LUT, mạch nhân Wallace có số LUT (109) chỉ nhiều hơn số LUT của mạch nhân Vedic (103).

Vậy khi ta muốn muốn thiết kế mạch nhân 8 bit tối ưu khi đã đảm bảo về mặt chi phí thì ta nên chọn phương pháp nhân cây Wallace, phương pháp này tối ưu về diện tích và tốc độ thực thi nhanh. Vì vậy, phương pháp nhân cây Wallace thường được ứng dụng trong các hệ thống số học hiệu năng cao như bộ vi xử lý và các bộ xử lý tín hiệu số (DSP).

Tuy nhiên, khi thiết kế một hệ thống nhất định thì ta nên căn cứ vào nhu cầu và các thông số kỹ thuật cũng như chi phí để lựa chọn phương pháp nhân thích hợp nhằm có được kết quả tối ưu nhất.

1. **Kết luận và phương hướng phát triển**
   1. **Kết luận**

Đề tài Thiết kế mạch nhân 8 bit – mô tả bằng ngôn ngữ Verilog giúp ta hiểu cách mà máy tính thực thi phép nhân, tối ưu tốc độ của các phần tử trong một máy tính. Nghiên cứu nhiều phương pháp nhân, thuật toán khác nhau giúp ta có cái nhìn toàn diện, phân tích được nguyên lý hoạt động, cấu tạo mạch,ưu điểm, hạn chế, thuật toán mà mạch sử dụng, biết được cùng một yêu cầu kỹ thuật thì phương pháp nhân nào là tối ưu để đưa vào hệ thống… Từ đó, đưa ra những giải pháp, thiết kế phù hợp với yêu cầu của khách hàng. Có hiểu biết về các hệ thống máy tính sử dụng mạch nhân 8 bit. Mạch nhân được sử dụng rộng rãi trong nhiều ứng dụng khác nhau, từ các bộ vi xử lý đơn giản trong các thiết bị gia dụng, hệ thống điều khiển phức tạp trong công nghiệp cho đến hệ thống các siêu máy tính ( các dòng vi xử lý, vi điều khiển: Atvel AVR series, Microchip PIC series, Intel 8051 series, MSP430,…)

Ngoài ra, ta học thêm được cách sử dụng ngôn ngữ mô tả phần cứng và phần mềm để triển khai thiết kế, kiểm tra các thông số kỹ thuật của mạch để đảm bảo yêu cầu. Ta áp dụng được các kiến thức về Kiến trúc và tổ chức máy tính và Thiết kế số vào trong đề tài… Tuy nhiên, nhóm tác giả vẫn còn nhiều hạn chế về kiến thức về FPGA, vi xử lý, vi điều khiển, trình độ ngoại ngữ để đọc tài liệu và sử dụng công cụ, thời gian,…nên sẽ có sự sai lệch giữa các thông số lý thuyết và mô phỏng – tổng hợp.

* 1. **Phương hướng phát triển**

Đề tài Thiết kế mạch nhân 8 bit – mô tả bằng ngôn ngữ Verilog tập hợp nhiều kiến thức chuyên môn liên quan đến Kiến trúc tổ chức máy tính, Thiết kế hệ thống số FPGA và tư duy thiết kế, logic, lập trình. Trong tương lai, khi đáp được các kiến thức chuyên môn như đã liệt kê, có đủ thời gian, công cụ, trang thiết bị thì nhóm tác giả có thể hoàn chỉnh đề tài một cách chi tiết và hiệu quả nhất . Cụ thể, như là tối ưu Coding, ràng buộc về thông số thời gian, hiểu rõ quy trình Synthesis, Place&Route, Timing Analysis, Tapout và sau đó là thử nghiệm trên testboard. Mở rộng mạch nhân 8 bit lên nhiều bit và thực hiện được nhiều chức năng số học khác.

1. **Tài liệu tham khảo**

1. Digital Computer Arithmetic Datapath Design Using Verilog HDL, James E. Stine, trang 28, 56-61.

2. Design and Implementation of Wallace Tree Multipliers using Higher Order Compressors, Dhanya Mravi.

3. Report: A 8x8 bit multiplier using Vedic Mathematics, Jyoti Kumawat, Sunil Sharma.

4. Article: Implementation of an Efficient Multiplier based on Vedic Mathematics, Kedar N. Palata, Vinobha K. Nadar , Jatin S. Jethawa , Tushar J. Surwadkar , Rajan S. Deshmukh.

5. Giáo trình Kỹ thuật số, Đại học Sư phạm – Kỹ thuật Thành phố Hồ Chí Minh.