HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGUYỄN TIẾN LẬP

NGHIÊN CỨU THIẾT KẾ MODUL ĐÓNG KHUNG E1 BẰNG FPGA

LUẬN VĂN THẠC SĨ KỸ THUẬT

(***Theo định hướng ứng dụng***)

HÀ NỘI - 2019

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGUYỄN TIẾN LẬP

NGHIÊN CỨU THIẾT KẾ MODUL ĐÓNG KHUNG E1 BẰNG FPGA

CHUYÊN NGÀNH: KỸ THUẬT VIỄN THÔNG

MÃ SỐ: 8.52.02.08

LUẬN VĂN THẠC SĨ KỸ THUẬT

(***Theo định hướng ứng dụng***)

NGƯỜI HƯỚNG DẪN KHOA HỌC:

TS. NGUYỄN NGỌC MINH

HÀ NỘI - 2019

LỜI CAM ĐOAN

Tôi cam đoan đây là công trình nghiên cứu của riêng tôi. Các số liệu và kết quả trình bày trong luận văn là trung thực và chưa được công bố bởi bất kỳ tác giả nào hay ở bất kỳ công trình nào khác.

Hà Nội, tháng 11 năm 2019

Tác giả luận văn

Nguyễn Tiến Lập

LỜI CẢM ƠN

Tôi xin bày tỏ sự biết ơn sâu sắc tới TS. Nguyễn Ngọc Minh, người thầy đã định hướng và hướng dẫn tôi thực hiện thành công đề tài nghiên cứu.

Tôi xin chân thành cảm ơn Ban giám đốc, Khoa Đào tạo sau đại học - Học viên Công nghệ Bưu chính Viễn thông cũng như lãnh đạo, chỉ huy và các đồng chí ở Trung tâm Kỹ thuật thông tin công nghệ cao – Binh chủng Thông tin liên lạc, nơi tôi đang công tác, đã tạo mọi điều kiện thuận lợi cho tôi trong suốt quá trình thực hiện luận văn.

Tôi xin chân thành cảm ơn các thầy cô giáo trong và ngoài trường đã trang bị cho tôi những kiến thức trong quá trình hoàn thành các học phần cao học.

Tôi xin được cám ơn những người thân, bạn bè đã thường xuyên quan tâm, giúp đỡ, chia sẻ kinh nghiệm, cung cấp các tài liệu hữu ích trong thời gian học tập, nghiên cứu cũng như trong suốt quá trình thực hiện luận văn tốt nghiệp.

Cuối cùng, tôi xin chân thành gửi lời cảm ơn tới gia đình đã kiên trì chia sẻ và động viên tôi trong suốt quá trình thực hiện nội dung luận văn.

Hà Nội, tháng 11 năm 2019

Tác giả luận văn

Nguyễn Tiến Lập

MỤC LỤC

THUẬT NGỮ VIẾT TẮT

ASIC	Aplication Specific Integrated Circuit	Mạch tích hợp chuyên dụng
CAS	Channel Associated Signalling	Báo hiệu kênh kết hợp
CPLD	Complex Programmable Logic Device	Thiết bị logic khả trình phức tạp
CPU	Central Processor Unit	Bộ xử lý trung tâm
DCM	Digital Clock Manager	Quản lý xung nhịp kỹ thuật số
DSP	Digital Signal Processor	Bộ xử lý tín hiệu số
EDIF	Electronic Data Interchange Format	Định dạng trao đổi thiết kế điện tử
EMI	ElectroMagnetic Interference	Nhiễu điện từ trường
FDM	Frequency Division Multiplexing	Ghép kênh phân chia theo tần số
FPGA	Field Programmable Gate Array	Mảng cổng lập trình được dạng trường
IC	Integrated Circuit	Mạch tích hợp
IEEE	Institute of Electrical and Electronics Engineers	Viện các kỹ sư điện và điện tử
ISE	Integrated Synthesis Environment	Môi trường tổng hợp tích hợp
MAC	Multiplication And Accumulation	Bộ nhân cộng
PAL	Programmable Array Logic	Logic mảng khả trình
PCM	Pulse Code Modulation	Điều chế mã xung
PDH	Plesiosynchronous Digital Hierarchy	Phân cấp tốc độ số cận đồng bộ
PLA	Programmable Logic Array	Mảng logic khả trình
RAM	Random Access Memory	Bộ nhớ truy xuất ngẫu nhiên
ROM	Read-Only Memory	Bộ nhớ chỉ đọc
SDH	Synchronous Digital Hierarchy	Phân cấp tốc độ số đồng bộ
SPLD	Simple Programmable Logic Device	Thiết bị logic khả trình đơn giản

SRAM	Static Random Access Memory	Bộ nhớ truy xuất ngẫu nhiên tĩnh
TDM	Time Division Multiplexing	Ghép kênh phân chia theo thời gian
TS	Time Slot	Khe thời gian
VHDL	Very High Speed Integrated Circuit Hardware Description Language	Ngôn ngữ mô tả phần cứng mạch tích hợp tốc độ cao
XCITE	Xilinx Controlled Impedance Technology	Công nghệ trở kháng được điều khiển của Xilinx

DANH MỤC CÁC BẢNG

DANH MỤC HÌNH VẼ

MỞ ĐẦU

Ngày nay cùng với các công nghệ xử lý tín hiệu số hiện đại, các thiết bị truyền dẫn số nói chung đã và đang thay thế dần các phương tiện truyền dẫn cũ và lạc hậu. Việc ứng dụng các kỹ thuật xử lý tín hiệu số hiện đại trong các hệ thống thông tin liên lạc ngày càng đem lại lợi ích và hiệu quả rõ rệt, nhất là đối với các thiết bị số được chế tạo trên công nghệ mới như xử lý số (DSP), chip trắng lập trình được (FPGA) và công nghệ chế tạo mạch in nhiều lớp tốc độ cao.

Hiện nay, việc mềm hóa các chip chuyên dụng đang phát triển mạnh, đem lại khả năng thích ứng cao và có thể tái sử dụng, cấu hình lại theo yêu cầu. Trên thế giới xu hướng sử dụng phần mềm để định nghĩa phần cứng và thực hiện trên chíp trắng đã được sử dụng rộng rãi, có thể nói hầu như tất cả các thiết bị băng rộng cũng như băng hẹp hiện đại đều sử dụng công nghệ này thay thế dần công nghệ chíp chuyên dụng như trước đây.

Đối với các cơ sở nghiên cứu trong nước, đây thực sự là cơ hội để các cán bộ nghiên cứu áp dụng công nghệ hiện đại vào trong thiết kế sản xuất. Việc mềm hóa các phần cứng mang lại nhiều hiệu quả thiết thực. Giảm thiểu độ rủi ro so với khi thiết kế hoàn toàn bằng phần cứng. Điều quan trọng là có thể thiết kế một lần và dùng lại, có phần mềm hỗ trợ mô phỏng trước khi thực hiện trên phần cứng. Đó là những lợi ích mà phương pháp thiết kế mới mang lại.

Với ưu điểm vượt trội của công nghệ FPGA và ngôn ngữ mô tả phần cứng (VHDL), tôi đã chọn đề tài luận văn là: "**Nghiên cứu thiết kế modul đóng khung E1 bằng FPGA**".

Mục đích nghiên cứu

Mục đích của đề tài là nghiên cứu kỹ thuật đóng khung dữ liệu E1, ứng dụng thuật toán xử lý tín hiệu số thiết kế modul đóng khung E1 trên công nghệ chíp trắng lập trình được (FPGA) sử dụng ngôn ngữ mô tả phần cứng (VHDL).

Luận văn được chia làm 3 chương:

Chương 1 Các vấn đề kỹ thuật

Trình bày tổng quan về lý thuyết đóng gói luồng E1, trong đó tìm hiểu về lý thuyết chung ghép kênh theo thời gian, cấu trúc khung E1 theo chuẩn ITU-T. Trình bày tóm tắt về sự hình thành, phát triển, ứng dụng của công nghệ FPGA. Trình bày tóm tắt về ngôn ngữ lập trình mô tả phần cứng VHDL.

Chương 2 Thiết kế modul đóng khung E1 bằng FPGA trên bảng mạch thực tế

Thực hiện thiết kế board mạch phần cứng sử dụng công cụ thiết kế mạch Altium Designer theo các bước thiết kế sơ đồ nguyên lý, layout, đặt gia công tại nhà máy và hàn dán các linh kiện. Board mạch này thực chất là mạch trung kế luồng E1 trong một thiết bị truyền dẫn, trong đó modul đúng khung E1 được mềm hóa bằng FPGA.

Thực hiện thiết kế modul đóng khung E1 bằng FPGA sử dụng ngôn ngữ mô tả phần cứng VHDL. Sử dụng công cụ ISE tiến hành lập trình và thực hiện mô phỏng trên công cụ ModelSim.

Chương 3 Thực thi và kết quả

Tiến hành tổng hợp thiết kế trên công cụ ISE, tạo ra file thực thi cho FPGA (dạng .bit và dạng .mcs), nạp cho FPGA và xem xét kết quả.

Trong quá trình nghiên cứu, học viên luôn cố gắng bám sát các tài liệu khoa học. Nội dung chi tiết của luận văn sẽ được trình bày dưới đây.

CHƯƠNG 1 – CÁC VẤN ĐỀ KỸ THUẬT

1.1 – Tổng quan đóng gói luồng E1

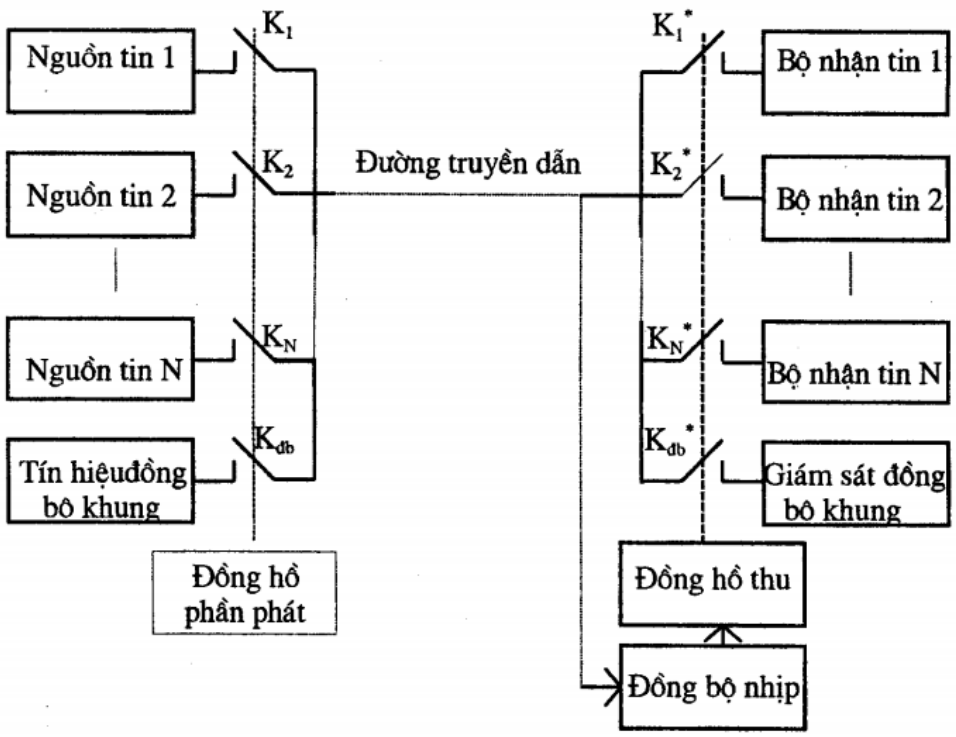
1.1.1 – Nguyên lý ghép kênh theo thời gian

Việc chia sẻ đường truyền dẫn thành nhiều kênh liên lạc cho nhiều nguồn thông tin cùng sử dụng được gọi chung là ghép kênh. Trong kỹ thuật truyền dẫn tín hiệu, có hai phương pháp ghép kênh cơ bản là: a) ghép kênh theo tần số (FDM: *Frequency Division Multiplexing*), trong đó băng tần truyền dẫn của hệ thống được chia thành nhiều băng con hình thành nhiều kênh liên lạc phân biệt với nhau về tần số; b) ghép kênh theo thời gian (TDM: *Time Division Multiplexing*), trong đó thời gian sử dụng đường truyền dẫn được chia thành các phần khác nhau gọi là khe thời gian và việc truyền đưa tin tức từ các nguồn tin khác nhau được thực hiện trong các khe thời gian riêng biệt.

Về nguyên tắc, phương pháp ghép kênh theo thời gian cũng có thể áp dụng cho các tín hiệu *analog*, thí dụ như ghép kênh theo thời gian trong thiết bị P404 của Liên Xô cũ. Tuy nhiên, các tín hiệu *analog* thường được xem là có phổ tương đối hạn chế hay chí ít cũng có thể thực hiện hạn phổ mà không ảnh hưởng lắm đến chất lượng liên lạc. Thêm vào đó, việc chuyển phổ của các tín hiệu *analog* lên băng tần đường dây và sắp xếp chúng phân biệt nhau về dải tần có thể thực hiện một cách khá dễ dàng. Do đó, trong các hệ thống truyền dẫn *analog* việc ghép nhiều kênh liên lạc thường được thực hiện theo phương pháp ghép kênh theo tần số.

Tín hiệu số có một đặc điểm cơ bản là các phần tử tín hiệu (xung tín hiệu) có thời gian tồn tại hữu hạn. Thời gian tồn tại của từng phần tử chỉ phụ thuộc vào độ rộng xung có thể tạo ra và xử lý được, mặc dầu khoảng cách giữa các phần tử kế tiếp nhau là một đại lượng cố định gọi là độ dài khung của tín hiệu (125 μs đối với tín hiệu thoại PCM chẳng hạn). Một khi độ rộng xung tín hiệu khá nhỏ hơn độ dài khung tín hiệu, có thể chia khung tín hiệu thành một số khe thời gian và ghép một số xung tín hiệu từ một số nguồn tin số vào cùng một khung tín hiệu. Tín hiệu từ mỗi một nguồn tin như vậy được truyền đi trên một khe thời gian riêng. Đối với các hệ thống truyền dẫn số, việc ghép kênh theo thời gian như trên có thể thực hiện khá

thuận lợi. Nguyên lý ghép kênh theo thời gian có thể giải thích một cách đơn giản thông qua sơ đồ hình 1.1.



Hình 1.1: Nguyên lý ghép kênh theo thời gian

Dưới tác động của các xung đồng hồ (xung nhịp) các khóa K_1, K_2, …, K_N lần lượt nối trong những khe thời gian xác định các nguồn tin thứ 1, 2, …, N với đường truyền dẫn. Ở phía thu, các khóa K_1^*, K_2^*, …, K_N^* lần lượt nối đường truyền dẫn với các bộ nhận tin thứ 1, 2, …, N một cách tương ứng. Các thiết bị đóng vai trò hệ thống các khóa chuyển mạch ở phần phát và phần thu được gọi một cách tương ứng là bộ phân phối phát và bộ phân phối thu. Chúng là thành phần cốt lõi của các thiết bị ghép kênh (ở phần phát) và phân kênh (ở phần thu). Chu kì làm việc của bộ phân phối phát và phân phối thu chính là độ dài khung của một tín hiệu nhánh và được gọi là một khung. Một khi các bộ phân phối phát và phân phối thu hoạt động đồng bộ với nhau thì việc truyền tin giữa các cặp nguồn tin – bộ nhận tin sẽ diễn ra không lỗi. Việc mất đồng bộ giữa phân phối phát và phân phối thu có thể dẫn đến những sai lạc thông tin rất trầm trọng và vì vậy đồng bộ là chỉ tiêu hàng đầu trong ghép kênh theo thời gian. Để đảm bảo yêu cầu cao về đồng bộ (gióng đúng

thời gian đóng mở các cặp khóa K_i - K_i^* như trên hình 1.1) cần có các thiết bị đồng bộ thực hiện duy trì hoạt động đồng bộ của phân phối phát và phân phối thu, bao gồm cả đồng bộ nhịp và đồng bộ khung.

Đồng bộ khung trong ghép kênh số theo thời gian được theo dõi nhờ việc truyền liên tục tổ hợp đồng bộ khung đặc biệt trong một khe thời gian riêng trong khung tín hiệu. Bộ thu giám sát đồng bộ khung sẽ liên tục theo dõi tổ hợp đồng bộ khung. Việc sai liên tiếp tổ hợp đồng bộ khung này sẽ được hiểu là mất đồng bộ khung. Việc điều khiển đồng bộ trở lại được thực hiện bằng cách trượt khung đi từng khe thời gian cho tới khi tổ hợp đồng bộ khung được thu đúng.

Trong trường hợp mất đồng bộ khung, nói chung tổ hợp được giám sát bởi bộ giám sát đồng bộ khung sẽ bị sai. Tuy nhiên, lỗi truyền dẫn có thể dẫn đến tổ hợp này vẫn đúng, gây nên hiện tượng đồng bộ giả hết sức nguy hiểm. Hiển nhiên, xác suất đồng bộ giả rất thấp và càng nhỏ khi độ dài từ mã đồng bộ khung càng lớn. Tuy vậy, nếu tổ hợp đồng bộ khung quá dài thì việc đồng bộ trở lại sau khi phát hiện thấy mất đồng bộ khung có thể diễn ra càng lâu. Chính vì các lẽ trên mà độ dài của tổ hợp đồng bộ khung cần phải được lựa chọn một cách thích hợp.

Tín hiệu đồng hồ phần thu được đồng bộ theo đồng hồ phần phát. Thông thường, một thiết bị chuyên biệt sẽ thực hiện tách thông tin định thời từ chuỗi tín hiệu tới và điều khiển đồng hồ thu. Quá trình này được gọi là đồng bộ nhịp hay đồng bộ đồng hồ.

Nếu mỗi nguồn tin số nhánh có tốc độ *B* b/s thì tốc độ bit đường dây tổng cộng lớn hơn *NB* b/s một chút, lượng dôi tốc độ này dành cho truyền các thông tin phụ bao gồm thông tin đồng bộ, các tín hiệu báo hiệu và tín hiệu nghiệp vụ…Việc ghép kênh theo thời gian có thể thực hiện ghép theo bit hay theo tổ hợp mã.

1.1.2 – Ghép kênh đồng bộ và ghép kênh cận đồng bộ

Tùy theo cách thức duy trì đồng bộ giữa các bộ phân phối của thiết bị ghép/tách kênh với các nguồn/bộ nhận tin nhánh chúng ta có hai phương thức ghép kênh: (a) ghép kênh đồng bộ và (b) ghép kênh cận đồng bộ (không đồng bộ).

*** Ghép kênh đồng bộ**

Theo phương thức ghép kênh đồng bộ, các nguồn và bộ nhận tin nhánh được duy trì đồng bộ liên tục và tự động với các bộ phân phối của bộ ghép kênh và bộ phân kênh (*muldex: multiplexer - demultiplexer*). Tốc độ dòng bit lối ra của bộ ghép kênh đúng bằng *N* lần tốc độ dòng bit của các nhánh cộng với tốc độ của các thông tin phụ, với *N* là số nhánh được ghép. Điều này có nghĩa là, nếu không kể tới các thông tin phụ thì tần số nhịp của bộ ghép/tách kênh đúng bằng *N* lần tốc độ nhịp của các nhánh được ghép và quan hệ tốc độ này phải được duy trì một cách liên tục và tự động trong suốt quá trình ghép/tách kênh.

Đối với ghép kênh đồng bộ, các bit hoặc các từ mã của các nhánh được sắp khít nhau tạo nên dòng bit đường dây (trừ các khe dành cho đồng bộ khung, thông tin báo hiệu và nghiệp vụ). Hơn thế nữa, vị trí của các bit/từ mã của mỗi nhánh chiếm một vị trí xác định trong dòng bit đường dây, biết trước được ở phía thu. Do vậy, những ưu điểm căn bản của ghép kênh đồng bộ có thể kể đến là:

- Hiệu quả sử dụng đường truyền cao;

- Việc tách rẽ/ghép kênh tại các trạm trung gian có tách/ghép kênh thực hiện được khá dễ dàng.

Phương thức ghép kênh đồng bộ trong truyền dẫn tín hiệu số được thực hiện ở tốc độ sơ cấp (2,048 Mb/s – luồng E1 - từ 32 kênh 64 kb/s với hệ châu Âu bao gồm 30 kênh thoại số PCM, 2 kênh báo hiệu và đồng bộ; 1,544 Mb/s – luồng T1 – từ 24 kênh thoại số 64 kb/s đối với hệ Bắc Mỹ) và ở các tốc độ rất cao (STM-N của hệ thống phân cấp số đồng bộ SDH).

*** Ghép kênh cận đồng bộ**

Phương thức ghép không đồng bộ được thực hiện theo các phân cấp tốc độ số cận đồng bộ (PDH: *Plesiosynchronous Digital Hierarchy*) từ các cấp tốc độ số từ thứ hai trở lên đối với các hệ thống theo hệ châu Âu và hệ Mỹ, và từ tốc độ cấp ba trở lên đối với hệ Nhật Bản.

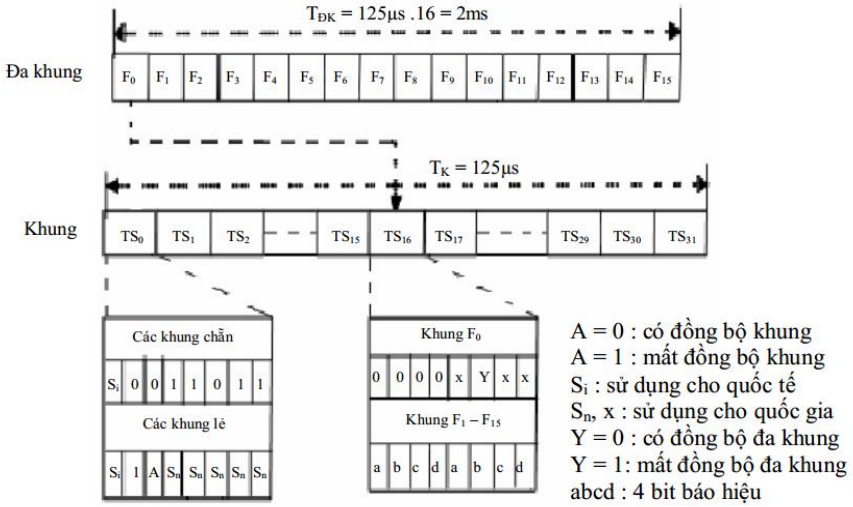
Trong các phương thức ghép không đồng bộ, các bộ phân phối thu và phát của bộ ghép/tách kênh không nhất thiết phải duy trì đồng bộ với các nguồn và bộ nhận tin nhánh. Tốc độ nhịp của bộ ghép kênh không đúng bằng *N* lần tốc độ nhịp

của từng nhánh. Thời điểm bắt đầu của các tin nhánh có thể không cố định trong dòng bit tổng cộng. Việc xác định dòng bit của từng nhánh trong dòng bit tổng cộng do vậy có thể rất khó khăn do sự khác biệt tốc độ nhịp giữa tốc độ nhịp của từng nhánh và *1/N* tốc độ nhịp của bộ ghép kênh và để xác định không nhầm lẫn, khối bit phụ (*header*) phải được thêm vào dòng bit tổng cộng. Điều này làm tăng tốc độ bit truyền dẫn.

Một vấn đề quan trọng đối với ghép kênh cận đồng bộ là việc xử lý chèn do sự không hoàn toàn đồng bộ giữa các nguồn nhánh và bộ phân phối của máy ghép kênh.

1.1.3 – Cấu trúc khung E1 theo tiêu chuẩn ITU-T

Tín hiệu số đầu ra thiết bị PCM-30 được sắp xếp thành khung và đa khung trước khi truyền. Cấu trúc của khung và đa khung như hình 1.2.



Hình 1.2: Cấu trúc khung E1 theo tiêu chuẩn ITU-T

Mỗi kung có thời hạn là 125 µs, được chia thành 32 khe thời gian và đánh số thứ tự từ TS_0 đến TS_{31}. Mỗi TS có thời hạn là 3,9 µs và ghép 8 bit số liệu.

Các khe TS_{16} của khung F_0 truyền từ mã đồng bộ đa khung vào vị trí các bit thứ nhất đến bit thứ tư, bit thứ 6 truyền cảnh báo xa khi mất đồng bộ đa khung (Y = 1), các bit x dùng cho quốc gia (nếu không dùng thì đặt x = 1). Các khe thời gian TS_{16} của khung F_1 đến khung F_{15} dùng để truyền báo hiệu. Báo hiệu của mỗi kênh thoại được mã hóa thành 4 bit a, b, c, d và ghép vào nửa khe thời gian TS_{16}. Nửa

bên trái truyền báo hiệu của các kênh thoại từ 1 đến 15, nửa bên phải truyền báo hiệu của các kênh thoại từ 16 đến 30. Như vậy, phải có 16 khe thời gian TS_{16} trong một đa khung mới đủ để truyền báo hiệu và đồng bộ đa khung. Đó cũng là lý do mỗi đa khung chứa 16 khung. Nếu các bit abcd không dùng cho báo hiệu thì đặt b=1, c=0, d=1. Ngoài ra cần lưu ý không dùng tổ hợp 0000 để truyền báo hiệu vì nó trùng với từ mã đồng bộ đa khung.

Các khe TS_0 của các khung truyền từ mã đồng bộ khung.

Tín hiệu các kênh thoại thứ nhất đến thứ 15 ghép vào các khe thời gian TS_1 đến TS_{15}, tín hiệu các kênh thoại thứ 16 đến 30 ghép vào các khe thời gian TS_{17} đến TS_{31}.

1.2 – Công nghệ FPGA

1.2.1 – Sơ lược về công nghệ FPGA

FPGA (*Field Programmable Gate Array*) là một loại mạch tích hợp cỡ lớn dùng cấu trúc mảng phần tử logic mà người dùng có thể lập trình được. Chữ "*Field*" ở đây muốn nói đến khả năng tái lập trình "bên ngoài" của người sử dụng, không phụ thuộc vào dây chuyền sản suất phức tạp của nhà máy bán dẫn. Vi mạch FPGA được cấu trúc từ các bộ phận sau:

- Các khối logic cơ bản lập trình được (*logic block*)
- Hệ thống mạch liên kết lập trình được
- Khối vào/ra (*IO pads*)
- Phần tử thiết kế sẵn khác như DSP slice, RAM, ROM, nhân vi xử lý…

FPGA cũng được xem như một loại vi mạch bán dẫn chuyên dụng ASIC, nhưng nếu so sánh FPGA với những ASIC đặc chế hoàn toàn hay ASIC thiết kế trên thư viện logic thì FPGA không đạt được mức độ tối ưu như những loại này, và hạn chế trong khả năng thực hiện những tác vụ đặc biệt phức tạp, tuy vậy FPGA ưu việt hơn ở chỗ có thể tái cấu trúc lại khi đang sử dụng, công đoạn thiết kế đơn giản do vậy giảm chi phí, rút ngắn thời gian đưa sản phẩm vào sử dụng.

Còn nếu so sánh với các dạng vi mạch bán dẫn lập trình được dùng cấu trúc mảng phần tử logic như PLA, PAL hay CPLD thì FPGA ưu việt hơn các điểm:

- Tác vụ tái lập trình của FPGA thực hiện đơn giản hơn
- Khả năng lập trình linh động hơn
- Đặc biệt là kiến trúc của FPGA cho phép nó có khả năng chứa khối lượng lớn cổng logic so với các vi mạch bán dẫn lập trình được có trước nó.

Thiết kế hay lập trình cho FPGA được thực hiện chủ yếu bằng các ngôn ngữ mô tả phần cứng HDL như VHDL, Verilog, AHDL, các hãng sản xuất FPGA lớn như Xilinx, Altera thường cung cấp các gói phần mềm và thiết bị phụ trợ cho quá trình thiết kế (chẳng hạn như phần mềm ISE hay Vivado của Xinlinx), cũng có một số các hãng thứ ba cung cấp các gói phần mềm kiểu này như Synopsys, Synplify…Các gói phần mềm này có khả năng thực hiện được tất cả các bước của toàn bộ quy trình thiết kế IC chuẩn với đầu vào là mã thiết kế trên HDL (còn gọi là mã RTL).

*** Lịch sử ra đời FPGA**

FPGA được thiết kế đầu tiên bởi Ross Freeman, người sáng lập công ty Xilinx vào năm 1984, kiến trúc mới của FPGA cho phép tích hợp số lượng tương đối lớn các phần tử bán dẫn vào một vi mạch so với kiến trúc trước đó là CPLD. FPGA có khả năng chứa tới từ 100.000 đến hàng vài tỷ cổng logic, trong khi CPLD chỉ chứa từ 10.000 đến 100.000 cổng logic; con số này đối với PAL, PLA còn thấp hơn nữa chỉ đạt vài nghìn đến 10.000 cổng.

CPLD được cấu trúc từ số lượng nhất định các khối SPLD (*Simple programmable Logic devices*, thuật ngữ chung chỉ PAL, PLA). SPLD thường là một mảng logic AND/OR lập trình được có kích thước xác định và chứa một số lượng hạn chế các phần tử nhớ đồng bộ (*clocked register*). Cấu trúc này hạn chế khả năng thực hiện những hàm phức tạp và thông thường hiệu suất làm việc của vi mạch phụ thuộc vào cấu trúc cụ thể của vi mạch hơn là vào yêu cầu bài toán.

Kiến trúc của FPGA là kiến trúc mảng các khối logic, khối logic, nhỏ hơn nhiều nếu đem so sánh với một khối SPLD, ưu điểm này giúp FPGA có thể chứa nhiều hơn các phần tử logic và phát huy tối đa khả năng lập trình của các phần tử

logic và hệ thống mạch kết nối, để đạt được mục đích này thì kiến trúc của FPGA phức tạp hơn nhiều so với CPLD.

Một điểm khác biệt với CPLD là trong những FPGA hiện đại được tích hợp nhiều những bộ logic số học đã sơ bộ tối ưu hóa, hỗ trợ RAM, ROM, tốc độ cao, hay các bộ nhân cộng (*multiplication and accumulation, MAC*), thuật ngữ tiếng Anh là DSP slice dùng cho những ứng dụng xử lý tín hiệu số DSP.

Ngoài khả năng tái cấu trúc vi mạch toàn cục, một số FPGA hiện đại còn hỗ trợ tái cấu trúc cục bộ, tức là khả năng tái cấu trúc một bộ phận riêng lẻ trong khi vẫn đảm bảo hoạt động bình thường cho các bộ phận khác.

*** Ứng dụng của FPGA**

Ứng dụng của FPGA bao gồm: xử lý tín hiệu số DSP, các hệ thống hàng không, vũ trụ, quốc phòng, tiền thiết kế mẫu ASIC (*ASIC prototyping*), các hệ thống điều khiển trực quan, phân tích nhận dạng ảnh, nhận dạng tiếng nói, mật mã học, mô hình phần cứng máy tính, máy đánh cờ (máy đánh cờ hydra có 32 bộ vi xử lý cộng thêm FPGA đã chiến thắng kiện tướng quốc tế Michael Adams trong năm 2005)…

Do tính linh động cao trong quá trình thiết kế cho phép FPGA giải quyết lớp những bài toán phức tạp mà trước kia chỉ thực hiện nhờ phần mềm máy tính, ngoài ra nhờ mật độ cổng logic lớn FPGA được ứng dụng cho những bài toán đòi hỏi khối lượng tính toán lớn và dùng trong các hệ thống làm việc theo thời gian thực.

*** Kiến trúc của FPGA**

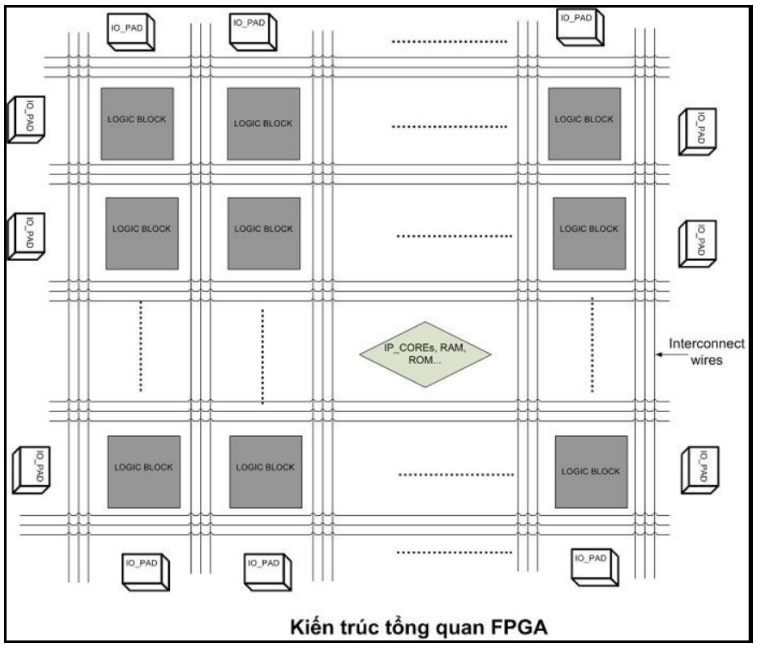
Cấu trúc tổng thể của một FPGA được minh họa ở Hình 1.3.

- Khối logic

Phần tử chính của FPGA là các khối logic (*logic block*). Khối logic được cấu thành từ LUT và một phần tử nhớ đồng bộ flip-flop, LUT (*look up table*) là khối logic có thể thực hiện bất kì hàm logic nào từ 4 đầu vào, kết quả của hàm này tùy vào mục đích mà gửi ra ngoài khối logic trực tiếp hay không qua phần tử nhớ flip-flop.

Trong tài liệu hướng dẫn của các dòng FPGA Xilinx còn sử dụng khái niệm SLICE, một Slice tạo thành từ 2 khối logic, số lượng các Slices thay đổi từ vài nghìn đến vài chục nghìn tùy theo loại FPGA. 4 slices tạo thành một Configurable Logic Blocks (CLBs). CLBs là phần tử cơ bản cấu thành FPGA, là nguồn tài nguyên logic chính tạo nên các mạch logic đồng bộ lẫn không đồng bộ.

Nếu nhìn cấu trúc tổng thể của mảng LUT thì ngoài 4 đầu vào kể trên còn hỗ trợ thêm 2 đầu vào bổ sung từ các khối logic phân bố trước và sau nó nâng tổng số đầu vào của LUT lên 6 chân. Cấu trúc này là nhằm tăng tốc các bộ số học logic.



Hình 1.3: Kiến trúc tổng quan FPGA

- Hệ thống mạch liên kết

Mạng liên kết trong FPGA được cấu thành từ các đường kết nối theo hai phương ngang và đứng, tùy theo từng loại FPGA mà các đường kết nối được chia thành các nhóm khác nhau, ví dụ trong XC4000 của Xilinx có ba loại kết nối: ngắn, dài và rất dài. Các đường kết nối được nối với nhau thông qua các khối chuyển mạch lập trình được (*programmable switch*), trong một khối chuyển mạch chứa một số lượng nút chuyển lập trình được đảm bảo cho các dạng liên kết phức tạp khác nhau.

- Các phần tử tích hợp sẵn

Ngoài các khối logic tùy theo các loại FPGA khác nhau mà có các phần tử tích hợp thêm khác nhau, ví dụ để thiết kế những ứng dụng SoC, trong dòng Virtex 4,5 của Xilinx có chứa nhân xử lý PowerPC, hay trong Atmel FPSLIC tích hợp nhân AVR…, hay cho những ứng dụng xử lý tín hiệu số DSP trong FPGA được tích hợp các DSP Slice là bộ nhân cộng tốc độ cao, thực hiện hàm A*B+C, ví dụ dòng Virtex của Xilinx chứa từ vài chục đến hàng trăm DSP slices với A, B, C 18-bit.

- Block RAM

Ngoài ra những FPGA của Xilinx còn có block RAM, có thể mường tượng như là bộ nhớ nhỏ nằm trong FPGA. Những block RAM này tuy nhỏ (khoảng chục kilobit cho đến vài triệu bit tùy theo loại FPGA) nhưng có thể dùng để tạo một bộ nhớ nhỏ như ROM, FIFO.

1.2.2 – Giải pháp và tổ chức phần mềm đảm bảo của Xilinx

Thiết kế logic lập trình được đã đưa ra kỉ nguyên mà trong đó mật độ của thiết bị ở đơn vị hàng triệu cổng, sự thực hiện của hệ thống ở tốc độ hàng trăm MHz. Xilinx đã đưa ra các công cụ thiết kế hoàn toàn đầy đủ mà nó cho phép thực hiện các thiết kế trong họ PLD của Xilinx. Các giải pháp phát triển kết hợp với các kỹ thuật mạnh tạo ra một sự linh hoạt, mềm dẻo, giao tiếp đồ họa dễ sử dụng giúp người thiết kế có được các thiết kế tốt nhất có thể trong một dự án lớn mà không cần dựa nhiều vào kinh nghiệm. Công cụ phần mềm thiết kế ISE (*Integrated Synthesis Invironment – Môi trường thiết kế tích hợp*) là công cụ thiết kế tổng thể, bao hàm các công cụ phần mềm thiết kế khác nhau và đây cũng là công cụ thiết kế được sử dụng nhiều nhất trong thiết kế các PLD (*Progammable Logic Device*) của Xilinx.

Phần mềm ISE cải thiện đáng kể thời gian đưa một sản phẩm ra thị trường bởi việc tăng tốc quá trình nhập thiết kế. Các bước thực hiện một thiết kế được cung cấp trong phần mềm ISE, ngoài ra chúng còn được hỗ trợ thêm bởi các phần mềm bổ sung khác như ChipScope, PlanAhead, Impact, System Generator, v.v…Để tiện cho việc nắm bắt và phân loại các loại phần mềm ta mô tả phần này theo thứ tự thực hiện một thiết kế.

*** Nhập thiết kế**

Các công cụ hỗ trợ các phương pháp phổ biến nhất ngày nay để tạo ra một thiết kế bao gồm: Nhập thiết kế bằng sơ đồ, bằng ngôn ngữ HDL, bằng việc tích hợp các lõi IP, hỗ trợ mạnh mẽ việc tái sử dụng các lõi IP. Sự đa dạng của việc nhập một thiết kế đã đưa ra một môi trường thiết kế dễ sử dụng nhất và cho phép với tất cả các thiết kế logic. Nó bao gồm các công cụ thiết kế sau: *Schematic Editor, HDL Editor, State Diagram Editor, Core Generator System, Pinout and Area Constraint Editor, Architecture Wizard, Xilinx System Generator for DSP.*

*** Tổng hợp thiết kế**

ISE cải tiến tổng hợp HDL để đưa ra kết quả tối ưu hóa cho việc tổng hợp trên các PLD, đây là một trong các bước cơ bản nhất trong phương pháp thiết kế. Nó lấy các định nghĩa của thiết kế trên HDL và tạo ra sự mô tả vật lý hoặc logic cho thiết bị silicon đích.

Bộ máy tổng hợp tiên tiến đưa ra một kết quả tối ưu hóa cao với một thời gian điều chỉnh và thời gian dịch nhanh. Để phù hợp với yêu cầu này, bộ máy tổng hợp cần phải được tích hợp chặt chẽ với công cụ thực hiện vật lý, hơn nữa sự bỏ qua việc thăm dò giữa thông tin thiết kế vật lý và mã thiết kế HDL đã cải thiện được thời gian biến đổi thiết kế.

Phần mềm ISE đưa ra một sự tích hợp gắn liền với các bộ máy tổng hợp chủ đạo như: *Mentor Graphics Leonardo Spectrum, Exempla, Synopsys và Synplicity Synplify/Pro, ABEL, XST (Xinlinx Synthesis Technology).*

*** Thực thi và nạp cấu hình**

Việc thực hiện thiết kế logic lập trình được là gán các chức năng logic được tạo trong suốt quá trình nhập thiết kế và tổng hợp chúng vào trong tài nguyên vật lý cụ thể. Thuật ngữ "*Sắp đặt và định tuyến*" được sử dụng để mô tả cho quá trình thực hiện cho FPGA còn "*Lắp đặt*" được sử dụng cho CPLD. Thực thi chính là nạp cấu hình cho thiết bị, mà sự thực thi này chính là tạo và tải một luồng các bit được tạo ra từ thông tin sắp đặt và định tuyến vào trong các thiết bị đích PLD. Để thực hiện

phần này có các công cụ hỗ trợ sau: *FloorPlanner, Constraints Editor, Timing Driven Place & Route, Modular Design, Timing Improvent Wizard.*

*** Tích hợp mức Board**

Phần mềm ISE đưa ra sự hỗ trợ mạnh mẽ để giúp người thiết kế đảm bảo thiết kế logic lập trình làm việc trong một hệ thống. Xilinx dự báo trước được các kết quả chính, chẳng hạn như việc sắp đặt một board mạch phức tạp, tích hợp các tín hiệu, giao tiếp bus tốc độ cao, độ rộng dải thông vào ra, các nhiễu điện từ cho người thiết kế mức hệ thống. Để có thể dễ dàng thực hiện các bước này Xilinx đã cung cấp các kỹ thuật chủ đạo cho FPGA:

- XCITE (Trở kháng điều khiển được số)
- DCM (Bộ quản lý đồng hồ số cho thời gian của hệ thống)
- EMI (Bộ quản lý nhiễu điện từ trường)
- Thông tin đóng gói cho sự tích hợp ở mức Board
- Kiểm tra ở mức Board ISE

Nó bao gồm các phần mềm sau: *IBIS Models, STAMP Models, LMG Models, ChipScope ILA.*

*** Các kỹ thuật kiểm tra**

Phần mềm ISE đưa ra các kỹ thuật kiểm tra mà nó hỗ trợ trong tất cả các giai đoạn của thiết kế cho đến khi tích hợp chúng trên Board.

- Kiểm tra tĩnh

Công cụ kiểm tra tĩnh cho phép người thiết kế kiểm tra thiết kế ngoài yêu cầu. Việc kiểm tra có thể thực hiện ở mọi khía cạnh hoặc kiểm tra theo sự chọn lựa, cho phép tìm lỗi trong quá trình thực thi. Công cụ kiểm tra tĩnh cũng đưa ra các khả năng gỡ rối và phân tích mạnh mẽ. Các công cụ kiểm tra tĩnh như: *Constraint Editor, Delay Calculator, Trace, Timinh Analyzer, Prime Time, Xpower, Formality, Conformal LEC, DRC, Chip Viewer.*

- Kiểm tra động

Bao gồm các công cụ sau: *HDL Bencher, ModelSim XE, State Bench, HDL Simulation Libraries.*

- Kiểm tra mức board:

Việc sử dụng công cụ kiểm tra tại mức board nhằm để đảm bảo rằng thiết kế thực hiện đúng theo dự định và chúng được tích hợp với phần còn lại của hệ thống. Các công cụ này bao gồm: *IBIS Models, Tau, BLAST, STAMP Models, Impact.*

1.2.3 – Các bước thực hiện thiết kế trên FPGA

*** Mô tả ban đầu về thiết kế (Specification)**

Quy trình được thể hiện qua hình 1.4. Khi thực hiện một bài toán sử dụng FPGA, thì nó có ý nghĩa cho một ứng dụng riêng biệt. Chính vì xuất phát từ mỗi ứng dụng trong thực tiễn cuộc sống, ta sẽ phải đặt ra yêu cầu thiết kế sao cho IC thực hiện được tối ưu nhất những ứng dụng đó. Bước đầu tiên của quy trình thiết kế này có nhiệm vụ tiếp nhận những yêu cầu của thiết kế và xây dựng nên kiến trúc tổng quát của thiết kế.

- Mô tả thiết kế (Design Specification)

Trong bước này, từ những yêu cầu của thiết kế và dựa trên khả năng của công nghệ hiện có, người thiết kế kiến trúc sẽ xây dựng nên toàn bộ kiến trúc tổng quan cho thiết kế. Nghĩa là trong bước này người thiết kế kiến trúc phải mô tả được những vấn đề sau:

+ Thiết kế có những khối nào

+ Mỗi khối có chức năng gì

+ Hoạt động của toàn bộ thiết kế và của mỗi khối ra sao

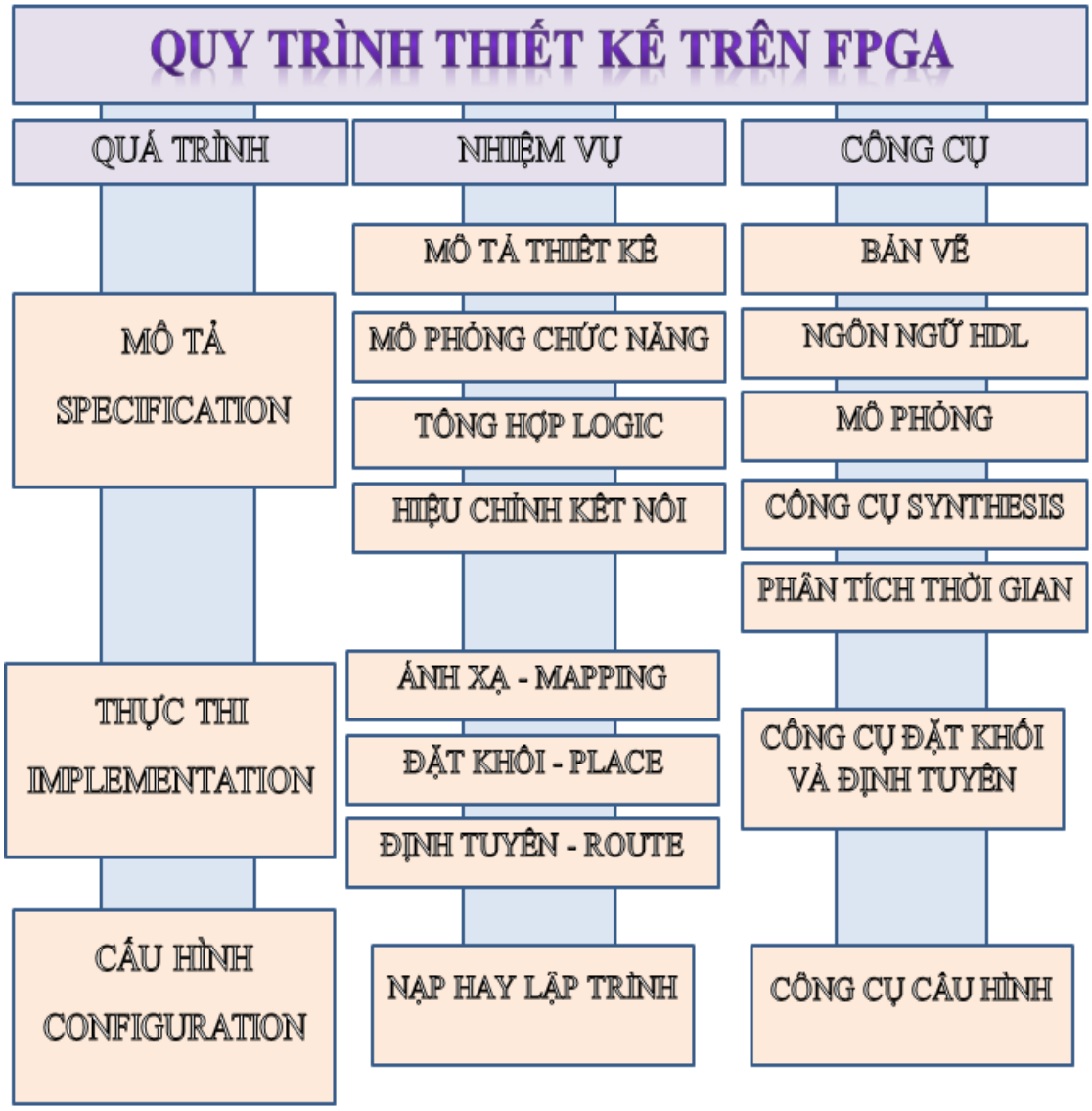
+ Phân tích các kỹ thuật sử dụng trong thiết kế và các công cụ, phần mềm hỗ trợ thiết kế.

Một thiết kế có thể được mô tả sử dụng ngôn ngữ mô tả phần cứng, như VHDL hay Verilog HDL hoặc có thể mô tả qua bản vẽ mạch (schematic capture). Một thiết kế có thể vừa bao gồm bản vẽ mạch mô tả sơ đồ khối chung, vừa có thể dùng ngôn ngữ HDL để mô tả chi tiết cho các khối trong sơ đồ.

- Mô phỏng chức năng (Function Simulation)

Sau khi mô tả thiết kế ta cần mô phỏng tổng thể thiết kế về mặt chức năng để kiểm tra thiết kế có hoạt động đúng với các chức năng yêu cầu hay không.

- Tổng hợp logic (Logic Synthesis)

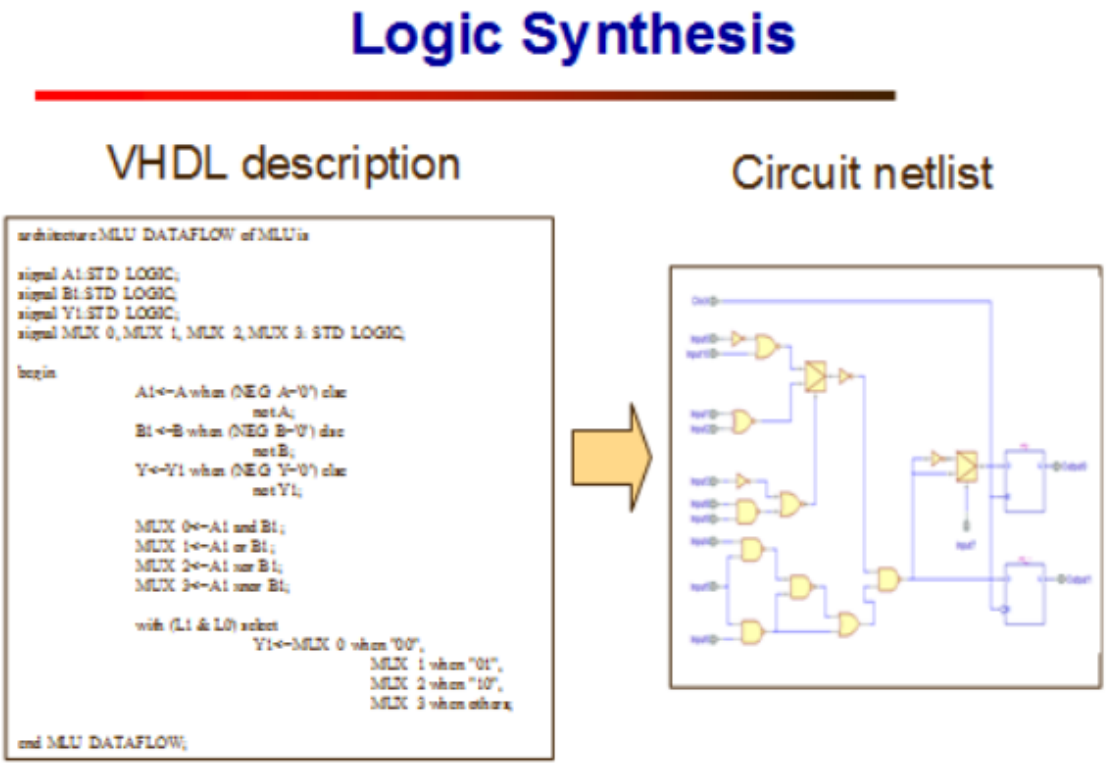


Hình 1.4: Quy trình thiết kế trên FPGA

Tổng hợp logic là quá trình tổng hợp các mô tả thiết kế thành sơ đồ bố trí mạch (netlist). Hình 1.5 thể hiện quá trình này. Quá trình chia thành 2 bước:

+ Chuyển đổi các mã RTL, mã HDL thành mô tả dưới dạng các biểu thức đại số Boolean.

+ Dựa trên các biểu thức này kết hợp với thư viện tế bào chuẩn sẵn có để tổng hợp nên một thiết kế tối ưu.



Hình 1.5: Tổng hợp logic thiết kế

- Hiệu chỉnh các kết nối (Datapath Schematic)

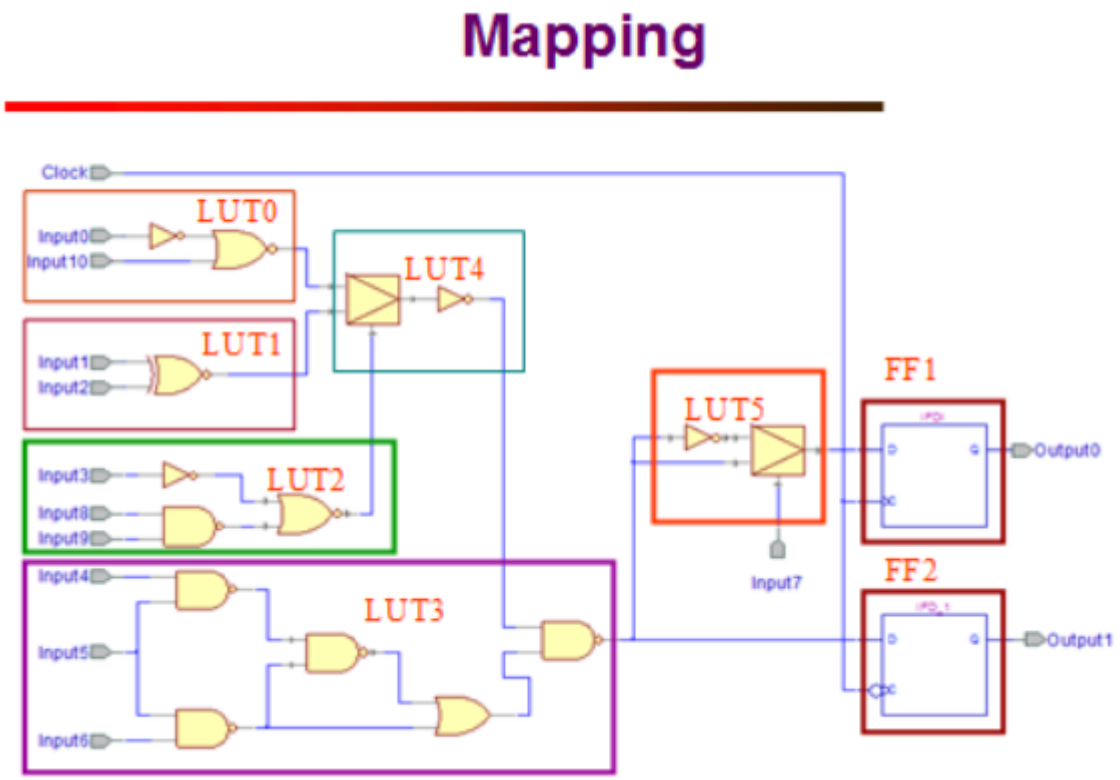
Nhập netlist và các ràng buộc về thời gian vào một công cụ phân tích thời gian (timing analysic). Công cụ phân tích này sẽ tách rời tất cả các kết nối của thiết kế, tính thời gian trễ của các kết nối dựa trên các ràng buộc. Dựa trên kết quả phân tích (report) của công cụ phân tích, xác định các kết nối không thỏa mãn về thời gian. Tùy theo nguyên nhân dẫn đến không thỏa mãn mà ta có thể viết lại mã và tiến hành tổng hợp logic hoặc hiệu chỉnh lại các ràng buộc.

*** Thực thi (Implementation)**

Khi đã có sơ đồ bố trí netlist mô tả tổng thể thiết kế tại mức cổng (chỉ gồm các cổng logic cơ bản và các mạch logic khác như MUX). Quá trình này sẽ đặt sơ đồ netlist lên chip, goi là quá trình thực thi (Device Implementation). Quá trình gồm các bước sau:

- Ánh xạ (Mapping hay còn gọi là Fitting – ăn khớp). Hình 1.6 mô tả bước Mapping. Quá trình này gồm có chuẩn bị dữ liệu đầu vào, xác định kích thước các

khối. Các khối này sẽ phải phù hợp với cấu trúc của một tế bào cơ bản của FPGA và đặt chúng vào các vị trí tối ưu cho việc đi dây.

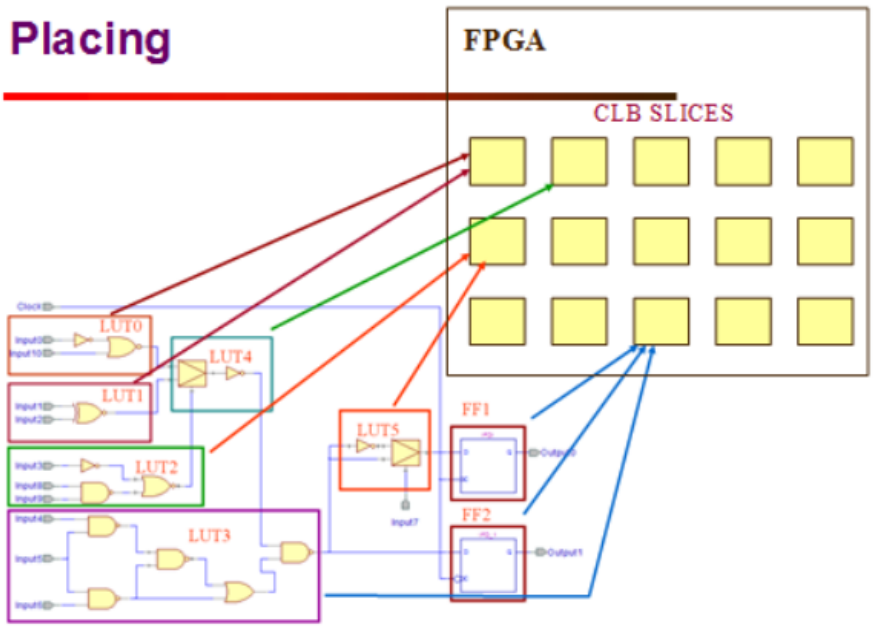


Hình 1.6: Ánh xạ sơ đồ netlist lên FPGA

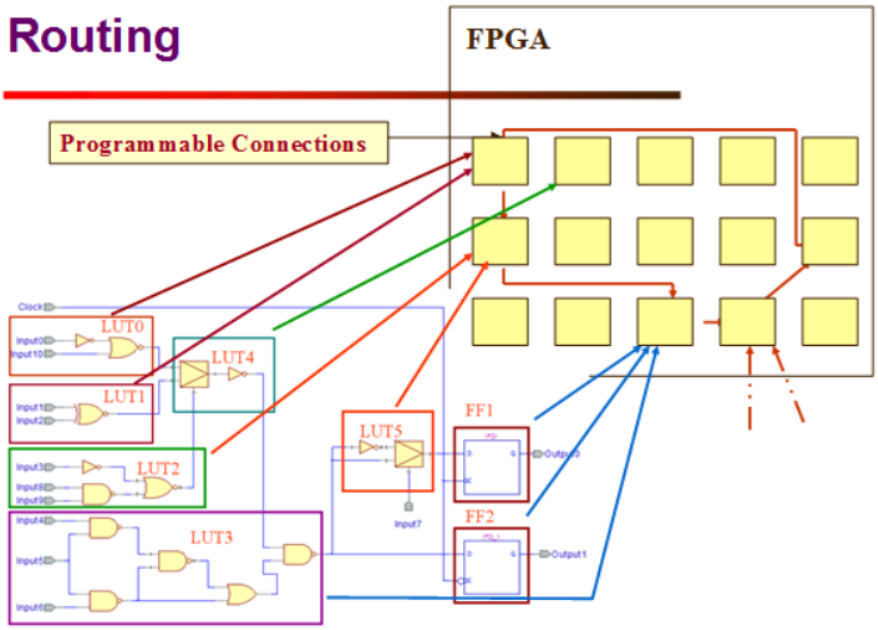
- Đặt khối và định tuyến (Place & Route)

+ Đặt khối: Đặt các khối ánh xạ vào các tế bào (cell) ở vị trí tối ưu cho việc đi dây. Hình 1.7 minh họa cho quá trình này.

+ Định tuyến: Bước này thực hiện việc nối dây các tế bào. Để thực hiện việc này, chúng ta cần có các thông tin sau: Các thông tin vật lý về thư viện tế bào, ví dụ kích thước tế bào, các điểm để kết nối, định thời, các trở ngại trong khi đi dây; một netlist được tổng hợp sẽ chỉ ra các chi tiết instance và mối quan hệ kết nối bao gồm cả các đường dẫn bị hạn chế trong thiết kế; tất cả các yêu cầu của tiến trình cho các lớp kết nối, bao gồm các luật thiết kế cho các lớp đi dây, trở kháng và điện dung, tiêu thụ năng lượng, các luật về sự dẫn điện trong mỗi lớp. Hình 1.8 minh họa cho quá trình này.



Hình 1.7: Đặt khối lên FPGA



Hình 1.8: Định tuyến lên FPGA

*** Quá trình nạp (download) và lập trình (program)**

Sau quá trình thực hiện, thiết kế cần được nạp vào FPGA dưới dạng dòng bit (bit stream). Quá trình nạp thiết kế (download) vào FPGA thường nạp vào bộ nhớ bay hơi, ví dụ như SRAM. Thông tin cấu hình sẽ được nạp vào bộ nhớ. Dòng bit

được truyền lúc này sẽ mang thông tin định nghĩa các khối logic cũng như kết nối của thiết kế. Tuy nhiên, SRAM sẽ mất dữ liệu khi mất nguồn nên thiết kế sẽ không lưu được đến phiên làm việc kế tiếp.

Lập trình (program) là thuật ngữ để mô tả cho quá trình nạp chương trình cho các bộ nhớ không bay hơi, ví dụ như ROM. Khi đó, thông tin cấu hình vẫn sẽ được lưu khi mất nguồn.

1.3 – Ngôn ngữ lập trình VHDL

VHDL là một ngôn ngữ mô tả phần cứng (*Hardware Description Language*), được dùng phổ biến trong việc thiết kế các mạch tích hợp và các hệ thống số một cách dễ dàng và hiệu quả với tốc độ cao.

Ngôn ngữ VHDL dựa trên một ngôn ngữ mô tả phần cứng khác là VHSIC (*Very High Speed Integated Circuit*) do Bộ quốc phòng Mỹ phát triển từ năm 1980. Phiên bản đầu tiên của ngôn ngữ VHDL là VHDL 87. VHDL là một ngôn ngữ mô tả phần cứng đầu tiên được tổ chức IEEE chứng nhận là một tiêu chuẩn chung.

Một trong những điều cơ bản thúc đẩy việc sử dụng ngôn ngữ VHDL đó là tính chuẩn hóa, độc lập về công nghệ của các nhà cung cấp…VHDL được ứng dụng trực tiếp trên các công nghệ FPGA, CPLDs…Code VHDL được viết để thực thi (Implement) các mạch tích hợp trong các thiết bị lập trình được của các hãng như Xilinx, Altera, Amtel…

Một số ưu điểm của ngôn ngữ VHDL có thể kể ra như sau:

- Thứ nhất là tính công cộng: VHDL được phát triển dưới sự bảo trợ của chính phủ Mỹ và hiện nay là một tiêu chuẩn của IEEE. VHDL được sự hỗ trợ bởi nhiều nhà sản xuất thiết bị cũng như nhiều nhà cung cấp công cụ thiết kế mô phỏng hệ thống.

- Thứ hai là khả năng hỗ trợ nhiều công nghệ và phương pháp thiết kế. VHDL cho phép thiết kế bằng nhiều phương pháp ví dụ phương pháp thiết kế từ trên xuống, hay từ dưới lên dựa vào các thư viện sẵn có. VHDL cũng hỗ trợ cho nhiều loại công cụ xây dựng mạch như sử dụng công nghệ đồng bộ hay không đồng bộ, sử dụng ma trận lập trình được hay sử dụng mã ngẫu nhiên.

- Thứ ba là tính độc lập với công nghệ: VHDL hoàn toàn độc lập với công nghệ chế tạo phần cứng. Một mô tả hệ thống dùng VHDL thiết kế ở mức cổng có thể được chuyển thành các bản tổng hợp mạch khác nhau tùy thuộc công nghệ chế tạo phần cứng mới ra đời nó có thể được áp dụng ngay cho các hệ thống đã thiết kế.

- Thứ tư là khả năng mô tả mở rộng: VHDL cho phép mô tả hoạt động của phần cứng từ mức hệ thống số cho đến mức cổng. VHDL có khả năng mô tả hoạt động của hệ thống trên nhiều mức nhưng chỉ sử dụng một cú pháp chặt chẽ thống nhất cho mọi mức. Như thế ta có thể mô phỏng một bản thiết kế bao gồm cả các hệ con được mô tả chi tiết.

- Thứ năm là khả năng trao đổi kết quả: Vì VHDL là một tiêu chuẩn được chấp nhận, nên một mô hình VHDL có thể chạy trên mọi bộ mô tả đáp ứng được tiêu chuẩn VHDL. Các kết quả mô tả hệ thống có thể được trao đổi giữa các nhà thiết kế sử dụng công cụ thiết kế khác nhau nhưng cùng tuân theo tiêu chuẩn VHDL. Cũng như một nhóm thiết kế có thể trao đổi mô tả mức cao của các hệ thống con trong một hệ thống lớn (trong đó các hệ con đó được thiết kế độc lập).

- Thứ sáu là khả năng hỗ trợ thiết kế mức lớn và khả năng sử dụng lại các thiết kế: VHDL được phát triển như một ngôn ngữ lập trình bậc cao, vì vậy nó có thể được sử dụng để thiết kế một hệ thống lớn với sự tham gia của một nhóm nhiều người. Bên trong ngôn ngữ VHDL có nhiều tính năng hỗ trợ việc quản lý, thử nghiệm và chia sẻ thiết kế. Và nó cũng cho phép dùng lại các phần đã có sẵn.

1.3.1 – Các cấu trúc cơ bản của ngôn ngữ lập trình VHDL

Các thành phần chính xây dựng trong ngôn ngữ VHDL được chia thành năm nhóm cơ bản như sau:

- Entity
- Architecture
- Package
- Configuration
- Library

Entity: Trong một hệ thống số, thông thường được thiết kế theo một sự xếp chồng các modul, mà mỗi modul này tương ứng với một thực thể thiết kế (được gọi là *entity*) trong VHDL. Mỗi một *entity* gồm 2 phần:

- Khai báo thực thể (**Entity**)
- Thân kiến trúc (**Architecture Bodies**)

Một khai báo *Entity* được dùng để miêu tả giao tiếp bên ngoài của một phần tử (**Component**), nó bao gồm các khai báo các cổng đầu vào, các cổng đầu ra của phần tử đó. Phần thân của kiến trúc được dùng để mô tả sự thực hiện bên trong của thực thể đó.

Packages: Các đóng gói chỉ ra thông tin dùng chung, mà các thông tin này được sử dụng bởi một vài **Entity** nào đó.

Configuration: Định cấu hình, nó cho phép gắn kết các thể hiện của phần tử nào đó cần dùng của một thiết kế nào đó có dạng một cấu trúc và đưa các thể hiện này vào trong cặp **Entity** và **Architecture**.

Nó cho phép người thiết kế có thể thử nghiệm để thay đổi các sự thực thi khác nhau trong một thiết kế. Mỗi một thiết kế dạng VHDL bao gồm một vài đơn vị thư viện, mà một trong các thư viện này được dịch sẵn và cất trong một thư viện thiết kế.

1.3.2 – Các đối tượng dữ liệu

Một đối tượng dữ liệu giữ một giá trị của một kiểu nhất định. Trong VHDL có 3 lớp đối tượng dữ liệu:

- Các hằng (*constants*)
- Các biến (*variables*)
- Các tín hiệu (*signals*)

Lớp của một đối tượng được chỉ ra bởi một từ khóa và nó được chỉ ra ở điểm bắt đầu của một khai báo.

*** Các hằng**

Một hằng là một đối tượng mà nó được khởi tạo để chỉ ra một giá trị cố định và không bị thay đổi. Khai báo hằng được phép khai báo trong các đóng gói, các

entity, các kiến trúc, các chương trình con, các khối, và trong phát biểu của các quá trình *processes*.

*** Các biến**

Các biến được dùng để lưu dữ liệu tạm thời, chúng chỉ được phép khai báo trong phát biểu *process* hoặc các chương trình con.

*** Các kiểu tín hiệu**

Tín hiệu được dùng để kết nối các *entity* của thiết kế lại với nhau và trao đổi các giá trị biến đổi ở trong phát biểu *process*. Chúng có thể được xem như các dây dẫn hay các bus nối ở trong các mạch thực tế. Tín hiệu có thể được khai báo trong các đóng gói (*package*), trong các khai báo *entity*, trong khai báo kiến trúc (*architecture*), trong các khối (*block*). Với các tín hiệu được khai báo trong các *package* thì tín hiệu này được gọi là tín hiệu toàn cục (các thiết kế có thể sử dụng chúng), các tín hiệu được khai báo trong *entity* là tín hiệu toàn cục trong một *entity*, tương tự với tín hiệu được khai báo trong một kiến trúc, nó là tín hiệu dùng chung trong một kiến trúc đó.

1.3.3 – Các kiểu dữ liệu

Tất cả các đối tượng dữ liệu trong VHDL cần phải được định nghĩa với một kiểu dữ liệu. Một khai báo kiểu phải chỉ ra tên và dải của kiểu đó. Khai báo kiểu dữ liệu được phép đặt trong phần khai báo các đóng gói, trong phần khai báo *entity*, trong phần khai báo kiến trúc, trong phần khai báo các chương trình con và trong phần khai báo các *process*. Các kiểu dữ liệu bao gồm:

- Kiểu liệt kê
- Kiểu nguyên
- Các kiểu dữ liệu tiền định nghĩa
- Kiểu mảng
- Kiểu bản ghi
- Kiểu dữ liệu chuẩn logic
- Kiểu dữ liệu có dấu và không dấu
- Các kiểu phụ

1.3.4 – Các toán tử

VHDL cung cấp 6 lớp toán tử, mỗi một toán tử có một mức ưu tiên nhất định. Tất cả các toán tử trong cùng một lớp thì có cùng một mức ưu tiên.

*** Các toán tử logical**

Kiểu toán tử logic không chấp nhận các toán hạng là các kiểu tiền định nghĩa như kiểu BIT, BOOLEAN và các kiểu mảng các bit, các toán hạng cần phải là cùng kiểu và cùng độ dài.

*** Các toán tử quan hệ**

Các toán tử quan hệ cho ta kết quả có kiểu BOOLEAN, các toán hạng cần phải có cùng kiểu và cùng độ dài.

*** Các toán tử cộng**

Các toán tử cộng bao gồm "+", "-", và "&", trong đó toán tử "&" là toán tử kết nối chuỗi và các đối tượng là mảng các thanh ghi. Với số có dấu và không dấu có thể được dùng với các số nguyên và các kiểu BIT_VECTOR.

1.3.5 – Các kiểu toán hạng

Trong một biểu thức các toán tử sử dụng các toán hạng để tính toán các giá trị của chúng. Các toán hạng trong một biểu thức bao gồm:

- Kiểu chữ
- Kiểu định danh
- Các tên được đánh theo chữ số
- Tên các Slice
- Tên các đặc tính
- Các biểu thức điều kiện
- Các lời gọi hàm
- Các biểu thức chuyển đổi

1.3.6 – Các phát biểu tuần tự

Phát biểu tuần tự chỉ ra sự thực hiện từng bước của một quá trình. Chúng thực hiện từ câu lệnh đầu tiên, câu lệnh thứ hai, …câu lệnh cuối cùng. Các phát biểu

nằm trong một phát biểu quá trình (phát biểu *process*) được gọi là phát biểu tuần tự. Các phát biểu sau đây là các phát biểu tuần từ được định nghĩa trong VHDL:

- Các phát biểu gán biến *variable*
- Các phát biểu gán tín hiệu *signal*
- Các phát biểu *if*
- Các phát biểu *case*
- Các phát biểu *null*
- Các phát biểu xác nhận *assertion*
- Các phát biểu vòng lặp *loop*
- Các phát biểu *next, exit*
- Các phát biểu *while*
- Các phát biểu *procedure*
- Các phát biểu *return*

1.3.7 – Các phát biểu đồng thời

Các phát biểu đồng thời được thực hiện song song trong cùng thời điểm mô phỏng, chúng không thực hiện theo thứ tự mà chúng được viết ra trong một kiến trúc. Các phát biểu đồng thời chuyển thông tin thông qua các đường tín hiệu.

Dưới đây là các phát biểu đồng thời được định nghĩa trong VHDL:

- Các phát biểu gán của một quá trình (*process*)
- Các phát biểu gán tín hiệu đồng thời
- Các phát biểu gán tín hiệu điều kiện
- Các phát biểu gán tín hiệu được chọn lựa
- Các phát biểu *block*
- Các lời gọi thủ tục đồng thời
- Các phát biểu xác nhận đồng thời

1.3.8 – Các đóng gói

Có thể đóng gói để cất các chương trình con, các kiểu dữ liệu, các hằng … thường dùng để sử dụng chúng trong các thiết kế khác. Một *package* bao gồm 2 phần chính: Phần khai báo và phần thân package.

1.3.9 – Mô hình cấu trúc

Thông thường một hệ thống số được mô tả theo tập hợp các thứ bậc của các thành phần. Mỗi thành phần bao gồm một tập các cổng để có thể giao tiếp được với các thành phần khác. Khi mô tả một thiết kế trong VHDL và một thiết kế có thứ bậc chính là một thiết kế đưa ra các khai báo của các thành phần và các phát biểu thể hiện thành phần đó.

Một đơn vị cơ sở để diễn tả hành vi hoạt động chính là các phát biểu *process*, còn đơn vị cơ sở để diễn tả theo kiểu cấu trúc chính là các phát biểu thể hiện của các đơn vị thành phần. Cả 2 loại này đều có thể có mặt trong một thân của một kiến trúc (*architecture*).

1.4 – Tổng kết chương 1

Chương 1 đã nêu ra các lý thuyết tổng quát liên quan đến đóng gói luồng E1 như nguyên lý ghép kênh theo thời gian, ghép kênh đồng bộ và ghép kênh cận đồng bộ, cấu trúc khung E1 theo tiêu chuẩn ITU-T; đã giới thiệu sơ lược về công nghệ FPGA, các giải pháp và tổ chức phần mềm đảm bảo của Xilinx, các bước thực hiện thiết kế trên FPGA; đồng thời khái quát về ngôn ngữ lập trình VHDL như các cấu trúc cơ bản, các đối tượng dữ liệu, các kiểu dữ liệu, các toán tử, các kiểu toán hạng, các phát biểu tuần tự, các phát biểu đồng thời, các đóng gói, mô hình cấu trúc v.v…

Như vậy trong chương 1 đã khái quát lên được các vấn đề kỹ thuật liên quan đến thiết kế, làm cơ sở lý thuyết cũng như làm công cụ tham chiếu để thiết kế được thực hiện chính xác và thành công.

CHƯƠNG 2 – THIẾT KẾ MODUL ĐÓNG KHUNG E1 BẰNG FPGA TRÊN BẢNG MẠCH THỰC TẾ

2.1 – IC spartan xc3s500E

2.1.1 – Họ IC Spartan-3E

Họ sản phẩm Spartan-3E của FPGA được thiết kế đặc biệt để đáp ứng nhu cầu của các ứng dụng điện tử tiêu dùng nhạy cảm với chi phí cao. Dòng IC gồm 5 thành viên cung cấp mật độ từ 100.000 đến 1.6 triệu cổng hệ thống, như trong bảng 2.1.

Bảng 2.1: Họ sản phẩm FPGA Spartan-3E của Xilinx

Device	System Gates	Equivalent Logic Cells	CLB Array (One CLB = Four Slices)				Distributed RAM bits(1)	Block RAM bits(1)	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
			Rows	Columns	Total CLBs	Total Slices						
XC3S100E	100K	2,160	22	16	240	960	15K	72K	4	2	108	40
XC3S250E	250K	5,508	34	26	612	2,448	38K	216K	12	4	172	68
XC3S500E	500K	10,476	46	34	1,164	4,656	73K	360K	20	4	232	92
XC3S1200E	1200K	19,512	60	46	2,168	8,672	136K	504K	28	8	304	124
XC3S1600E	1600K	33,192	76	58	3,688	14,752	231K	648K	36	8	376	156

Họ Spartan-3E xây dựng dựa trên thành công của họ Spartan-3 trước đó bằng cách tăng lượng cổng logic trên mỗi I/O, giảm đáng kể chi phí cho mỗi phần tử logic. Các tính năng mới cải thiện hiệu năng hệ thống và giảm chi phí cấu hình. Những cải tiến này, kết hợp với công nghệ xử lý 90nm tiên tiến, cung cấp nhiều chức năng và băng thông hơn mức có thể trước đây, thiết lập các tiêu chuẩn mới trong công nghiệp logic lập trình. Bởi vì Spartan-3E có chi phí đặc biệt thấp cho một loạt các ứng dụng điện tử tiêu dùng, bao gồm truy cập băng thông rộng, mạng gia đình, màn hình / trình chiếu và thiết bị truyền hình kỹ thuật số. Họ Spartan-3E là một sự vượt trội so với ASIC được lập trình. FPGA tránh chi phí ban đầu cao, các chu kì phát triển dài và tính không linh hoạt vốn có của các ASIC thông thường. Ngoài ra khả năng lập trình của các FPGA cho phép nâng cấp thiết kế trong lĩnh vực mà không cần thay đổi thiết kế phần cứng, điều không thể áp dụng với ASIC.

2.1.2 – Tính năng cơ bản

- Chi phí rất thấp, giải pháp logic hiệu suất cao cho các ứng dụng hướng tới người tiêu dùng.

- Ứng dụng công nghệ 90nm.

- Giao diện đa điện áp, đa tiêu chuẩn chân I/O:

+ Lên đến 376 chân I/O hoặc 156 cặp tín hiệu vi sai.

+ Các tiêu chuẩn tín hiệu kết thúc đơn như LVCMOS, LVTTL, HSTL và SSTL.

+ Có các tín hiệu 3.3V, 2.5V, 1.8V, 1.5V và 1.2V.

+ Tốc độ truyền dữ liệu 622+ Mb/s trên I/O.

+ Hỗ trợ tốc độ dữ liệu kép (DDR) nâng cao.

+ Hỗ trợ DDR SDRAM lên đến 333Mb/s.

- Nguồn tài nguyên logic dồi dào, linh hoạt:

+ Mật độ lên tới 33,192 tế bào logic, bao gồm thanh ghi dịch hoặc hỗ trợ RAM phân tán.

+ Hiệu quả ghép kênh rộng, logic rộng.

+ Các bộ nhân 18x18 cải tiến với đường ống tùy chọn.

+ Cổng nạp chương tình và gỡ lỗi theo chuẩn IEEE 1149.1/1532.

- Kiến trúc bộ nhớ SelectRAM phân cấp:

+ Khối RAM nhanh lên đến 648 Kbits

+ RAM phân tán lên đến 231 Kbits

- Có tới 8 trình quản lý đồng hồ kỹ thuật số (DCMs):

+ Loại bỏ lệch đồng hồ (trì hoãn khóa vòng)

+ Tổng hợp tần số, bộ nhân, bộ chia

+ Dịch pha độ phân giải cao

+ Dải tần rộng (5MHz đến trên 300MHz)

- Tám đồng hồ toàn bộ cộng với 8 đồng hồ bổ sung cho mỗi nửa thiết bị, cộng với định tuyến độ lệch thấp phong phú.

- Giao diện cấu hình cho các PROM tiêu chuẩn công nghiệp

+ Flash PROM chuẩn nối tiếp SPI chi phí thấp, tiết kiệm không gian

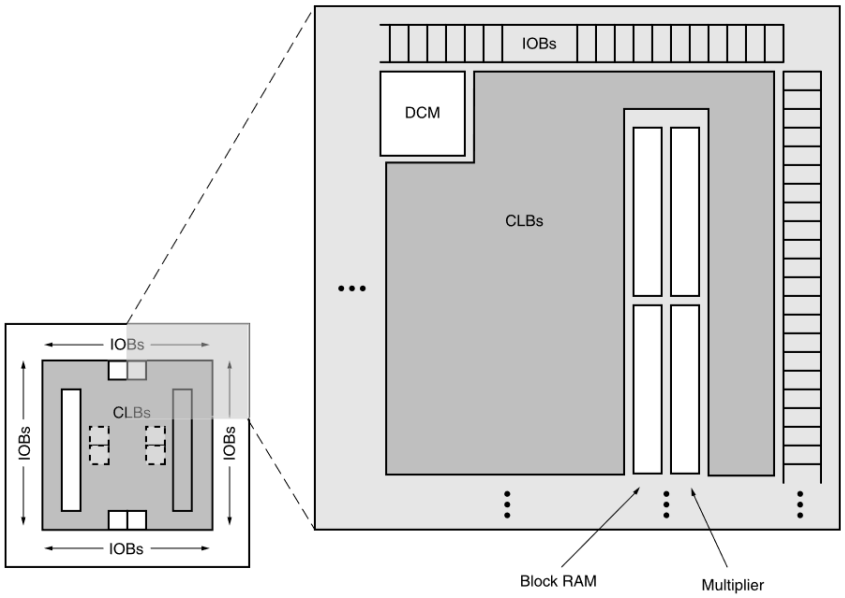
+ Flash PROM chuẩn song song tốc độ x8 hoặc x8/x16

+ Nền tảng cấu hình JTAG chi phí thấp

- Phần mềm Xilinx ISE và WebPACK hoàn chỉnh

- Lõi xử lý nhúng PicoBlaze và MicroBlaze
- Có các gói QFP và BGA chi phí thấp

2.1.3 – Kiến trúc tổng quan



Hình 2.1: Kiến trúc tổng quan của IC xc3s500E

Kiến trúc IC xc3s500E nói riêng và họ spartan-3E nói chung bao gồm 5 yếu tố chức năng lập trình cơ bản:

- Các khối logic có khả năng định cấu hình (CLB) chứa các bảng tra cứu linh hoạt (LUT) thực hiện logic cộng với các thành phần lưu trữ được sử dụng làm flip-flop hoặc chốt. CLB thực hiện rất nhiều chức năng logic cũng như lưu trữ dữ liệu.

- Các khối vào/ra (IOB) kiểm soát luồng dữ liệu giữa các chân I/O và logic bên trong của thiết bị.Mỗi IOB hỗ trợ luồng dữ liệu hai chiều cộng với hoạt động 3 trạng thái. Hỗ trợ nhiều tiêu chuẩn tín hiệu, bao gồm bốn tiêu chuẩn khác biệt hiệu suất cao. Các thanh ghi dữ liệu tốc độ kép (DDR) cũng được bao gồm trong đó.

- Block RAM hỗ trợ lưu trữ dữ liệu dưới dạng các khối cổng kép 18 Kbits.

- Các khối nhân cho phép 2 số nhị phân 18 bit làm đầu vào và tính toán kết quả.

- Khối quản lý đồng hồ kỹ thuật số (DCM) cung cấp các giải pháp tự hiệu chỉnh, hoàn toàn kỹ thuật số để phân phối, trì hoãn, nhân, chia và tín hiệu đồng hồ chuyển pha.

Các phần tử này được tổ chức như trong hình 2.1. Một vòng các IOB bao quanh một dãy các CLB thông thường. Mỗi thiết bị có 2 cột Block RAM. Mỗi cột RAM bao gồm một số khối RAM 18 Kbit. Mỗi Block RAM được liên kết với một số nhân chuyên dụng. Các DCM được định vị ở trung tâm với 2 ở phía trên và 2 ở phía dưới của cùng thiết bị.

IC có một mạng lưới phong phú kết nối tất cả 5 thành phần chức năng, truyền tín hiệu giữa chúng. Mỗi thành phần chức năng có một ma trận chuyển đổi liên quan cho phép nhiều kết nối đến định tuyến.

2.2 – Thiết kế phần cứng, phần mềm

2.2.1 – Thiết kế phần cứng

Phần này tiến hành thiết kế một board mạch (thực chất là card trung kế E1 trong một thiết bị tương đương tổng đài quân sự) trong đó có sử dụng IC xc3s500E và thực hiện modul đóng gói E1 trong IC đó phục vụ cho chức năng trung kế E1. Phần cứng này được thiết kế dựa trên phần mềm Altium, một phần mềm thiết kế mạch rất phổ biến hiện nay.

*** Sơ đồ nguyên lý**

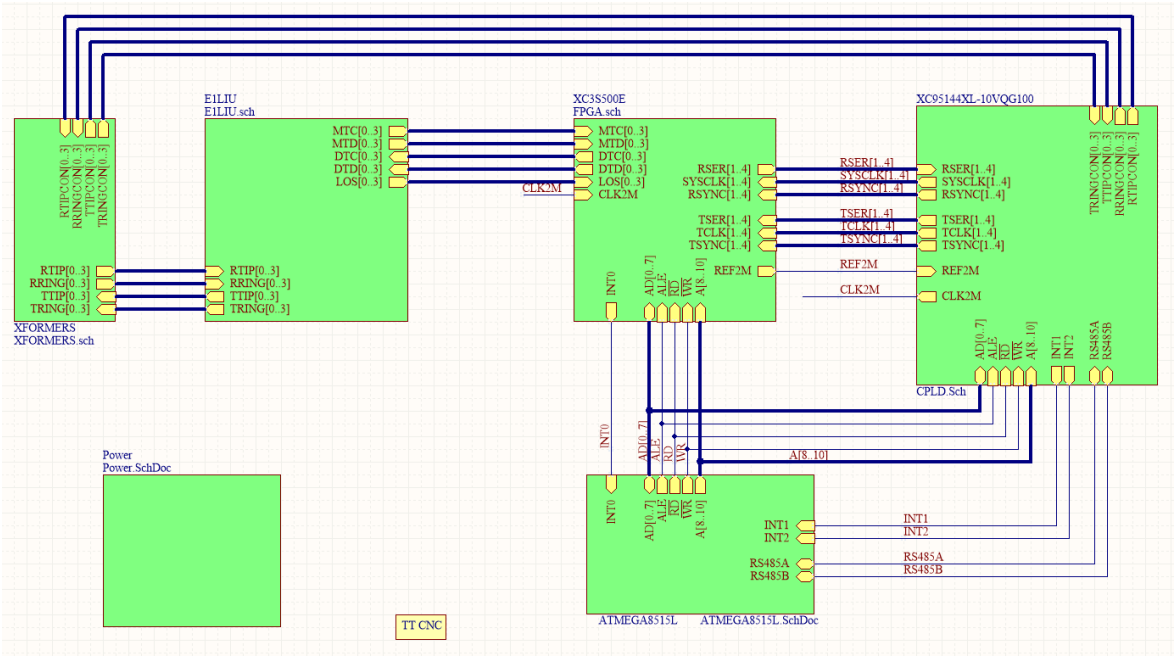
- Sơ đồ kết nối tổng quát:

Trong đó khối Power là khối cấp nguồn cho các IC, linh kiện trong mạch. Khối cấp 3 loại nguồn cơ bản là 3.3VDC, 2.5VDC, 1.2VDC từ nguồn đầu vào 5VDC.

Khối ATMEGA8515L là khối vi điều khiển thực hiện chức năng điều khiển chung toàn bộ card mạch và kết nối card mạch với các card mạch khác và với toàn bộ thiết bị.

Khối E1LIU là khối có chức năng chuyển đổi tín hiệu dạng bit sang tín hiệu mã đường dây HDB3 để chuyển tải sang thiết bị truyền dẫn và ngược lại chuyển đổi từ tín hiệu HDB3 nhận được thành luồng bit để tiến hành xử lý. Ngoài ra còn các chức năng khôi phục tín hiệu định thời clock, xử lý rung pha, v.v…

Khối XFORMERS là khối biến áp có chức năng phối hợp trở kháng giữa tín hiệu HDB3 đầu ra sau khối E1LIU và đường dây.



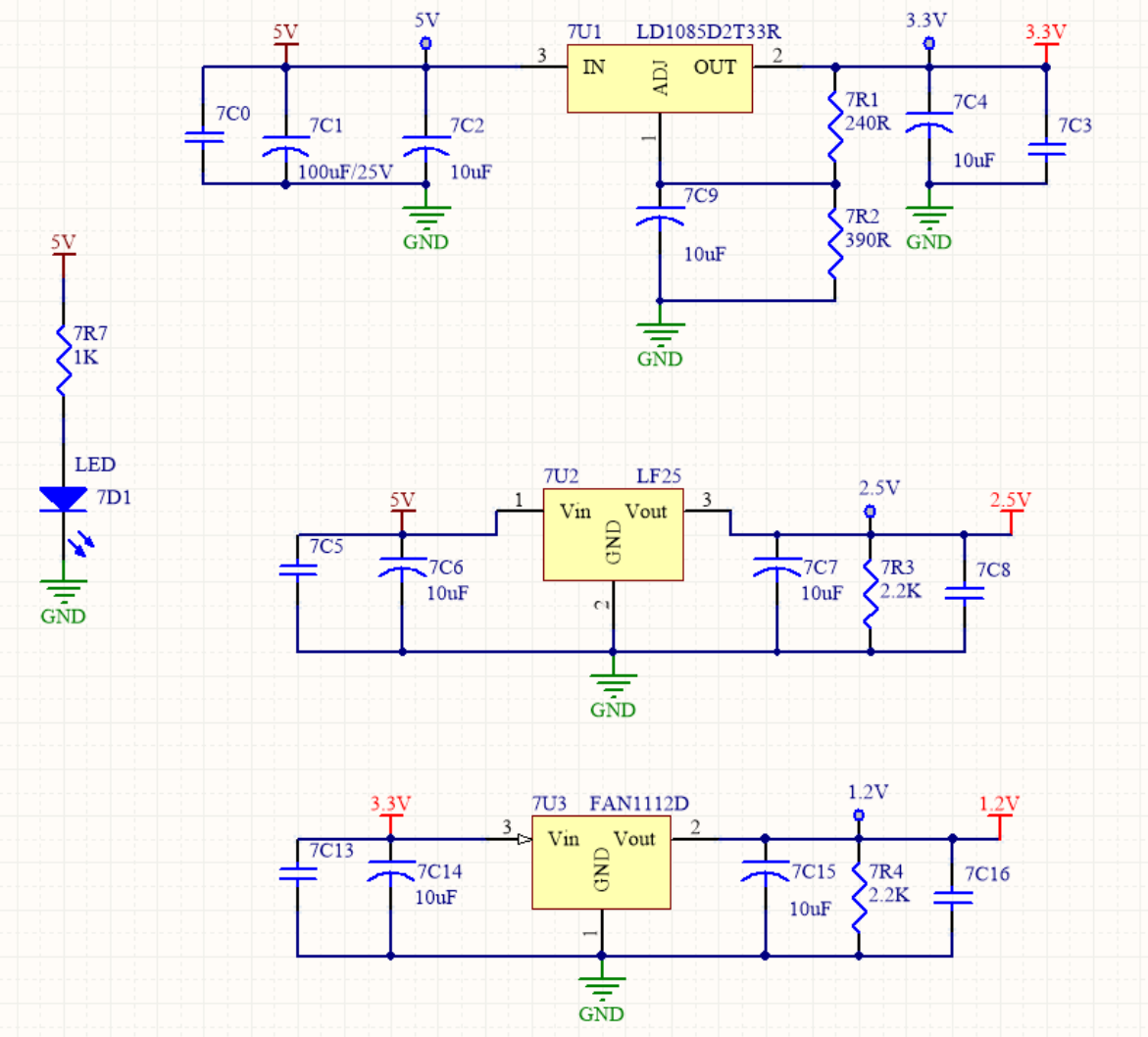
Hình 2.2: Sơ đồ kết nối tổng quát trong card mạch

Khối FPGA là khối chính mà ta đang phát triển, trong đó thực hiện xử lí tín hiệu số, đóng gói E1 từ luồng dữ liệu đầu vào để đưa sang khối E1LIU chuyển tải lên đường dây. Trong khối này có hạt nhân chính là IC xc3s500E.

Khối CPLD là khối thực hiện các chức năng giao tiếp giữa vi xử lý và FPGA, đồng thời tạo ra các xung đồng bộ cho FPGA hoạt động.

- Sơ đồ nguyên lý khối cấp nguồn: Hình 2.3 mô tả sơ đồ nguyên lý khối cấp nguồn.

Đầu vào của khối là nguồn 5VDC. Trong khối sử dụng IC LD1805D2T33R để tạo nguồn đầu ra 3.3VDC, IC LF25 để tạo nguồn đầu ra 2.5VDC, IC FAN1112D để tạo nguồn đầu ra 1.2VDC. Ngoài ra có các điện trở và tụ điện để thực hiện các chức năng lọc, ghép nối và các LED cảnh báo.

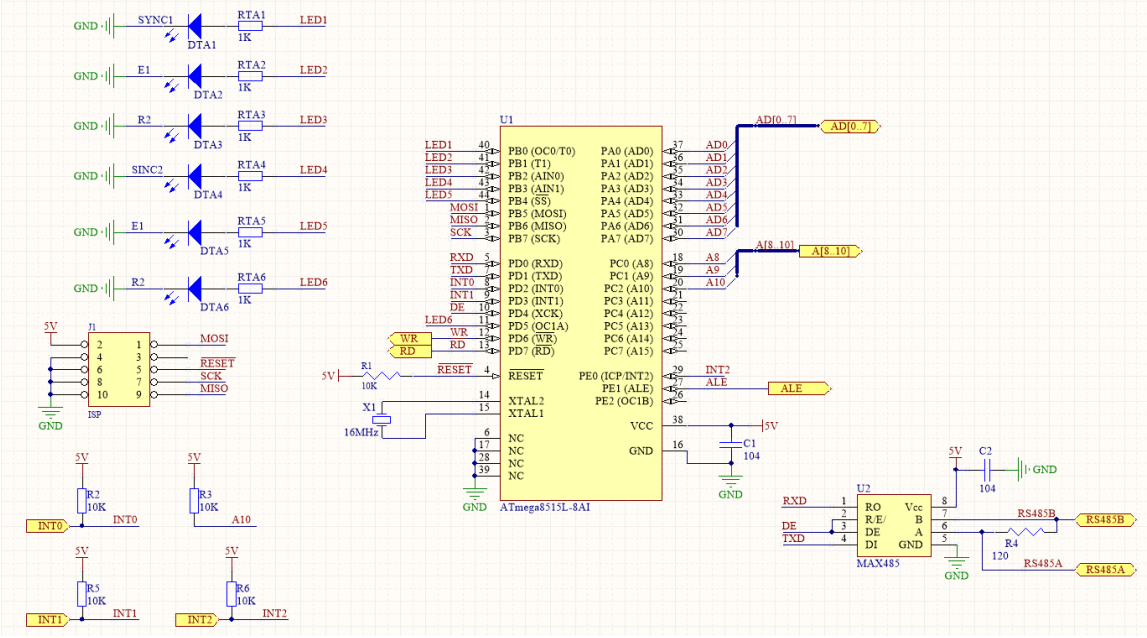


Hình 2.3: Sơ đồ nguyên lý khối cấp nguồn

- Sơ đồ nguyên lý khối điều khiển

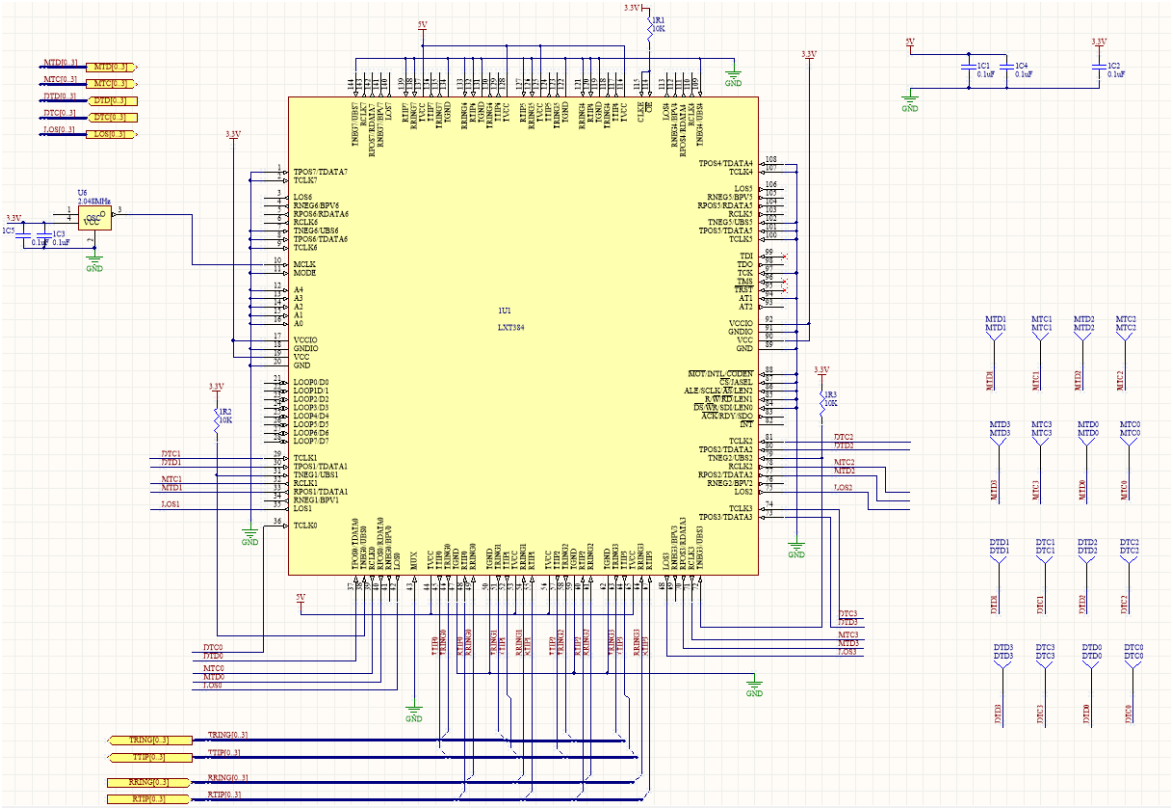
Hình 2.4 mô tả sơ đồ nguyên lý khối điều khiển sử dụng vi điều khiển ATMEGA8515L.

Trong khối có sử dụng IC vi điều khiển 8515L để thực hiện điều khiển hoạt động của card mạch đồng thời kết nối với các card mạch khác và toàn bộ thiết bị tổng đài. IC MAX485 cung cấp cổng console phục vụ cho việc kiểm tra và bug lỗi.



Hình 2.4: Sơ đồ nguyên lý khối điều khiển

- Sơ đồ nguyên lý khối E1LIU

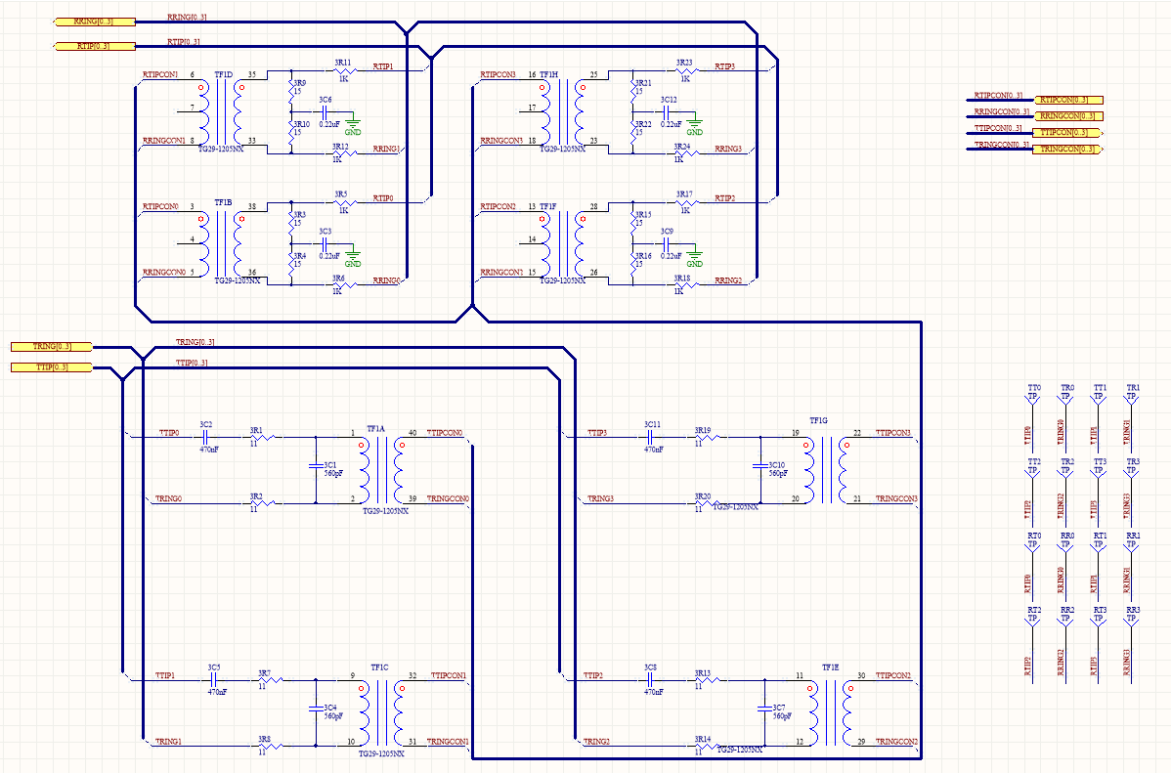


Hình 2.5: Sơ đồ nguyên lý khối E1LIU

Hình 2.5 mô tả sơ đồ nguyên lý khối E1LIU. Trong khối có IC LXT384 là IC giao tiếp luồng. Chức năng của nó là biến luồng tín hiệu số dạng bit 0,1 sau khi đã

đóng thành khung E1 thành luồng tín hiệu mã đường dây HDB3 để có thể truyền tải trên đường dây (hướng phát) và ngược lại chuyển đổi tín hiệu dạng HDB3 thu được thành dạng bit để xử lý (hướng thu). Đồng thời thực hiện tách ghép và khôi phục nguồn tín hiệu định thời clock từ luồng dữ liệu đầu vào, xử lý rung pha, v.v…

- Sơ đồ nguyên lý khối giao diện luồng: Hình 2.6 mô tả sơ đồ nguyên lý khối giao diện luồng.



Hình 2.6: Sơ đồ nguyên lý khối giao diện luồng

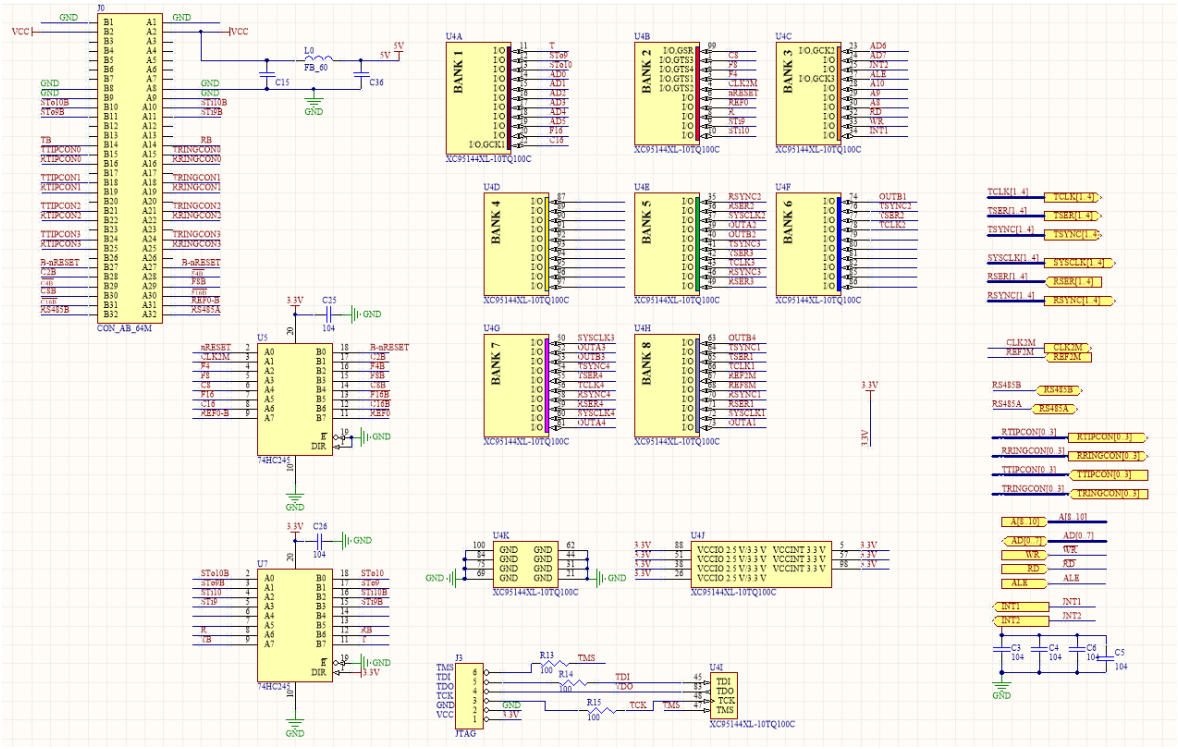
Đây là khối biến áp thực hiện phối hợp trở kháng và mức tín hiệu IC luồng ra đường dây và ngược lại. Mỗi một IC luồng lại tương thích với một biến áp riêng tùy theo đặc tính của nó. Đối với IC luồng LXT384 thì ta sử dụng biến áp TG29. Ngoài ra là các bộ lọc phụ trợ và các tụ điện ghép tầng.

- Sơ đồ nguyên lý khối tạo các tín hiệu định thời và giao tiếp với CPU

Hình 2.7 mô tả sơ đồ nguyên lý khối tạo các tín hiệu định thời và giao tiếp với CPU (sử dụng CPLD).

Trong khối sử dụng IC XC95144XL, là một CPLD có năng lực vừa phải, vừa đủ để thực hiện một số chức năng giao tiếp, tạo các xung đồng bộ, các xung khe

phục vụ cho FPGA nhằm giảm tải cho FPGA (thực ra khối này có thể thiết kế ngay trong FPGA).

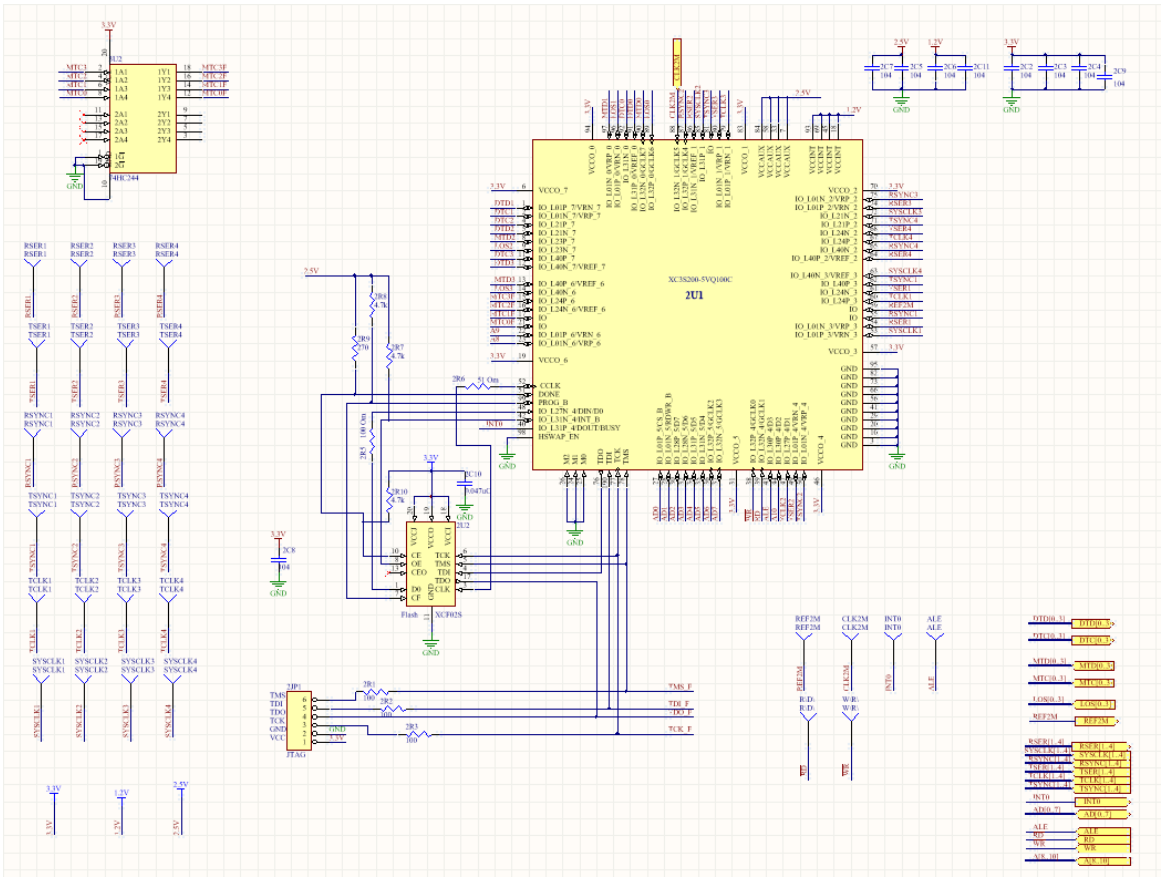


Hình 2.7: Sơ đồ nguyên lý khối tạo các tín hiệu định thời và giao tiếp với CPU

- Sơ đồ nguyên lý khối FPGA

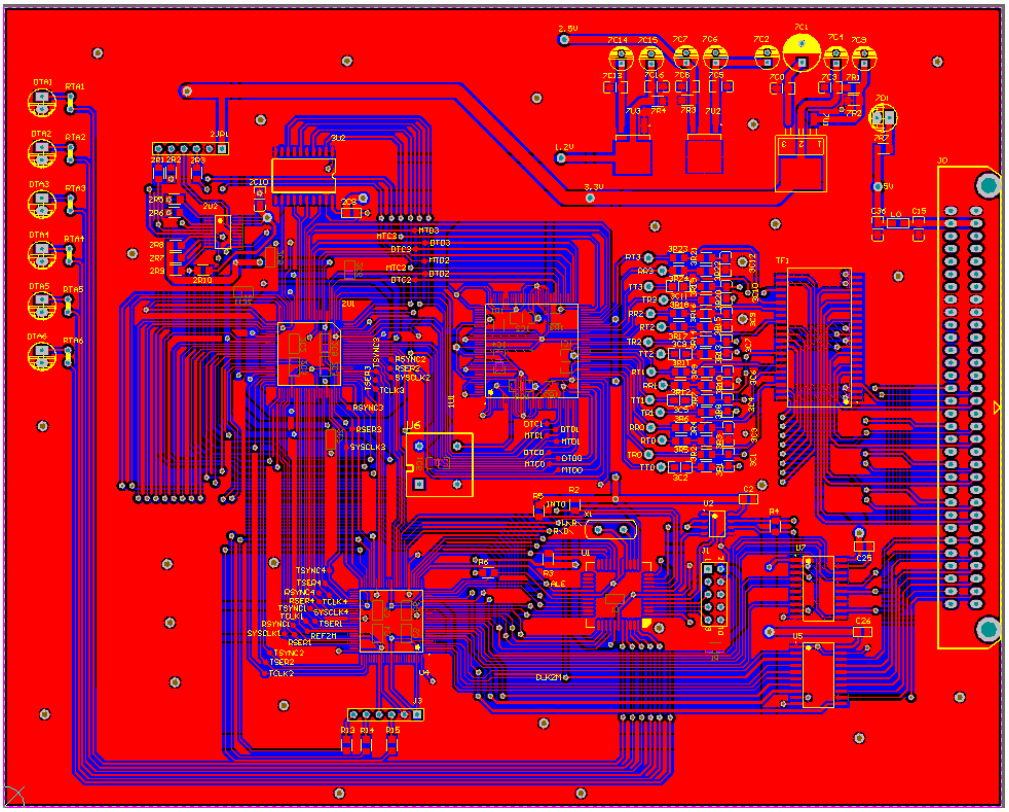
Hình 2.8 mô tả sơ đồ nguyên lý khối FPGA.

Trong khối sử dụng IC XC3S500E phục vụ cho việc xử lý tín hiệu số, đóng gói E1 và một số chức năng khác. Đi kèm với nó có IC xcf04 là bộ nhớ lưu giữ chương trình cho FPGA.

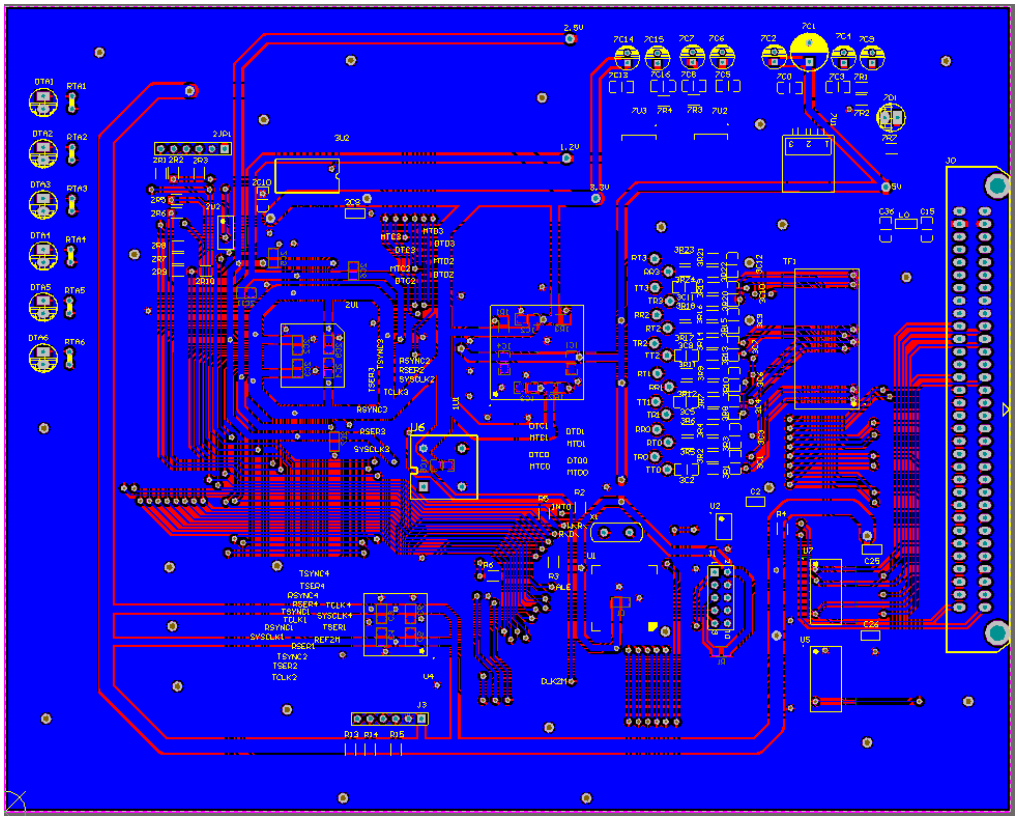


Hình 2.8: Sơ đồ nguyên lý khối FPGA

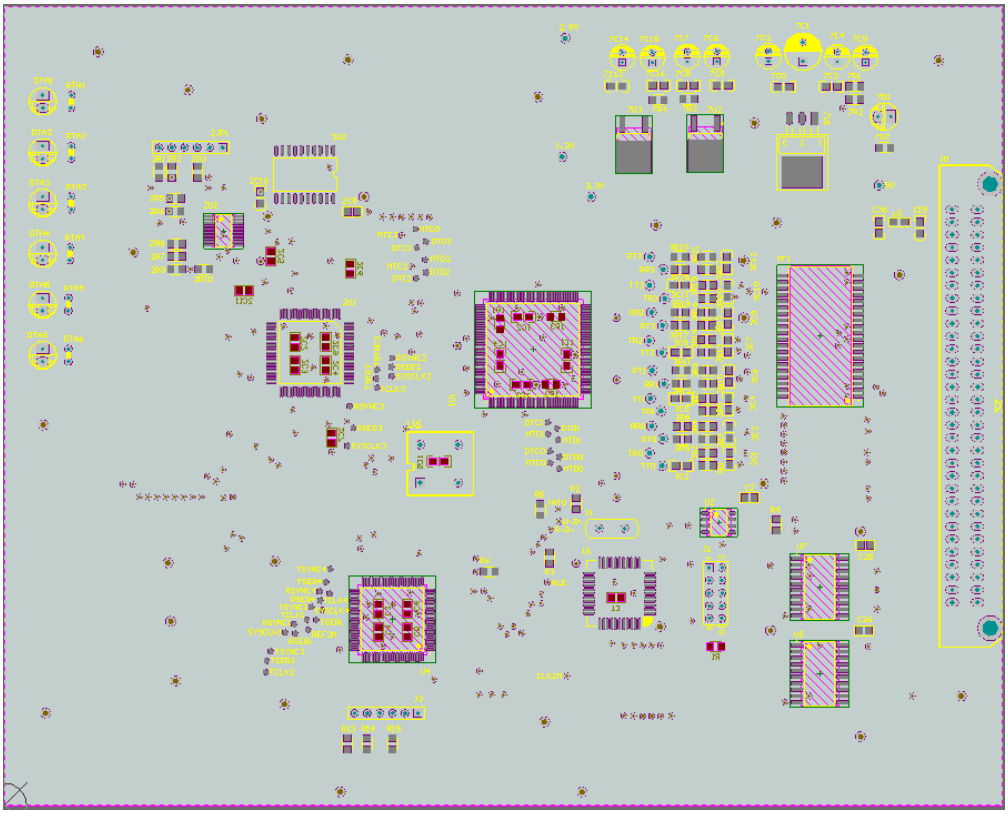
*** Sơ đồ mạch in**



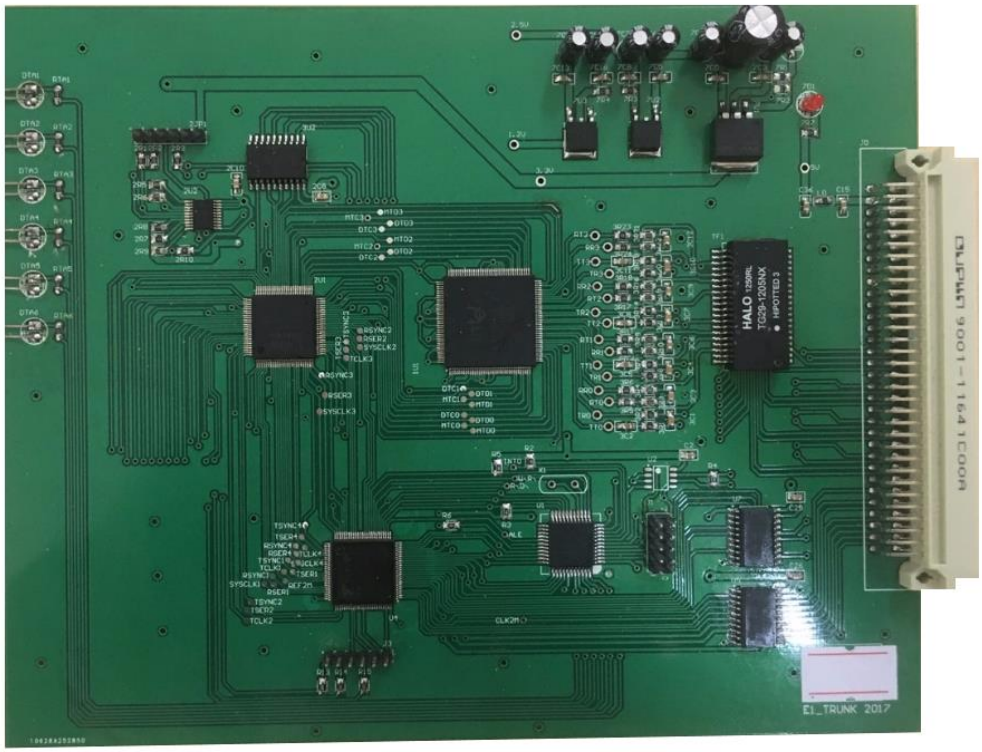
Hình 2.9: Sơ đồ mạch in lớp TOP



Hình 2.10: Sơ đồ mạch in lớp BOTTOM



Hình 2.11: Sơ đồ bố trí linh kiện



Hình 2.12: Mạch thực tế

2.2.2 – Thiết kế phần mềm

Ta thực hiện thiết kế modul đóng khung E1 bằng cách viết chương trình theo ngôn ngữ VHDL trên phần mềm ISE (phiên bản 10.1) sau đó dịch ra các file định dạng .bit và .mcs để nạp cho các IC.

*** Giới thiệu về ISE**

ISE (Integrate Software Environment) – môi trường phần mềm tích hợp – là một bộ phần mềm thiết kế của Xilinx. ISE cho phép tạo ra các sản phẩm thiết kế thông qua việc nhập các thiết kế vào thiết bị chương trình hóa của Xilinx. ISE cho phép có thể chọn một hoặc nhiều phương án thiết kế khác nhau bao gồm:

- Thiết kế bằng ngôn ngữ mô tả phần cứng HDL (VHDL, verilog HDL, ABEL).
- Thiết kế dưới dạng sơ đồ cổng logic (Schematic).
- EDIF (Electronic Data Interchange Format).
- NGC/NGO (kết hợp với Core Generator).
- Đồ hình trạng thái (State Machines).
- IP codes.

Với mỗi phương án thiết kế, có một file nguồn, ISE cho phép nhanh chóng xác minh chức năng của file nguồn này nhờ khả năng mô phỏng tích hợp bên trong, bao gồm MODELSIM và HDL Bench. Với phương án thiết kế bằng ngôn ngữ mô tả phần cứng (HDL) thiết kế được tổng hợp bằng công nghệ tổng hợp của Xilinx (XST), nó có thể là một phần mềm chạy độc lập, hay được tích hợp vào trong ISE. Công cụ thực hiện thiết kế tiếp tục quá trình sắp đặt và kết nối trong FPGA và kết thúc của quá trình một dòng bit được hình thành cấu hình nên thiết kế của ta. Giao diện Project Navigator cung cấp một quá trình toàn diện toàn bộ quá trình thiết kế.

*** Sơ đồ khối của thiết kế**

Sơ đồ khối của thiết kế được mô tả ở hình 2.12. Trong hình có 5 khối chính.

Khối *Control_signals and Common_registers* thực hiện 2 chức năng cơ bản. Một là tạo các tín hiệu điều khiển chung cho cả modul hoạt động, trong đó có việc tạo các tín hiệu định thời, các bộ đếm và xung nhịp nhằm đồng bộ các tín hiệu và

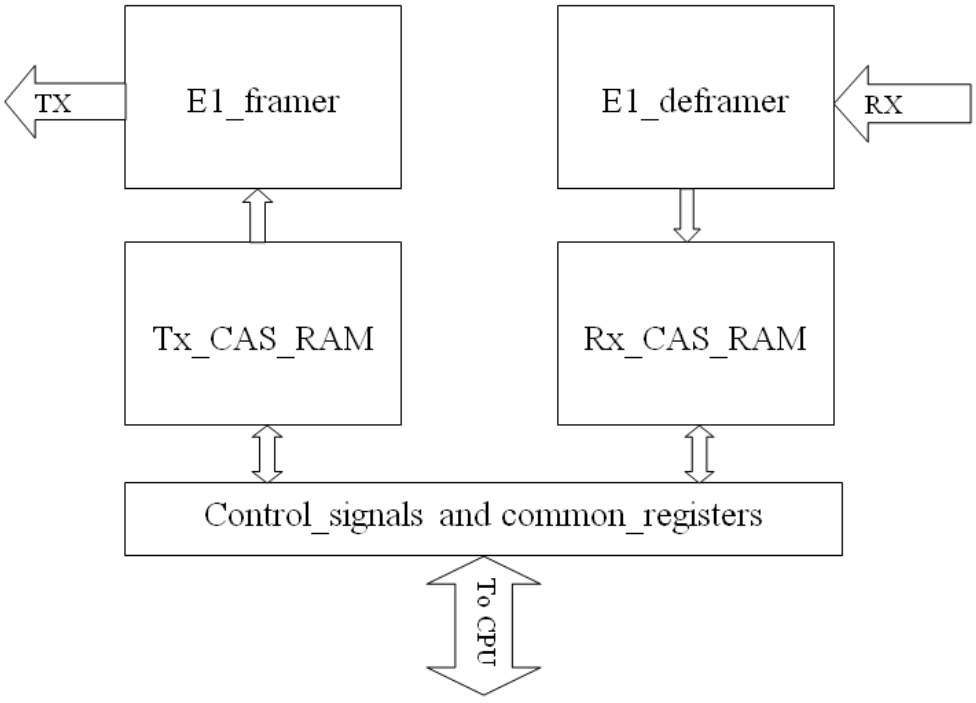
các khối. Hai là thực hiện giao tiếp với CPU để lấy thông tin báo hiệu (CAS) từ CPU cần truyền đi, đồng thời nhận các tín hiệu báo hiệu từ kênh truyền dẫn đưa về cho CPU xử lý.

Khối *Tx_CAS_RAM* là khối thực hiện lưu trữ và đồng bộ thông tin báo hiệu nhận được từ CPU để truyền đi.

Khối *Rx_CAS_RAM* là khối thực hiện tách ghép, xử lý và lưu trữ thông tin báo hiệu nhận được từ kênh truyền để đưa về CPU.

Khối *E1_framer* là khối thực hiện ghép dữ liệu cần truyền đi và thông tin báo hiệu được lưu trữ ở khối *Tx_CAS_RAM* thành khung E1 theo chuẩn của ITU-T.

Khối *E1_deframer* là khối thực hiện tách dữ liệu nhận được từ kênh truyền (theo chuẩn E1) thành dữ liệu của các khe thời gian tương ứng, đồng thời tách thông tin báo hiệu nhận được để gửi tới khối *Rx_CAS_RAM*.



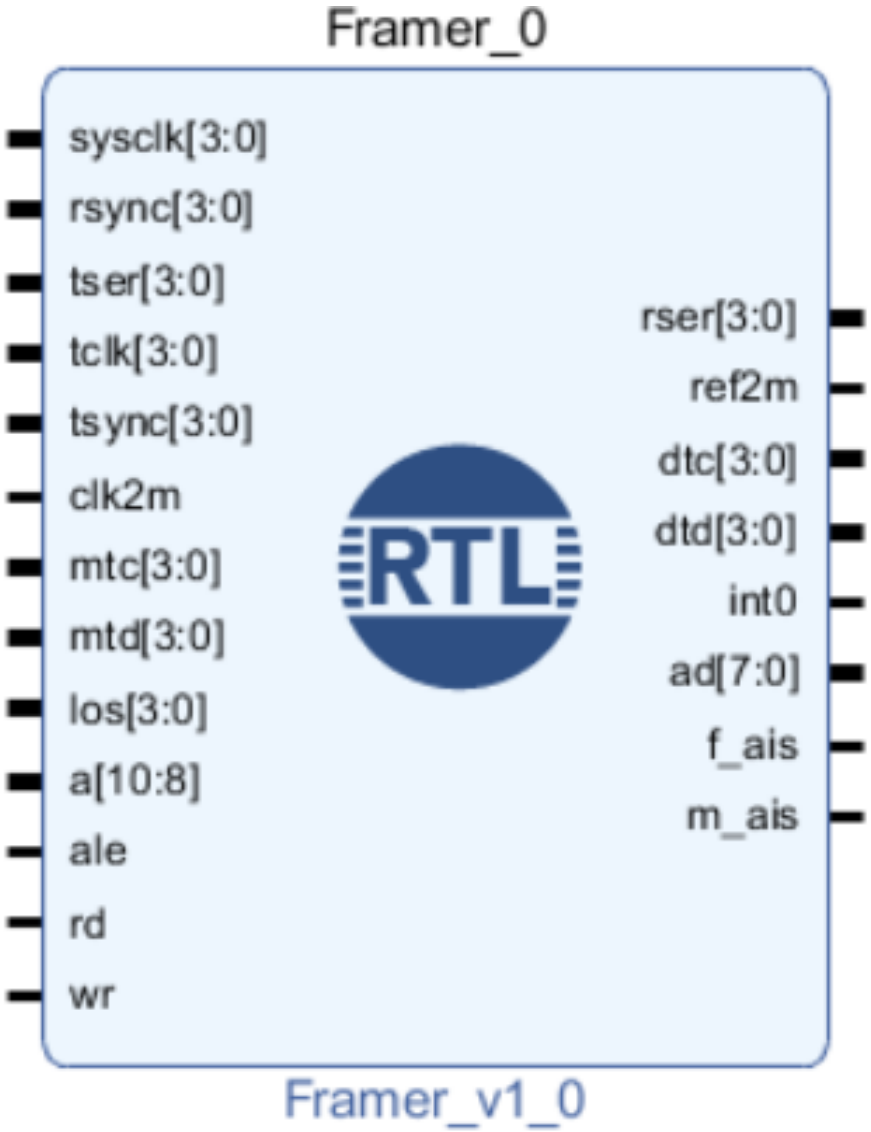
Hình 2.13: Sơ đồ khối thiết kế phần mềm

*** Mô tả RTL các khối**

- Khối top:

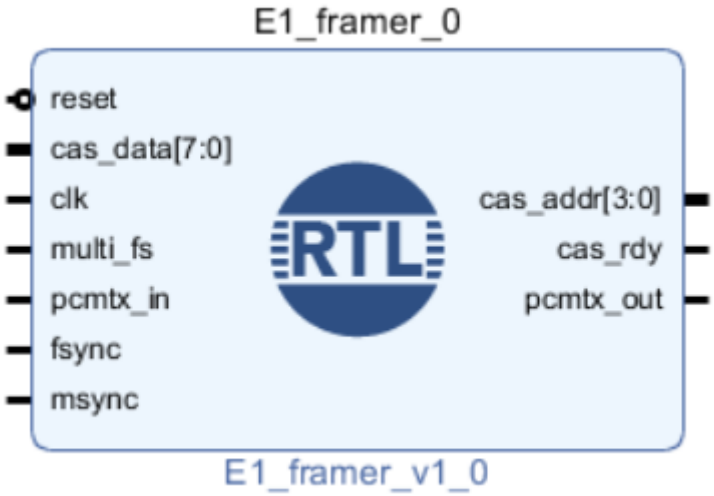
Hình 2.14 mô tả RTL modul đóng khung E1. Trong số đó có các tín hiệu phía *STbus side* như đường dữ liệu vào *tser* (dữ liệu cần đóng gói E1 và phát đi), đường

dữ liệu ra *rser* (dữ liệu đã giải đóng khung), các tín hiệu clock *sysclk, tclk*, các tín hiệu định thời đa khung *tsync, rsync*; phía *LIU side* có đường dữ liệu vào *mtd* (dữ liệu nhận được từ đường truyền), đường dữ liệu ra *dtd* (dữ liệu đã đóng khung và phát ra đường truyền), đường clock vào *mtc*, đường clock ra *dtc*; ngoài ra là các tín hiệu giao tiếp với CPU như *ale, a, ad, rd, wr*.



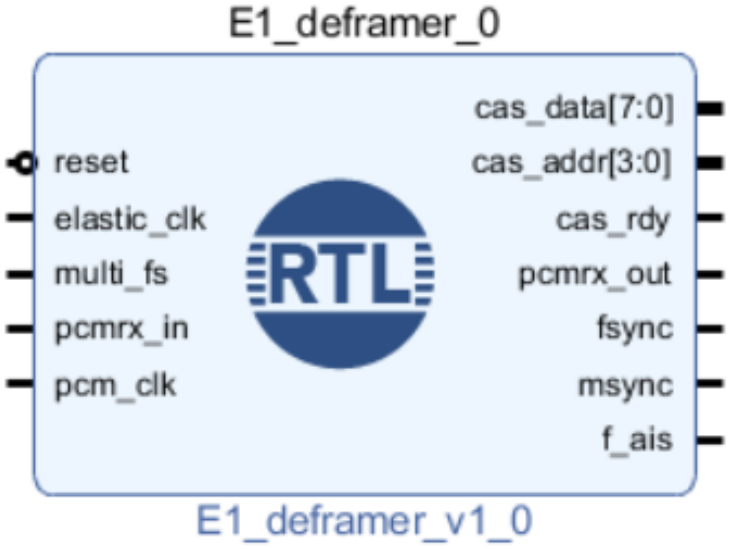
Hình 2.14: Mô tả vào ra của khối top

- Khối E1_framer: Hình 2.15 mô tả RTL khối E1_framer, chính là khối đóng gói dữ liệu phát. Dữ liệu cần phát đi *pcmtx_in* kết hợp với thông tin báo hiệu *cas_data* và các thông tin đồng bộ khung, đa khung được xử lý đóng gói thành luồng dữ liệu đầu ra *pcmtx_out*.



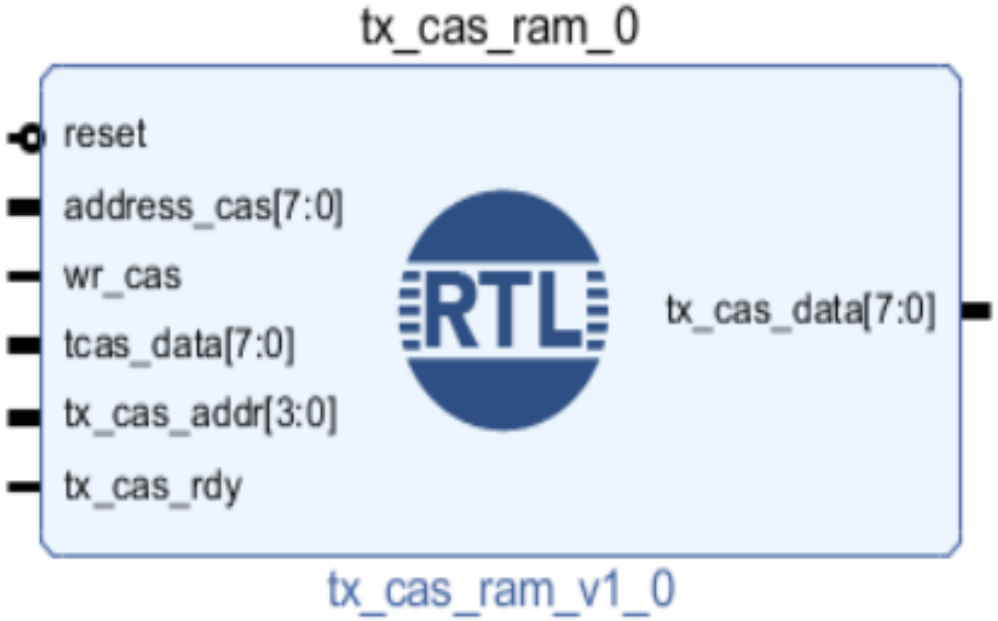
Hình 2.15: Mô tả khối E1_framer

- Khối E1_deframer: Hình 2.16 mô tả RTL khối E1_deframer, chính là khối giải đóng gói dữ liệu thu. Dữ liệu thu được *pcmrx_in* được xử lý tách khung, đa khung theo đúng timeslot dựa vào các thông tin đồng bộ thành luồng dữ liệu thu về *pcmrx_out* và thông tin báo hiệu thu về *cas_data*.



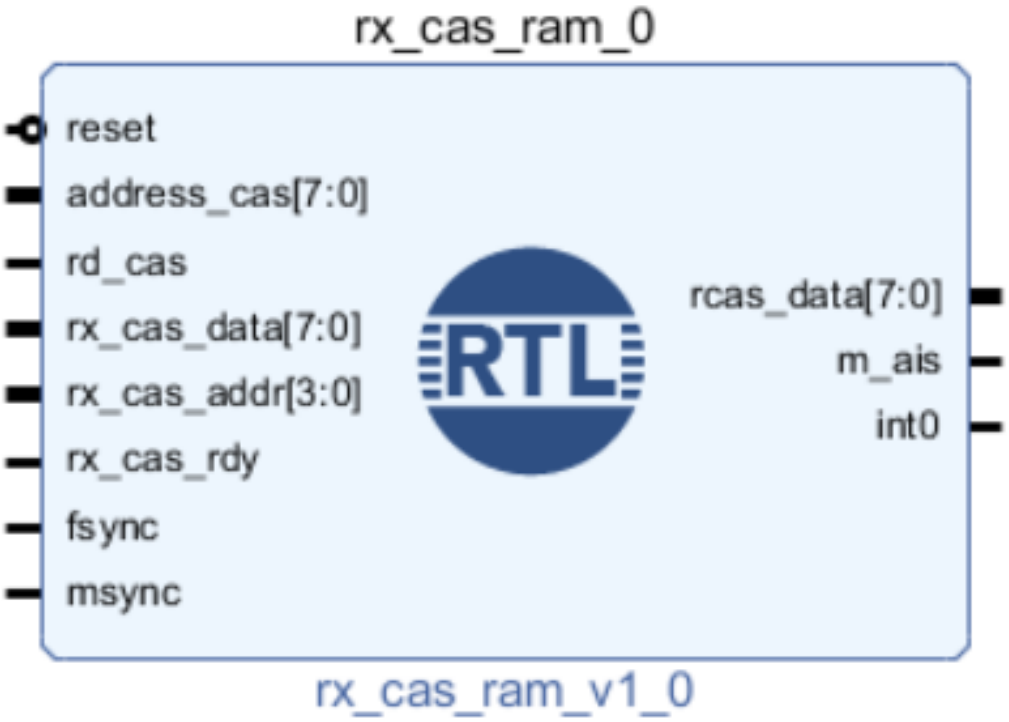
Hình 2.16: Mô tả khối E1_deframer

- Khối Tx_CAS_RAM: Hình 2.17 mô tả RTL khối Tx_CAS_RAM. Thông tin báo hiệu *tcas_data* được truyền từ CPU xuống, được lưu vào khối Tx_CAS_RAM tương ứng với kênh của nó, đầu ra là *tx_cas_data* sẽ được đưa vào khung E1 dựa vào các thông tin định thời.



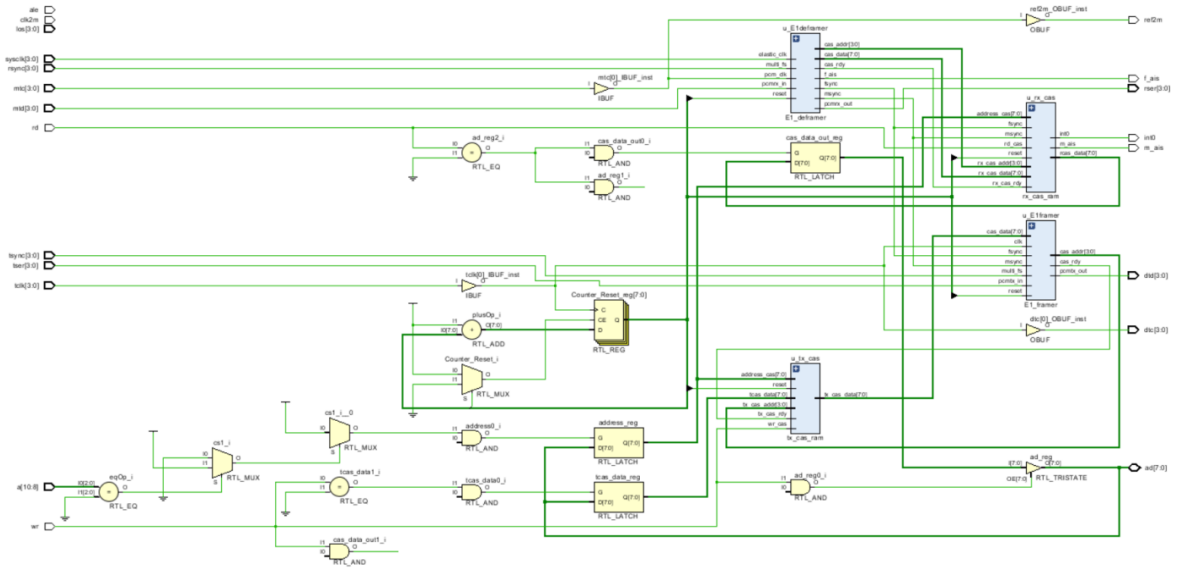
Hình 2.17: Mô tả khối lưu trữ thông tin báo hiệu CAS truyền đi

- Khối Rx_CAS_RAM: Hình 2.18 mô tả RTL khối Rx_CAS_RAM. Thông tin báo hiệu thu được từ khối E1_deframer *rx_cas_data* được lưu vào khối Rx_CAS_RAM tương ứng với kênh của nó, đầu ra là *rcas_data* sẽ được CPU đọc về.



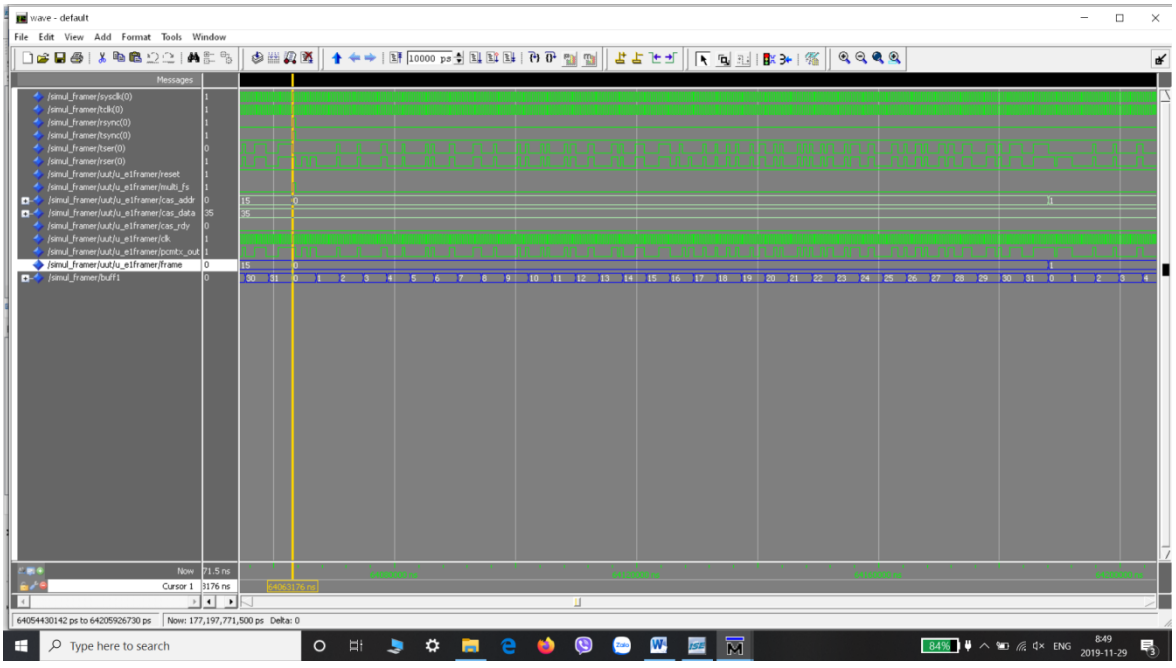
Hình 2.18: Mô tả khối lưu trữ thông tin báo hiệu CAS nhận được

*** Sơ đồ nguyên lý của thiết kế**



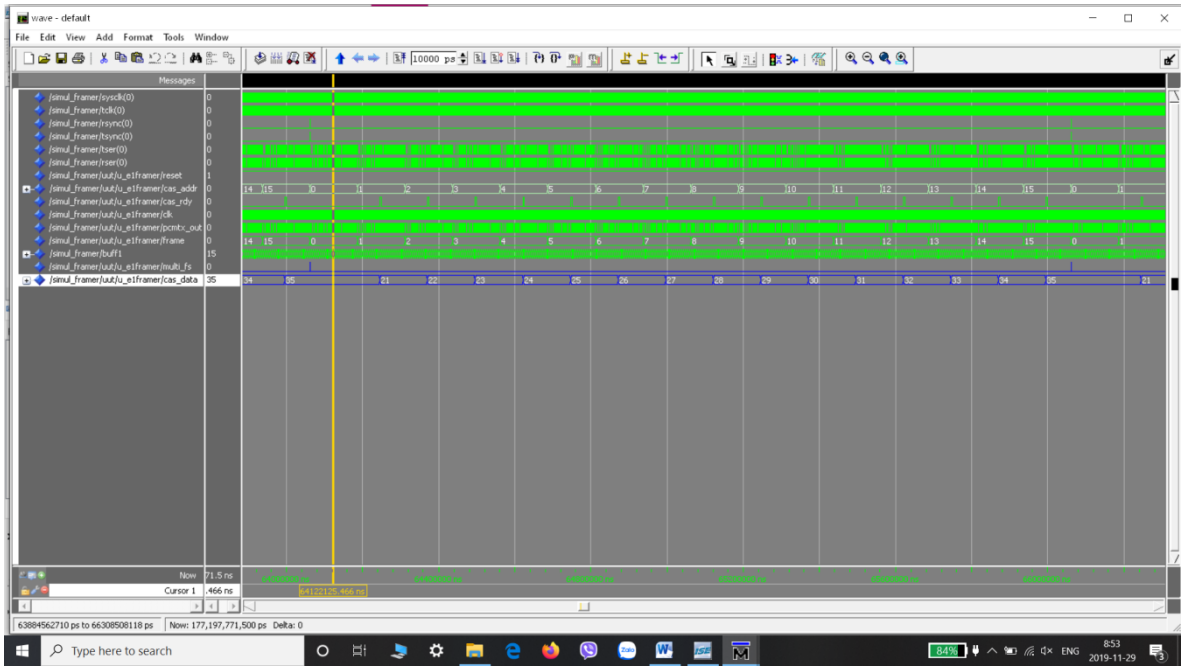
Hình 2.19: Sơ đồ nguyên lý của thiết kế

2.3 – Mô phỏng

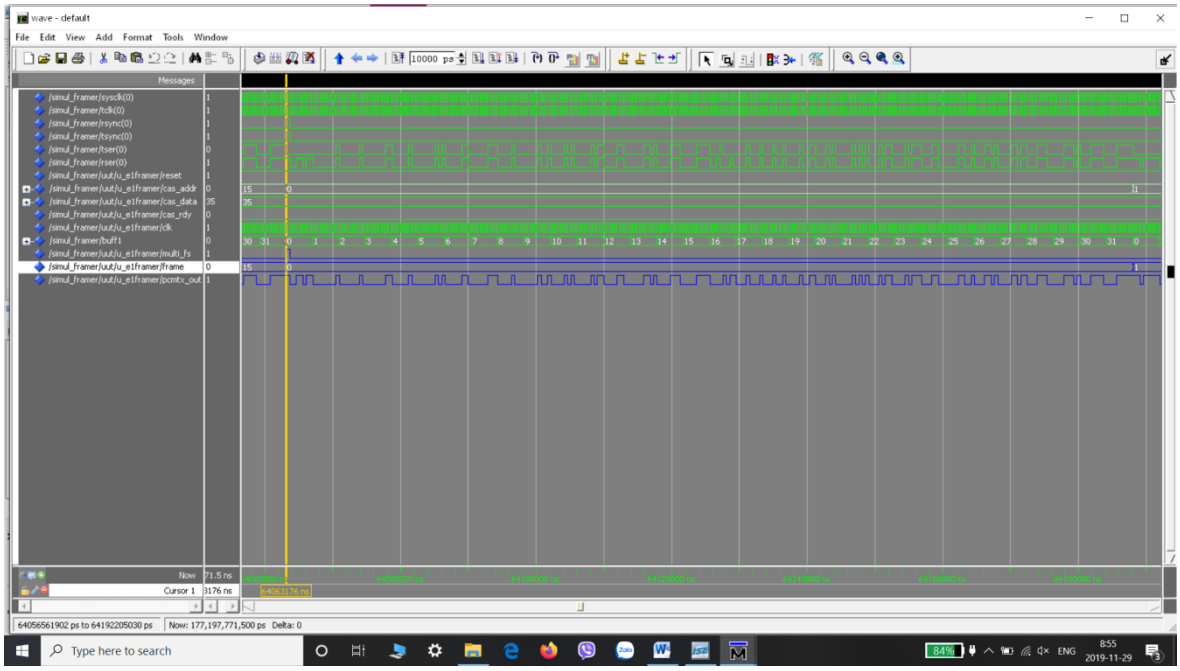


Hình 2.20: Mẫu dữ liệu mô phỏng đưa vào hệ thống

Dữ liệu đầu vào đưa vào hệ thống (Hình 2.20): Truyền 1 chuỗi dữ liệu là các byte có giá trị từ 0 đến 31 tương ứng với 32 timeslot (tuy nhiên những timeslot dùng cho đồng bộ khung, đồng bộ đa khung và báo hiệu CAS sẽ nhận dữ liệu từ nguồn khác khi truyền ra đường truyền).

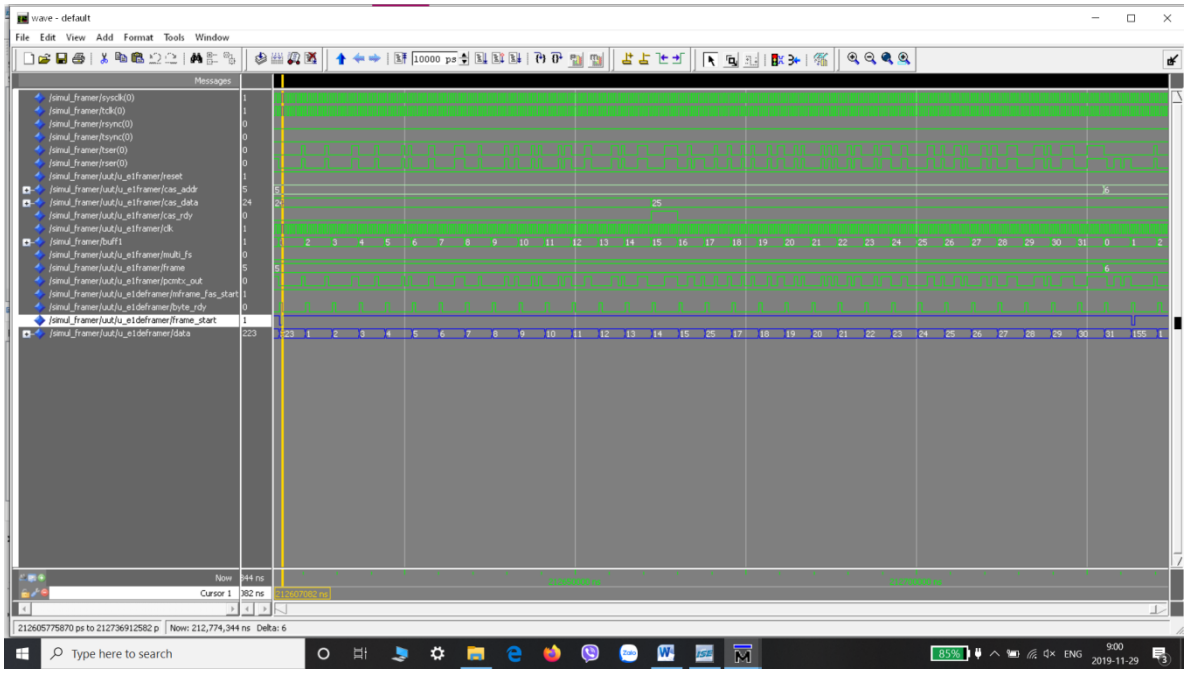


Hình 2.21: Mẫu dữ liệu báo hiệu CAS đưa vào hệ thống

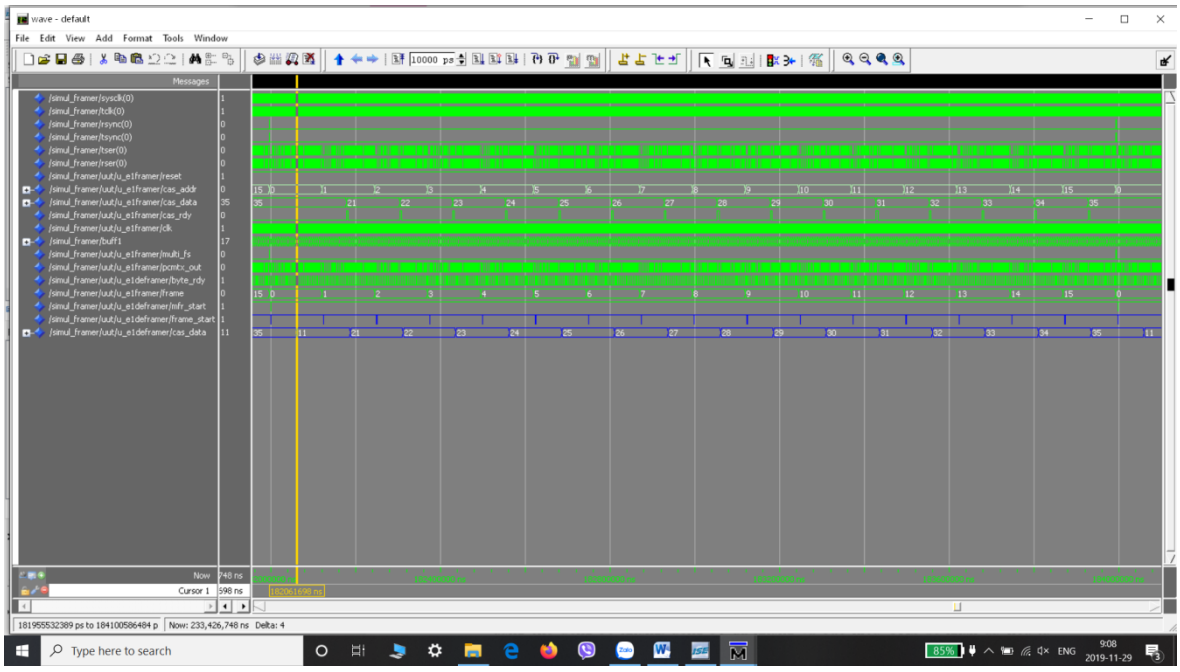


Hình 2.22: Dạng xung đưa ra đường dây sau khi đã đóng khung E1

Dữ liệu kênh báo hiệu CAS đầu vào: Truyền 1 chuỗi các byte có giá trị từ 21 đến 35 tương ứng với timeslot thứ 16 của F1 cho đến khung F15 trong một đa khung.



Hình 2.23: Chuỗi dữ liệu nhận được sau khi giải đóng khung



Hình 2.24: Chuỗi thông tin báo hiệu CAS nhận được sau khi giải đóng khung

Sử dụng luôn chuỗi xung đầu ra khối đóng khung E1 làm đầu vào cho khối giải đóng khung E1 để kiểm tra về mặt lý thuyết hoạt động của thiết kế mà ta xây dựng.

Ta nhận thấy chuỗi dữ liệu nhận được (ngoại trừ timeslot 0 và timeslot 16) trùng khớp với chuỗi dữ liệu mà ta đưa vào. Như vậy là hệ thống đã truyền nhận dữ liệu chính xác.

Thông tin báo hiệu nhận được chính là chuỗi thông tin báo hiệu ta đã gửi đi, 21 đến 35, tương ứng với timeslot 16 của khung F1 đến khung F15. Khung F0 sẽ nhận được thông tin đồng bộ đa khung.

Như vậy hệ thống đóng khung và giải đóng khung mà ta thiết kế đã thực hiện được chức năng của nó thông qua việc truyền đi một chuỗi dữ liệu và thu về được đúng chuỗi dữ liệu đó.

2.4 – Tổng kết chương 2

Chương 2 tiến hành xem xét cụ thể IC mà ta dùng cho thiết kế là xc3s500E, một FPGA thuộc họ Spartan 3E, xem xét cấu trúc và tài nguyên của nó, làm cơ sở để đánh giá độ phù hợp và mức đáp ứng cho thiết kế.

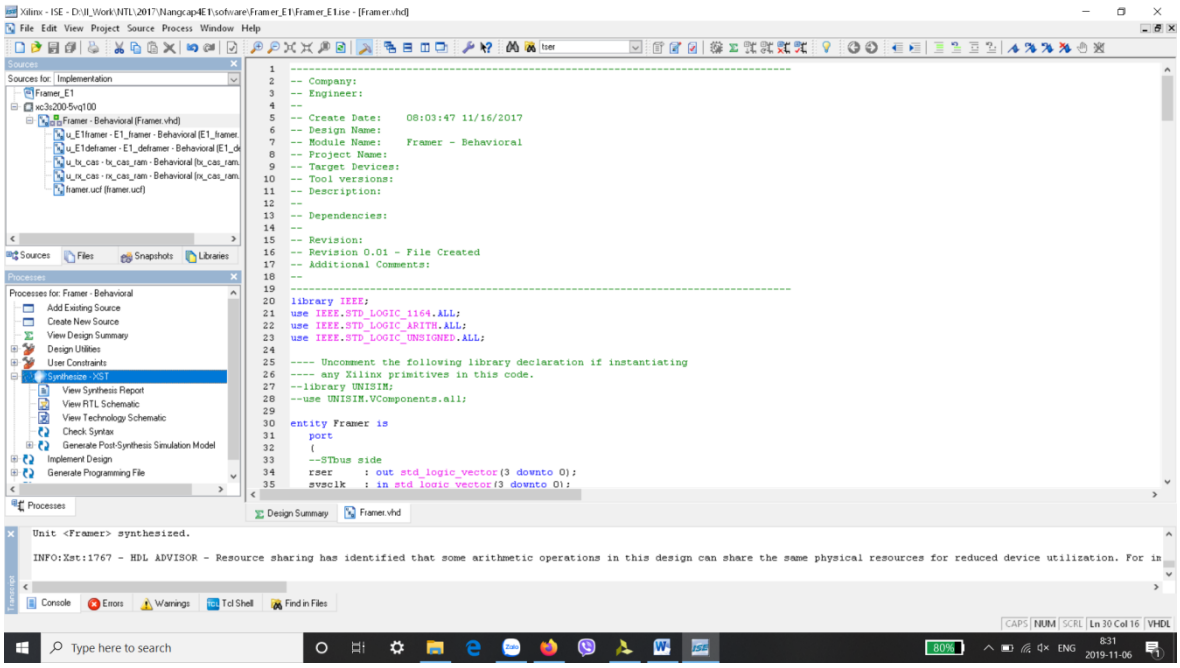
Chương 2 cũng tiến hành các bước thiết kế phần cứng từ sơ đồ nguyên lý đến bố trí linh kiện và layout ra board mạch thực thi, đặt gia công ở nhà máy, tiến hành hàn dán các linh kiện lên board mạch, kiểm tra các đường cấp nguồn, đất và tín hiệu của board mạch.

Đồng thời tiến hành các bước thiết kế phần mềm như lên sơ đồ khối chức năng, sơ đồ khối chi tiết, xây dựng thuật toán, tiến hành viết code, thực hiện mô phỏng.

CHƯƠNG 3 – THỰC THI VÀ KẾT QUẢ

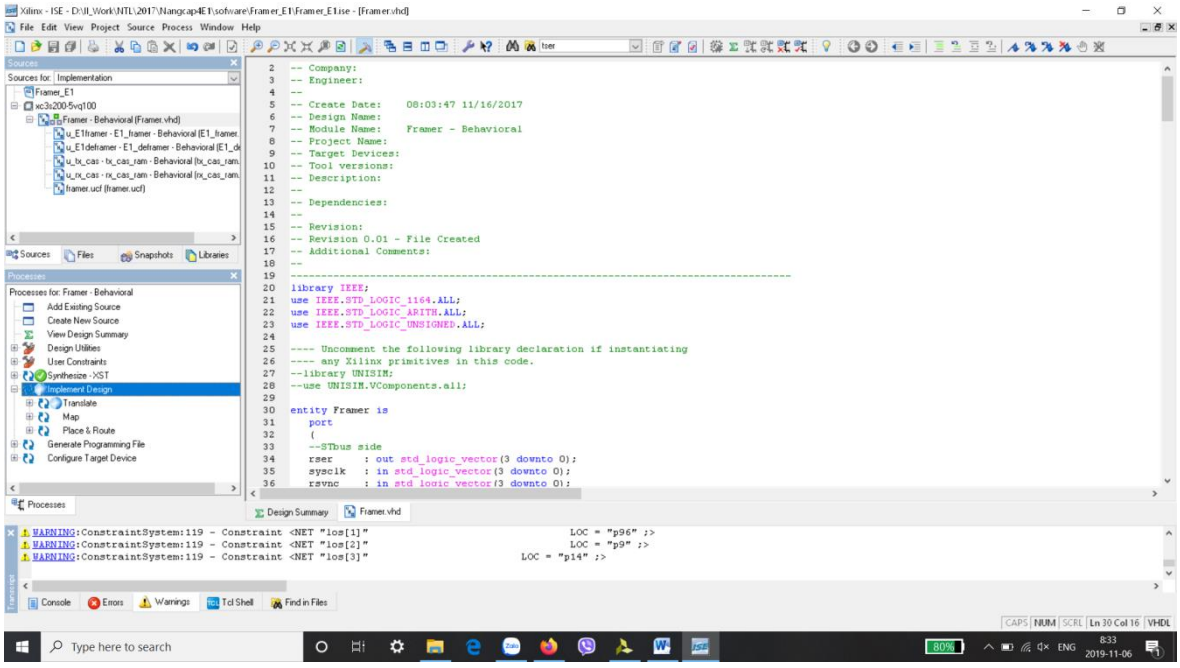
3.1 – Tổng hợp thiết kế và chạy thử

3.1.1 – Tổng hợp thiết kế



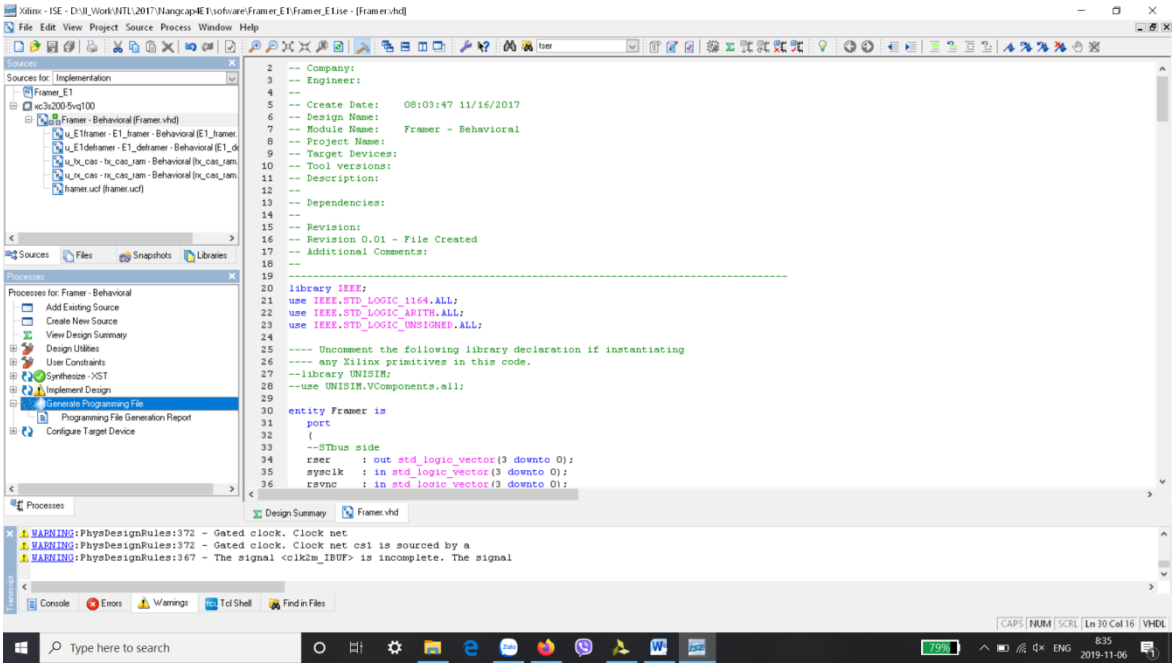
Hình 3.1: Tổng hợp thiết kế trên công cụ ISE

3.1.2 – Thực hiện thiết kế



Hình 3.2: Thực hiện thiết kế trên công cụ ISE

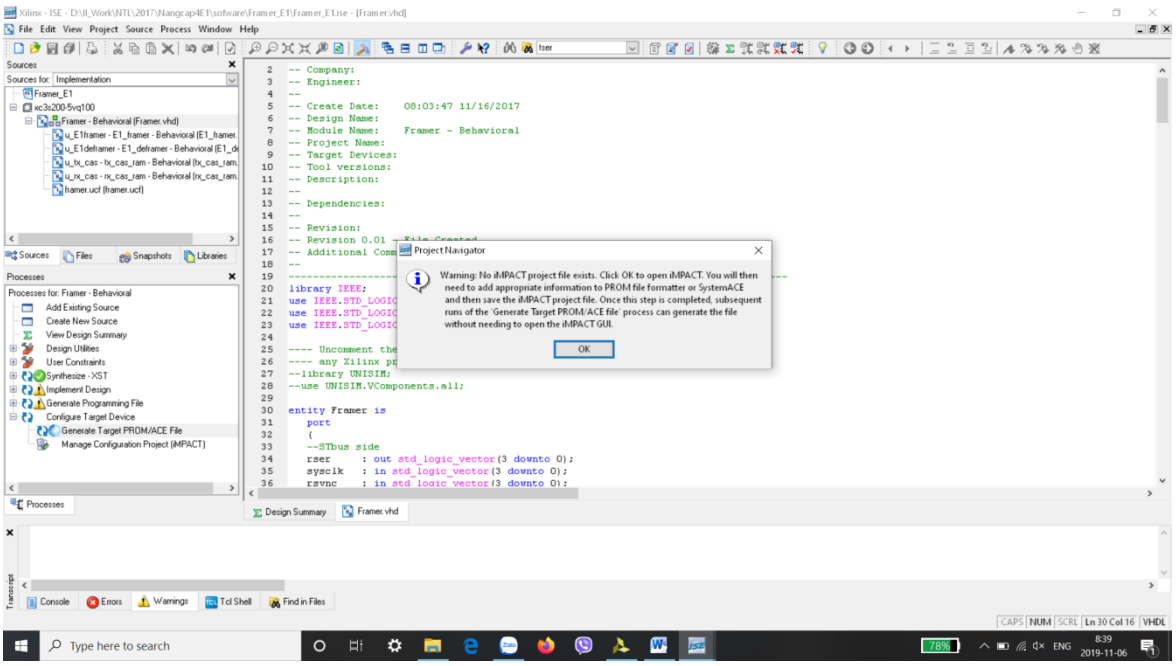
3.1.3 – Tạo bitstream nạp vào FPGA



Hình 3.3: Tạo file bitstream nạp vào FPGA

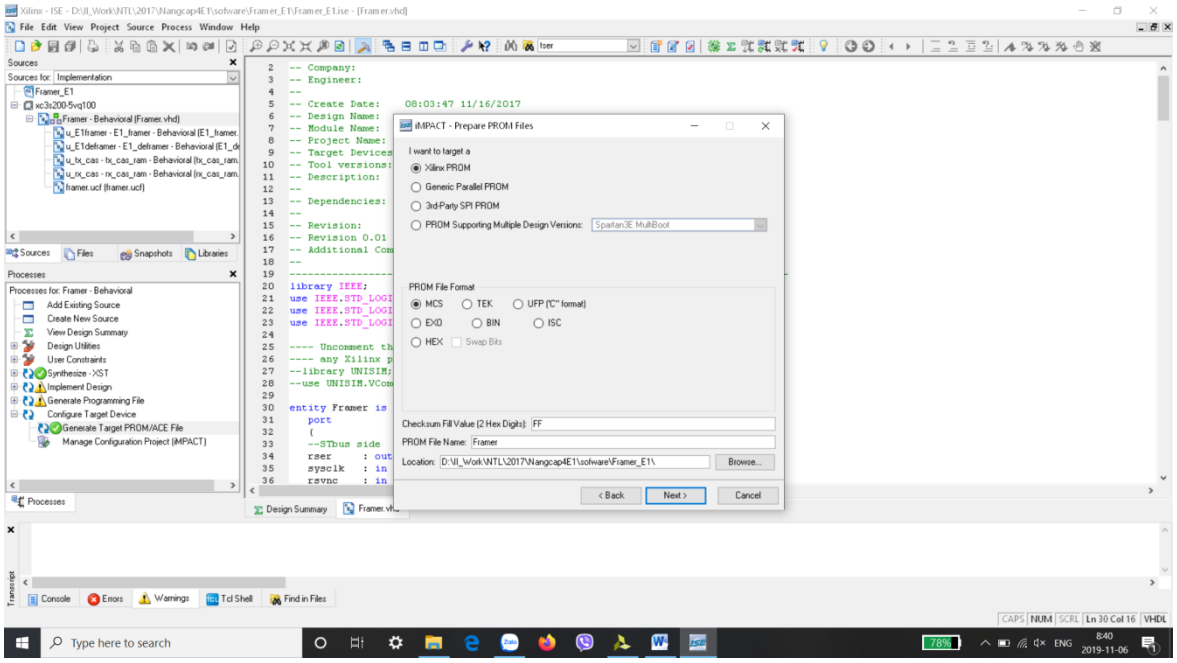
3.1.4 – Tạo file nạp cho ROM ngoài của FPGA

Mở công cụ impact:



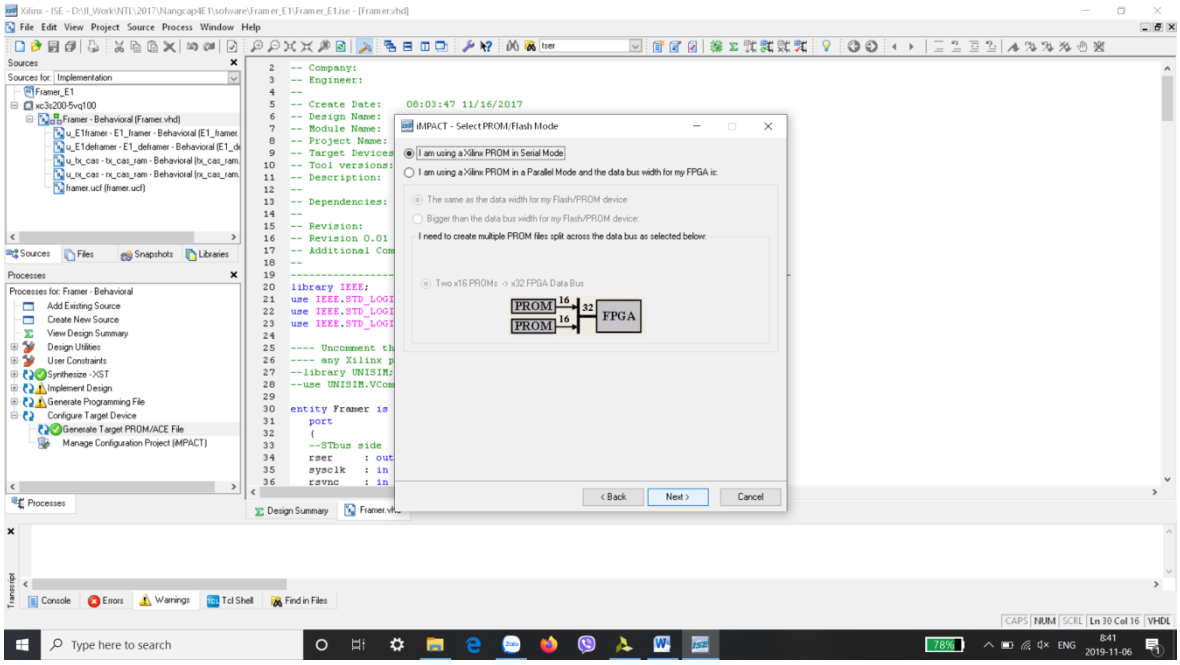
Hình 3.4: Mở công cụ impact

Đặt tên và chọn định dạng file:



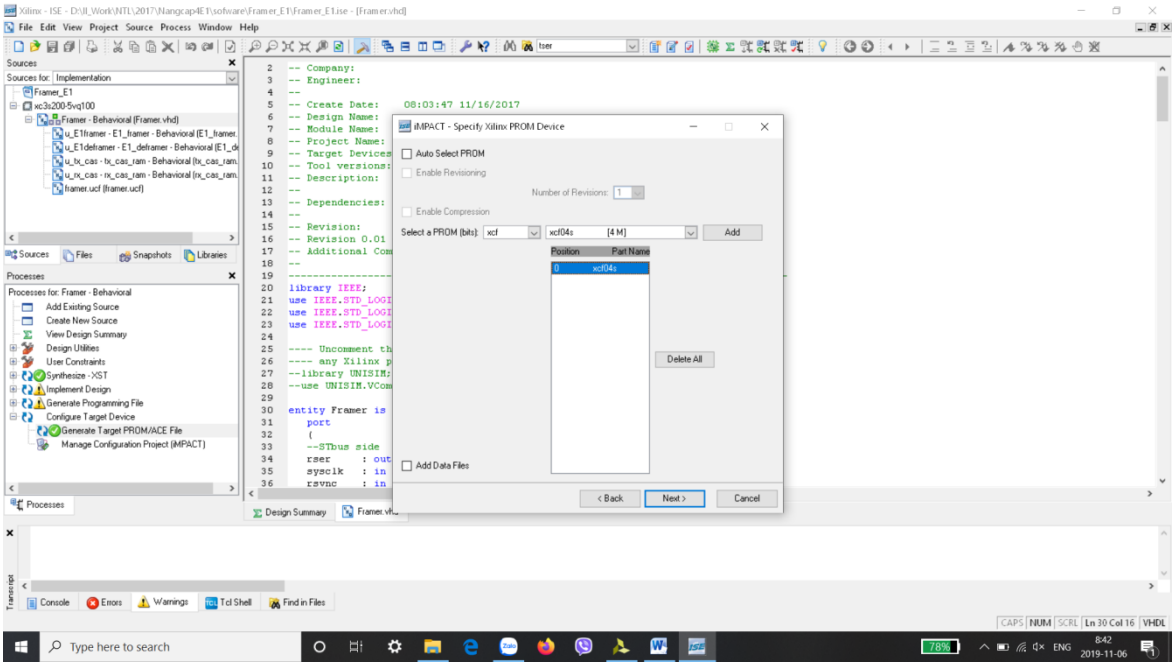
Hình 3.5: Đặt tên và chọn định dạng file

Tiếp theo chọn chế độ nạp:



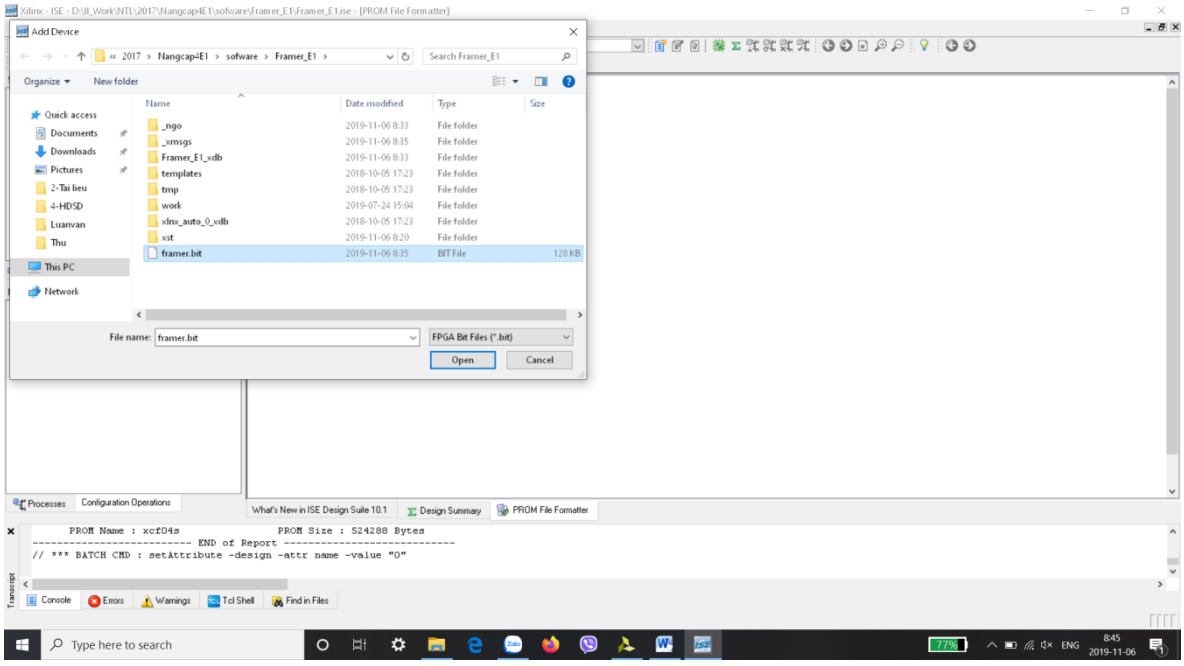
Hình 3.6: Chọn chế độ nạp

Chọn loại ROM tương ứng với FPGA đã chọn



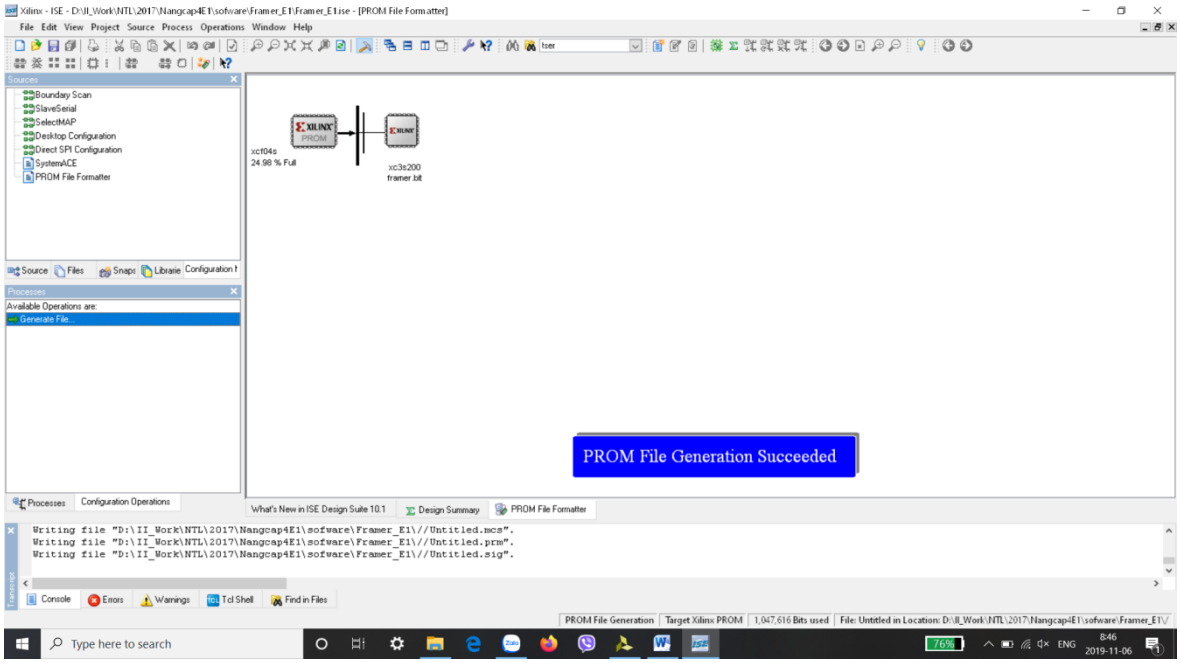
Hình 3.7: Chọn loại ROM tương ứng với FPGA đã chọn

Chọn file bitstream của thiết kế vừa tạo:



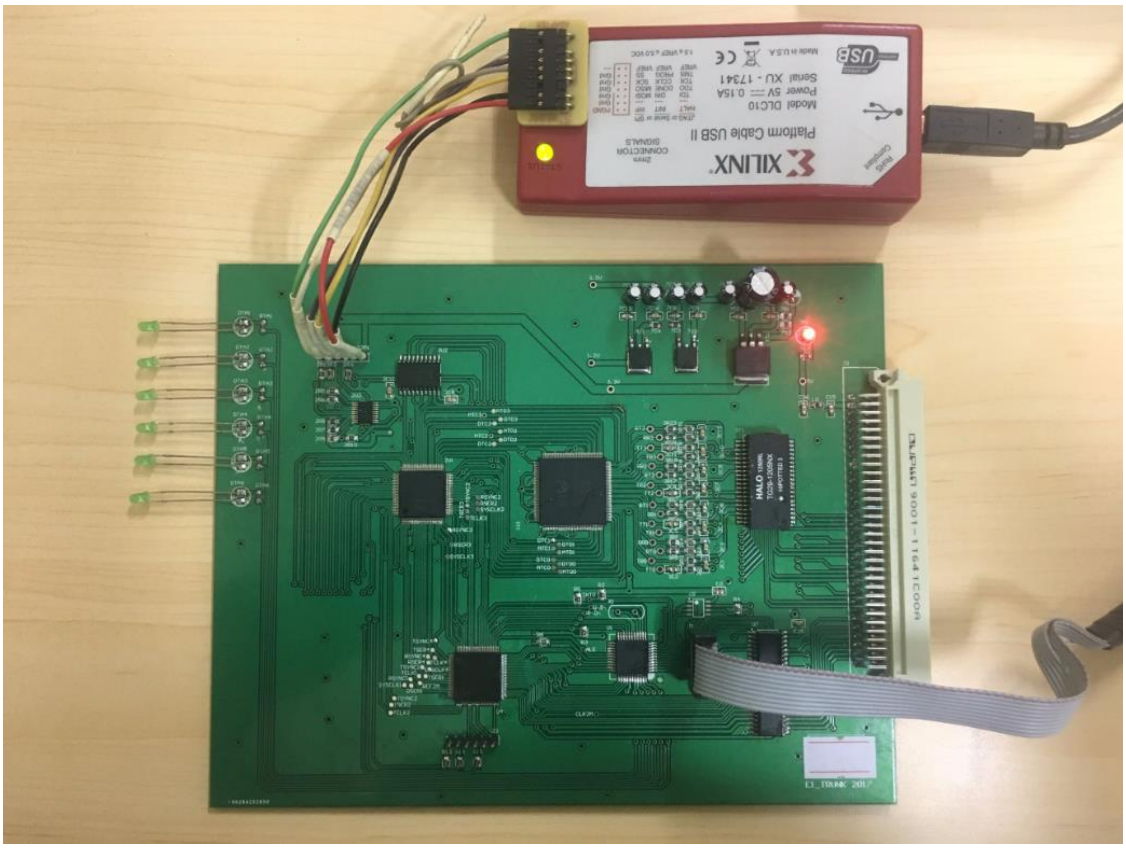
Hình 3.8: Chọn file bitstream

Tạo file .mcs để nạp vào ROM:

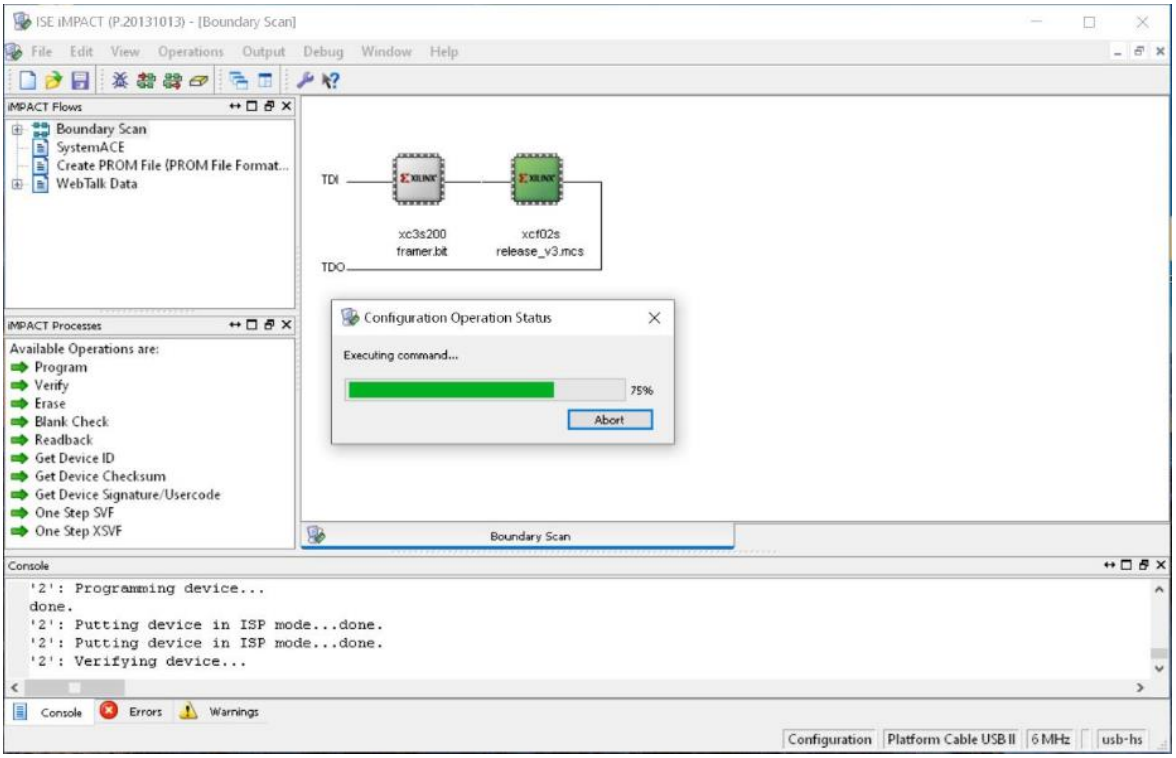


Hình 3.9: Tạo file .mcs từ file bitstream đã có

3.1.5 – Nạp file cấu hình cho FPGA và ROM



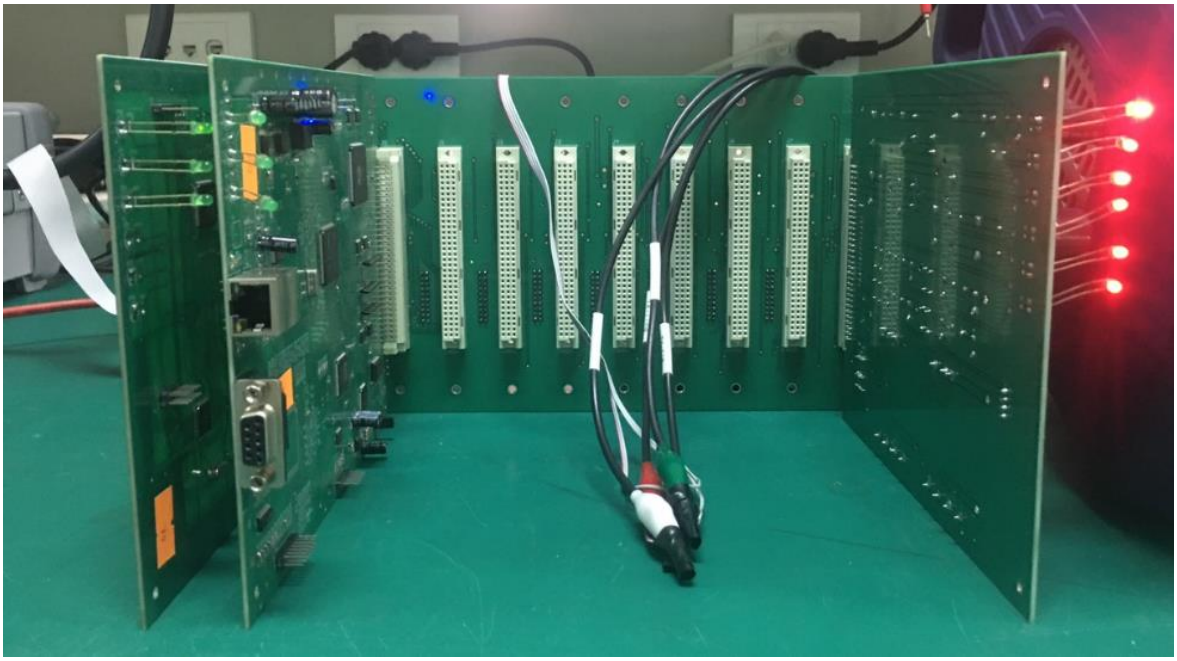
Hình 3.10: Sử dụng bộ nạp DLC10 của Xilinx để nạp cho FPGA và ROM ngoài theo chuẩn JTAG



Hình 3.11: Tiến trình nạp file cấu hình cho FPGA và ROM

3.2 – Kết quả

Sử dụng máy đo luồng VeEX UX400 để đánh giá luồng E1 được phát đi bởi modul đóng khung E1 mà ta thiết kế. Thiết lập hệ thống đo gồm có 1 card backlpane thực hiện nối các tín hiệu giữa các card, 1 card nguồn để cấp nguồn cho toàn hệ thống, 1 card CPU điều khiển toàn bộ hoạt động của hệ thống, và 1 card trung kế E1 mà lõi là modul đóng khung E1.



Hình 3.12: Thiết lập hệ thống đo kiểm

Hình 3.13: Kết nối hệ thống với máy đo luồng VeEX UX400

Hình 3.14: Kết quả đo trên máy đo luồng VeEX UX400

Hình 3.15: Các báo cảnh trên máy đo

Kết nối luồng E1 của hệ thống mà ta xây dựng với luồng E1 của máy đo (TX của hệ thống vào RX của máy đo và ngược lại).

Từ máy đo phát đi 1 luồng E1 và nhận lại báo cảnh đồng bộ ở card luồng E1 thông qua hiển thị của các đèn LED (nháy đều 1s).

Đồng thời từ hệ thống phát đi một luồng E1 và thu được báo cảnh đồng bộ ở máy đo. Hình 3.14 và 3.15 thể hiện điều đó. Luồng E1 mà máy đo nhận được từ hệ thống hiển thị đồng bộ, không có lỗi (*No errors - OK*). Các báo cảnh LOS (*Loss Of Signal*), AIS (*Alarm Indication Signal*), LOF (*Loss Of Frame*), RDI (*Remote Defect Indication*), Cod (*Violation on coding sequence*), FAS (*Frame Alignment Signal*) đều hiển thị màu xanh, tức là không có báo cảnh (nếu có báo cảnh sẽ hiển thị màu vàng).

3.3 – Nhận xét và đánh giá kết quả

Trong quá trình nghiên cứu, luận văn đã áp dụng các lý thuyết về xử lý tín hiệu số, ghép kênh số. Luận văn đã tham khảo nhiều tài liệu khác nhau, các khuyến nghị, tiêu chuẩn của các tổ chức uy tín hàng đầu thế giới, từ đó xây dựng các khối một cách mềm dẻo, tối ưu sao cho đáp ứng được bài toán đặt ra ban đầu.

Kết quả mô phỏng cho thấy về mặt lý thuyết modul đóng khung E1 hoạt động chính xác theo yêu cầu đã đề ra khi thực hiện giả lập đưa một chuỗi dữ liệu vào modul thực hiện phát đi trên kênh truyền sau đó thu lại thực hiện giải đóng khung nhận được chuỗi dữ liệu trùng khớp chuỗi ban đầu.

Kết quả thực tế nhận được qua việc thực hiện đo kiểm đánh giá bằng thiết bị đạt chuẩn quốc tế, kết nối luồng E1 mà modul tạo ra với luồng E1 của máy cho thấy hệ thống đồng bộ hoàn toàn, không có chỉ thị báo cảnh. Điều này cho phép kết luận modul mà luận văn xây dựng đáp ứng được các chỉ tiêu, yêu cầu đã đề ra.

3.4 – Tổng kết chương 3

Chương 3 tiến hành tổng hợp thiết kế (mã nguồn VHDL) trên công cụ ISE, tạo ra các file nạp cho FPGA (file .bit) và file nạp cho ROM ngoài của FPGA (file .mcs). Tiến hành nạp file cấu hình đã tạo ra cho FPGA và ROM bằng công cụ Impact của Xilinx.

Triển khai hệ thống đo kiểm cấu hình tương đương một tổng đài trong đó modul đóng khung E1 mà luận văn thiết kế đóng vai trò là một trung kế E1, ngoài ra có thêm card backplane, card nguồn và card CPU. Kết nối luồng đầu ra của trung kế E1 với máy đo luồng VeEX UX400. Cấu hình cho máy đo luồng ở đúng chế độ đo E1/PDH, PCM30.

Kết quả đo kiểm thu được thể hiện luồng đồng bộ giữa card E1 và máy đo luồng. Máy đo thể hiện không có lỗi, không có các báo cảnh. Có thể kết luận modul đóng khung E1 làm việc tốt.

KẾT LUẬN

Như vậy sau một thời gian nghiên cứu với sự nỗ lực của bản thân và sự hướng dẫn tận tình của TS. Nguyễn Ngọc Minh, đề tài **"Nghiên cứu thiết kế modul đóng khung E1 bằng FPGA"** của học viên đã hoàn thành với một số kết quả sau:

- Nắm được kiến thức cơ bản về công nghệ FPGA, hiểu được tư tưởng luồng thiết kế trên công nghệ FPGA, khả năng xử lý dữ liệu của công nghệ FPGA.

- Hiểu và lập trình được bằng ngôn ngữ mô tả phần cứng VHDL, sử dụng ngôn ngữ VHDL để thiết kế được 1 lõi xử lý tín hiệu số, cụ thể là modul đóng khung E1 trong hệ thống truyền dẫn số.

- Làm chủ và sử dụng thành thạo công cụ thiết kế mạch in Altium, công cụ thiết kế FPGA như phần mềm ISE, phần mềm mô phỏng modelSim.

- Nghiên cứu và thiết kế thành công modul đóng khung E1 ứng dụng vào bảng mạch thực tế trong một thiết bị truyền dẫn quân sự.

Những hạn chế và hướng phát triển của đề tài:

- Do thời gian thực hiện đề tài có hạn, chịu chi phối nhiều nhiệm vụ khác nhau nên chưa tối ưu được thiết kế. Bản thân modul đóng khung E1 chưa tận dụng hết được nguồn tài nguyên khá lớn của IC đã chọn.

- Trong thời gian tới học viên sẽ tiếp tục hoàn thiện đề tài của mình, nâng cấp lên các khung lớn hơn như E2, E3 hay thậm chí các đóng gói SDH như STM-1, STM-4. Sử dụng các công nghệ FPGA mới có năng lực rất lớn và sử dụng các công cụ cao cấp như Vivado thay vì ISE đang dần lỗi thời.

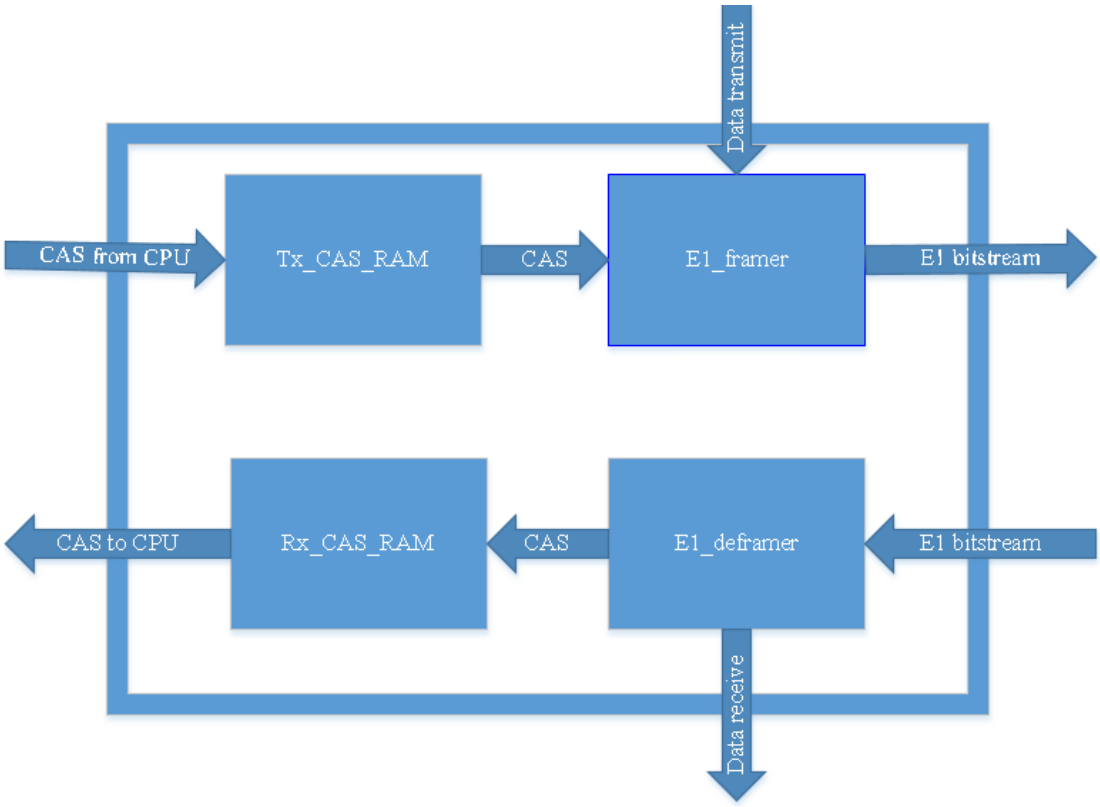
Học viên rất mong nhận được những góp ý của các nhà khoa học, đồng nghiệp và bạn bè để hoàn thiện đề tài của mình.

Hà Nội, tháng 11 năm 2019

Nguyễn Tiến Lập

PHỤ LỤC

Sơ đồ kết nối các khối trong thiết kế phần mềm modul đóng khung E1



TÀI LIỆU THAM KHẢO

[1]. J. Axelson (2015), *Serial port complete com ports usb virtual com ports and ports for embedded systems complete guides series*, 5th ed., Lakeview Research, Madison, United States, 524 pages.

[2]. Bezerra, Eduardo, Lettnin, Djones Vinicius (2016), *Synthesizable VHDL Design for FPGAs*, Springer, 165 pages.

[3]. Pong P. Chu (2006), *RTL Hardware Design Using VHDL: Coding for Efficiency, Portability, and Scalability*, Wiley, New York, United States, 694 pages.

[4]. Pong P. Chu (2011), *Embedded SoPC Design with Nios II Processor and VHDL Examples*, Wiley, New Jersey, United States, 703 pages.

[5]. Pong P. Chu (2017), *FPGA prototyping by VHDL examples*, 2nd ed., Wiley, New York, United States, 468 pages.

[6]. Pong P. Chu (2017), *FPGA prototyping by VHDL examples: Xilinx MicroBlaze MCS SoC*, 2nd ed., Wiley, Hoboken, United States, 632 pages.

[7]. Grevelink, Evelyn (2017), *How to Program Your First FPGA Device*, Intel, pp. 1-20.

[8]. International Telecommunication Union (1998), *ITU-T Recommendation G.704: Synchronous frame structures used ad 1544, 6312, 2048, 8448 and 44736 kbit/s hierarchical levels*, 37 pages.

[9]. Ricardo Jasinski (2016), *Effective Coding with VHDL: Principles and Best Practice*, MIT Press Ltd, Mass., United States, 624 pages.

[10]. Volnei A. Pedroni (2010), *Circuit Design and Simulation with VHDL*, MIT Press Ltd, Mass., United States, 632 pages.

[11]. Charles L. Phillips, John Parr, Eve Riskin (2013), *Signals, Systems & Transforms*, Pearson Education, United States, 816 pages.

[12]. Blaine Readler (2014), *VHDL by Example*, Full ARC Press, United States, 120 pages.

[13]. Andrew Rushton (2011), *VHDL for Logic Synthesis*, Wiley, New York, United States, 484 pages.

[14]. John Tibbs (2015), *Pocket Guide to the World of E1*, Wavetek Wandel Goltermann, Devon, UK, 51 pages.

[15]. Cem Unsanlan, Bora Tar (2017), *Digital System Design with FPGA: Implementation Using Verilog and VHDL*, McGraw-Hill Education, OH, United States, 400 pages.

[16]. Roger Woods, John McAllister, Gaye Lightbody, Ying Yi (2017), *FPGA-based Implementation of Signal Processing Systems*, Wiley, Hoboken, United States, 356 pages.

[17]. Jordan Christma (2018), *Learn VHDL and FPGA Development***,** Available: https://software.intel.com/en-us/articles/how-to-program-your-first-fpga-device.