BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC SAO ĐỎ

ĐINH VĂN THẮNG

THIẾT KẾ HỆ THỐNG NHÚNG VỚI VI ĐIỀU KHIỂN LÕI MỀM VÀ HỆ ĐIỀU HÀNH TRÊN FPGA

LUẬN VĂN THẠC SĨ

CHUYÊN NGÀNH: KỸ THUẬT ĐIỆN TỬ

NGƯỜI HƯỚNG DẪN KHOA HỌC:

TS. HỒ KHÁNH LÂM

HẢI DƯƠNG – NĂM 2018

LỜI CAM ĐOAN

Tôi xin cam đoan kết quả đạt được trong luận văn là sản phẩm của riêng cá nhân, là kết quả của quá trình học tập và nghiên cứu khoa học độc lập. Trong toàn bộ nội dung của luận văn, những nội dung được trình bày hoặc là của cá nhân hoặc là được tổng hợp từ nhiều nguồn tài liệu. Tất cả các tài liệu tham khảo đều có xuất xứ rõ ràng và được trích dẫn hợp pháp. Các số liệu, kết quả nêu trong luận văn là trung thực và chưa từng được ai công bố trong bất kỳ luận văn nào khác.

Tôi xin hoàn toàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của mình.

Hải Dương, ngày 10 tháng 7 năm 2018

TÁC GIẢ

Đinh Văn Thắng

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

Từ viết tắt	Nghĩa tiếng anh
ADC	Analog-to-Digital Converter - (Bộ chuyển đổi kỹ thuật số)
ABEL	Advanced Boolean Equation Language - (Ngôn ngữ phương trình nâng cao)
ALU	Arithmetic Logic Unit - (Đơn vị số học Logic)
ASIC	Application Specific Integrated Circuit - (Ứng dụng mạch tích hợp cụ thể)
DAC	Digital-to-Analog Converter - (Chuyển đổi công nghệ kỹ thuật số)
DCM	Digital Clock Management - (Quản lý đồng hồ kỹ thuật số)
CLB	Configurable Logic Block - (Khối cấu hình Logic)
EDK	Embedded Development Kit - (Bộ phát triển nhúng)
FF	Flip-Flop - (Phần tử lưu trữ dữ liệu)
FG	Function generator - (Máy phát chức năng)
FPGA	Field programmable Gate Array - (Mảng lập trình tùy biến)
IF	Interface - (Giao diện)
LMB	Local Memory Bus - (Bộ nhớ cục bộ)
LUT	Look-Up Table - (Bảng tra cứu)
MAC	Multiply-accumulate circuits - (Mạch tích lũy nhân)
NOC	Network On Chip - (Mạng kết nối trên Chip)
OPB	On-Chip Peripheral Bus - (Thiết bị ngoại vi trên Chip)
PLD	Programmable Logic Device - (Thiết bị lập trình logic)
PLB	Processor Local Bus - (Thiết bị xử lý cục bộ)
RISC	Reduced Instruction Set Computer - (Máy tính đặt lệnh)
SoC	System on Chip - (Hệ thống trên Chip)
VHDL	Very High Speed Hardware Desription Luaguage - (Ngôn ngữ tả phần cứng tốc độ cao)
VHSIC	Very High Speed Integrated Circuits - (Mạch tích hợp tốc độ cao)

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

MỞ ĐẦU

1. Lý do chọn đề tài

Ngày nay, những thiết bị điện tử phức tạp dựa trên kỹ thuật vi xử lý được ứng dụng trong mọi lĩnh vực, đặc biệt, là các thiết bị điều khiển trong công nghiệp, trong mạng lưới điện, trong tự động hóa, trong viễn thông.

Thiết kế các thiết bị điện tử số theo cách truyền thống trở nên khó khăn khi công nghệ vi mạch có mức tích hợp rất lớn (VLSI), phức tạp, và tốc độ cao. Trong năm 1980, Bộ quốc phòng Mỹ (DoD) đã tài trợ dự án chương trình VHSIC (Very high Speed Integrated Circuit) để tạo ra ngôn ngữ mô tả phần cứng chuẩn hóa. Năm 1983, DoD thiết lập các yêu cầu cho ngôn ngữ mô tả phần cứng VHSIC, gọi là VHDL (Very High speed integrated circuit hardware Description Language). Theo các nguyên tắc của IEEE, cứ 5 năm thì một chuẩn phải được đề xuất lại và được tiếp nhận. Theo đó, chuẩn VHDL 1076-1993 ra đời.

Kể từ năm 1980, các nhà công nghệ vi mạch tích hợp hàng đầu thế giới đã đẩy mạnh quá trình nghiên cứu về công nghệ vi mạch tích hợp mảng cổng lập trình được theo trường FPGA (field programmable Gate Array) và nhanh chóng cho ra các thế hệ FPGA với số lượng cổng và tốc độ ngày càng cao. FPGA được thiết kế đầu tiên bởi Ross Freeman, người sáng lập công ty Xilinx vào năm 1984. Các FPGA hiện nay, có số lượng cổng logic (logic gate) đủ lớn để có thể kết hợp thành một hệ thống bao gồm lõi CPU, Bộ điều khiển bộ nhớ (Memory Controller), các ngoại vi như SPI, Timer, I2C, GPIO, PWM, Video/Audio Controller…, tương đương với các hệ thống trên chip SoC (System on Chip)hiện đại. Tuy vậy FPGA không đạt được mức độ tối ưu như ASIC, và hạn chế trong khả năng thực hiện những tác vụ đặc biệt phức tạp, lẫn tốc độ hoạt động. Nhưng bù lại, FPGA có thể được lập trình bằng các ngôn ngữ mô tả phần cứng HDL (Hardware Description Language) như VHDL, hay Verilog. HDL để tạo ra các thiết kế mạch số từ số lượng lớn cổng logic, và có thể cấu trúc lại mạch thiết kế khi đang sử dụng. Như vậy công đoạn thiết kế của FPGA đơn giản, chi phí giảm thiểu, rút ngắn thời gian đưa sản phẩm vào sử dụng. FPGA cũng rất phù hợp cho thiết kế thử nghiệm các hệ thống nhúng phức tạp và thông minh được ứng dụng trong nhiều lĩnh vực như tự động điều khiển, robot, điện tử dân dụng, viễn thông, các thiết bị di động, các phương tiện vận tải, thuật vi xử lý, các thiết bị thám mã, xử lý tín hiệu số, kiến trúc máy tính hiệu năng cao, v.v...

Hệ thống nhúng (Embedded system) là một thuật ngữ để chỉ một hệ thống có khả năng tự trị được nhúng vào trong một môi trường hay một hệ thống mẹ. Đó là các hệ thống tích hợp cả phần cứng và phần mềm phục vụ các bài toán chuyên dụng trong nhiều lĩnh vực công nghiệp, tự động hoá điều khiển, quan trắc và truyền tin. Đặc điểm của các hệ thống nhúng là hoạt động ổn định và có tính năng tự động hoá cao.

Hệ thống dựa trên công nghệ FPGA đã được ứng dụng nhiều trong nhiều lĩnh vực: tự động hóa, robot, điện tử dân dụng, điện tử y tế, v.v...bởi chúng dễ dàng cấu hình sửa đổi theo yêu cầu ứng dụng mà không phải thay thế linh kiện, thiết kế phần cứng, tiết kiệm chi phí thiết kế, chi phí điện năng khi khai thác.

Do đó, tìm hiểu phương pháp thiết kế hệ nhúng trên FPGA sử dụng ngôn ngữ lập trình VHDL là lý do mà học viện chọn đề tài "Thiết kế hệ thống nhúng với vi điều khiển lõi mềm và hệ điều hành trên FPGA".

1.2. Tính cấp thiết của đề tài

Lõi mềm vi xử lý, hay vi điều khiển 32-bit khác với chip vi mạch vi xử lý hay vi điều khiển 32-bit (lõi cứng):

- Lõi mềm có nghĩa là không cố định, có thể bằng lập trình cấu hình lại cấu trúc hay sửa đổi chức năng của vi xử lý (vi điều khiển) tùy ý phụ thuộc vào nhu cầu ứng dụng mong muốn.

- Lõi cứng, chỉ có thể sử dụng sẵn chip vi xử lý (vi điều khiển) do nhà công nghệ cung cấp, người dùng chỉ còn cách hiểu biết qua các mô tả của nhà công nghệ (đặc tính kỹ thuật, tập lệnh) để thiết kế các hệ thống lớn hơn.

- Hệ thống nhúng trên FPGA và hệ điều hành nhúng đang là một trong những thiết kế thông dụng hiện nay.

Do tính ứng dụng rộng rãi trong nhiều lĩnh vực, mà FPGA và các ngôn ngữ HDL trở nên cấp thiết trong đầu tư ứng dụng và đào tạo.

2. MỤC TIÊU VÀ PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Mục tiêu của đề tài

- Tìm hiểu một trong ngôn ngôn ngữ mô tả phần cứng là VHDL
- Tìm hiểu công nghệ FPGA
- Tìm hiểu vi điều khiển lõi mềm 32-bit kiến trúc tập lệnh giảm thiểu (RISC)

- Thiết kế được vi xử lý lõi mềm bằng công cụ phần mềm thiết kế dựa vào HDL
- Tìm hiểu hệ điều hành nhúng và cài đặt trên hệ vi xử lý lõi mềm trên FPGA

2.2. Nội dung nghiên cứu

- Công nghệ FPGA
- Ngôn ngữ lập trình VHDL
- Vi điều khiển Nios II 32-bit kiến trúc tập lệnh rút gọn RISC
- Công cụ phần mềm phát triển Altera dùng cho thiết kế các hệ thốngsố trên Altera FPGA
- Các bước thiết kế vi điều khiển Nios II 32-bit nhờ sử dụng Altera Quartus II
- Các bước cài đặt hệ điều hành nhúng uClinux

2.3. Phương pháp luận và phương pháp nghiên cứu

2.3.1. Phương pháp luận

Dựa vào các phương pháp chuyên môn:

- Kỹ thuật điện tử và điện tử số
- Kỹ thuật vi xử lý
- Ngôn ngữ lập trình
- Thiết kế các hệ thống số bằng ngôn ngữ mô tả phần cứng

2.3.2. Phương pháp nghiên cứu

- Khảo sát và đánhgiá các công trình nghiên cứu, các tài liệu kỹ thuậtliên quan với đề tài
- Lựa chọn công nghệ FPGA của nhà công nghệ Altera cho đề tài
- Lựa chọn ngôn ngữ lập trình thiết kế hệ thống số (VHDL, Verrilog) phù hợp cho đề tài
- Trên cơ sở các mục tiêu của đề tài xây dựng kế hoạch thực hiện đề tài củaluận văn, đánh giá kết quả thực hiện
- Dựa trên các yêu cầu và đánh giá của giáo viên hướng dẫn, thực hiệncác chỉnh sửa, và hoàn chỉnh luận văn.

3. BỐ CỤC CỦA LUẬN VĂN

Nội dung luận văn gồm cóba chương và kết luận như sau:

Chương 1: Tổng quan hệ thống nhúng

Chương 2: Hệ thống nhúng trên FPGA

Chương 3: Thiết kế hệ vi điều khiển lõi mềm Nios II 32-bit và cài đặt uClinux

Kết luận và định hướng nghiên cứu

CHƯƠNG 1: TỔNG QUAN HỆ THỐNG NHÚNG

1.1. CÁC KHÁI NIỆM HỆ THỐNG NHÚNG

1.1.1. Định nghĩa hệ thống nhúng

Hệ thống nhúng (Embedded system) [10] là một thuật ngữ để chỉ một hệ thống có khả năng tự trị được nhúng vào trong một môi trường hay một hệ thống mẹ. Đó là các hệ thống tích hợp cả phần cứng và phần mềm phục vụ các bài toán chuyên dụng trong nhiều lĩnh vực công nghiệp, tự động hoá điều khiển, quan trắc và truyền tin. Đặc điểm của các hệ thống nhúng là hoạt động ổn định và có tính năng tự động hoá cao.

Hệ thống nhúng thường được thiết kế để thực hiện một chức năng chuyên biệt nào đó. Khác với các máy tính đa chức năng, chẳng hạn như máy tính cá nhân, một hệ thống nhúng chỉ thực hiện một hoặc một vài chức năng nhất định, thường đi kèm với những yêu cầu cụ thể và bao gồm một số thiết bị máy móc và phần cứng chuyên dụng mà ta không tìm thấy trong một máy tính đa năng nói chung. Vì hệ thống chỉ được xây dựng cho một số nhiệm vụ nhất định nên các nhà thiết kế có thể tối ưu hóa nó nhằm giảm thiểu kích thước và chi phí sản xuất. Các hệ thống nhúng thường được sản xuất hàng loạt với số lượng lớn. Hệ thống nhúng rất đa dạng, phong phú về chủng loại. Đó có thể là những thiết bị cầm tay nhỏ gọn như đồng hồ kĩ thuật số và máy chơi nhạc MP3, hoặc những sản phẩm lớn như đèn giao thông, bộ kiểm soát trong nhà máy hoặc hệ thống kiểm soát các máy năng lượng hạt nhân. Xét về độ phức tạp, hệ thống nhúng có thể rất đơn giản với một vi điều khiển hoặc rất phức tạp với nhiều đơn vị, các thiết bị ngoại vi và mạng lưới được nằm gọn trong một lớp vỏ máy lớn.

1.1.2. Các đặc điểm của hệ thống nhúng

Hệ thống nhúng thường có một số đặc điểm chung như sau:

- Các hệ thống nhúng được thiết kế để thực hiện một số nhiệm vụ chuyên dụng chứ không phải đóng vai trò là các hệ thống máy tính đa chức năng. Một số hệ thống đòi hỏi ràng buộc về tính hoạt động thời gian thực để đảm bảo độ an toàn và tính ứng dụng; một số hệ thống không đòi hỏi hoặc ràng buộc chặt chẽ, cho phép đơn giản hóa hệ thống phần cứng để giảm thiểu chi phí sản xuất.

- Một hệ thống nhúng thường không phải là một khối riêng biệt mà là một hệ thống phức tạp nằm trong thiết bị mà nó điều khiển.

- Phần mềm được viết cho các hệ thống nhúng được gọi là firmware và được lưu trữ trong các chip bộ nhớ ROM hoặc bộ nhớ flash chứ không phải là trong một ổ đĩa. Phần mềm thường chạy với số tài nguyên phần cứng hạn chế: không có bàn phím, màn hình hoặc có nhưng với kích thước nhỏ.

1.1.3. Giao diện và các thiết bị ngoại vi của hệ thống nhúng

1) Giao diện:

Các hệ thống nhúng có thể không có giao diện (đối với những hệ thống đơn nhiệm) hoặc có đầy đủ giao diện giao tiếp với người dùng tương tự như các hệ điều hành trong các thiết bị để bàn. Đối với các hệ thống đơn giản, thiết bị nhúng sử dụng nút bấm, đèn LED và hiển thị chữ cỡ nhỏ hoặc chỉ hiển thị số, thường đi kèm với một hệ thống menu đơn giản. Trong một hệ thống phức tạp hơn, một màn hình đồ họa, cảm ứng hoặc có các nút bấm ở lề màn hình cho phép thực hiện các thao tác phức tạp mà tối thiểu hóa được khoảng không gian cần sử dụng; ý nghĩa của các nút bấm có thể thay đổi theo màn hình và các lựa chọn. Các hệ thống nhúng thường có một màn hình với một nút bấm dạng cần điều khiển (joystick button). Sự phát triển mạnh mẽ của mạng toàn cầu đã mang đến cho những nhà thiết kế hệ nhúng một lựa chọn mới là sử dụng một giao diện web thông qua việc kết nối mạng. Điều này có thể giúp tránh được chi phí cho những màn hình phức tạp nhưng đồng thời vẫn cung cấp khả năng hiển thị và nhập liệu phức tạp khi cần đến, thông qua một máy tính khác. Điều này là hết sức hữu dụng đối với các thiết bị điều khiển từ xa, cài đặt vĩnh viễn. Ví dụ, các router là các thiết bị đã ứng dụng tiện ích này.

2) Thiết bị ngoại vi:

Hệ thống nhúng giao tiếp với bên ngoài thông qua các thiết bị ngoại vi như:

- Serial Communication Interfaces (SCI): RS-232, RS-422, RS-485...
- Synchronous Serial Communication Interface: I2C, JTAG, SPI, SSC và ESSI
- Universal Serial Bus (USB)
- Networks: Controller Area Network, LonWorks...
- Bộ định thời: PLL(s), Capture/Compare và Time Processing Units
- Discrete IO: General Purpose Input/Output (GPIO)

1.1.4. Kiến trúc CPU của hệ thống nhúng

Các bộ xử lý trong hệ thống nhúng có thể được chia thành hai loại: vi xử lý và vi điều khiển. Các vi điều khiển thường có các thiết bị ngoại vi được tích hợp trên chip nhằm giảm kích thước của hệ thống. Có rất nhiều loại kiến trúc CPU được sử dụng trong thiết kế hệ nhúng như ARM, MIPS, Coldfire/68k, PowerPC, x86, PIC, 8051, Atmel AVR, Renesas H8, SH, V850, FR-V, M32R, Z80, Z8 … Điều này trái ngược với các loại máy tính để bàn, thường bị hạn chế với một vài kiến trúc máy tính nhất định. Các hệ thống nhúng có kích thước nhỏ và được thiết kế để hoạt động trong môi trường công nghiệp thường lựa chọn PC/104 và PC/104++ làm nền tảng. Những hệ thống này thường sử dụng DOS, Linux, NetBSD hoặc các hệ điều hành nhúng thời gian thực như QNX hay VxWorks. Có các hệ thống nhúng có kích thước rất lớn thường sử dụng một cấu hình thông dụng là hệ thống on chip (System on a chip – SoC), hay là một bảng mạch tích hợp cho một ứng dụng cụ thể (an application-specific integrated circuit – ASIC). Hệ thống nhúng có thể được thiết kế trên các bảng mạch phát triển trên FPGA. Trong đó bằng lập trình ở ngôn ngữ mô phỏng phần cứng có thể tạo ra lõi xử lý mềm trên chip FPGA mong muốn.

1.1.5. Độ tin cậy và an toàn của hệ thống nhúng

Các hệ thống nhúng thường nằm trong các cỗ máy được kỳ vọng là sẽ chạy hàng năm trời liên tục mà không bị lỗi hoặc có thể khôi phục hệ thống khi gặp lỗi. Vì thế, các phần mềm hệ thống nhúng được phát triển và kiểm thử một cách cẩn thận hơn là phần mềm cho máy tính cá nhân. Ngoài ra, các thiết bị rời không đáng tin cậy như ổ đĩa, công tắc hoặc nút bấm thường bị hạn chế sử dụng. Việc khôi phục hệ thống khi gặp lỗi có thể được thực hiện bằng cách sử dụng các kỹ thuật như watchdog timer – nếu phần mềm không đều đặn nhận được các tín hiệu watchdog định kì thì hệ thống sẽ bị khởi động lại.

Một số vấn đề cụ thể về độ tin cậy như:

- Hệ thống không thể ngừng để sửa chữa một cách an toàn, ví dụ như ở các hệ thống không gian, hệ thống dây cáp dưới đáy biển, các đèn hiệu dẫn đường,… Giải pháp đưa ra là chuyển sang sử dụng các hệ thống con dự trữ hoặc các phần mềm cung cấp một phần chức năng.

- Hệ thống phải được chạy liên tục vì tính an toàn, ví dụ như các thiết bị dẫn đường máy bay, thiết bị kiểm soát độ an toàn trong các nhà máy hóa chất,… Giải pháp đưa ra là lựa chọn backup hệ thống.

- Nếu hệ thống ngừng hoạt động sẽ gây tổn thất rất nhiều tiền của ví dụ như các dịch vụ buôn bán tự động, hệ thống chuyển tiền, hệ thống kiểm soát ở các nhà máy …

1.1.6. Các công cụ phát triển hệ thống nhúng

Tương tự như các sản phẩm phần mềm khác, phần mềm hệ thống nhúng cũng được phát triển nhờ việc sử dụng các trình biên dịch (compilers), chương trình dịch hợp ngữ (assembler) hoặc các công cụ gỡ rối (debuggers). Tuy nhiên, các nhà thiết kế hệ thống nhúng có thể sử dụng một số công cụ chuyên dụng như:

- Bộ gỡ rối mạch hoặc các chương trình mô phỏng (emulator)

- Tiện ích để thêm các giá trị checksum hoặc CRC vào chương trình, giúp hệ thống nhúng có thể kiểm tra tính hợp lệ của chương trình đó.

- Đối với các hệ thống xử lý tín hiệu số, nhà phát triển hệ thống có thể sử dụng phần mềm workbench như MathCad hoặc Mathematica để mô phỏng phép toán.

- Các trình biên dịch và trình liên kết (linker) chuyên dụng được sử dụng để tối ưu hóa một thiết bị phần cứng.

- Một hệ thống nhúng có thể có ngôn ngữ lập trình và công cụ thiết kế riêng của nó hoặc sử dụng và cải tiến từ một ngôn ngữ đã có sẵn.

Các công cụ phần mềm có thể được tạo ra bởi các công ty phần mềm chuyên dụng về hệ thống nhúng hoặc chuyển đổi từ các công cụ phát triển phần mềm GNU. Đôi khi, các công cụ phát triển dành cho máy tính cá nhân cũng được sử dụng nếu bộ xử lý của hệ thống nhúng đó gần giống với bộ xử lý của một máy PC thông dụng.

1.1.7. Các hệ điều hành nhúng

Hệ thống nhúng có thể có hệ điều hành hoặc không. Một số loại kiến trúc phần mềm thông dụng trong các hệ thống nhúng như sau:

1) Vòng lặp kiểm soát đơn giản:

Theo thiết kế này, phần mềm được tổ chức thành một vòng lặp đơn giản. Vòng lặp gọi đến các chương trình con, mỗi chương trình con quản lý một phần của hệ thống phần cứng hoặc phần mềm.

2) Hệ thống ngắt điều khiển:

Các hệ thống nhúng thường được điều khiển bằng các ngắt. Có nghĩa là các tác vụ của hệ thống nhúng được kích hoạt bởi các loại sự kiện khác nhau. Ví dụ, một ngắt có thể được sinh ra bởi một bộ định thời sau một chu kỳ được định nghĩa trước, hoặc bởi sự kiện khi cổng nối tiếp nhận được một byte nào đó. Loại kiến trúc này thường được sử dụng trong các hệ thống có bộ quản lý sự kiện đơn giản, ngắn gọn và cần độ trễ thấp. Hệ thống này thường thực hiện một tác vụ đơn giản trong một vòng lặp chính. Đôi khi, các tác vụ phức tạp hơn sẽ được thêm vào một cấu trúc hàng đợi trong bộ quản lý ngắt để được vòng lặp xử lý sau đó. Lúc này, hệ thống gần giống với kiểu nhân đa nhiệm với các tiến trình rời rạc.

3) Đa nhiệm tương tác:

Một hệ thống đa nhiệm không ưu tiên cũng gần giống với kỹ thuật vòng lặp kiểm soát đơn giản ngoại trừ việc vòng lặp này được ẩn giấu thông qua một giao diện lập trình API. Các nhà lập trình định nghĩa một loạt các nhiệm vụ, mỗi nhiệm vụ chạy trong một môi trường riêng của nó. Khi không cần thực hiện nhiệm vụ đó thì gọi đến các tiến trình con tạm nghỉ (bằng cách gọi "pause", "wait", "yield" …) Ưu điểm và nhược điểm của loại kiến trúc này cũng giống với kiểm vòng lặp kiểm soát đơn giản. Tuy nhiên, việc thêm một phần mềm mới được thực hiện dễ dàng hơn bằng cách lập trình một tác vụ mới hoặc thêm vào hàng đợi thông dịch (queue-interpreter).

4) Đa nhiệm ưu tiên:

Ở loại kiến trúc này, hệ thống thường có một đoạn mã ở mức thấp thực hiện việc chuyển đổi giữa các tác vụ khác nhau thông qua một bộ định thời. Đoạn mã này thường nằm ở mức mà hệ thống được coi là có một hệ điều hành và vì thế cũng gặp phải tất cả những phức tạp trong việc quản lý đa nhiệm. Bất kỳ tác vụ nào có thể phá hủy dữ liệu của một tác vụ khác đều cần phải được tách biệt một cách chính xác. Việc truy cập tới các dữ liệu chia sẻ có thể được quản lý bằng một số kỹ thuật đồng bộ hóa như hàng đợi thông điệp (message queues), semaphores … Vì những phức tạp nói trên nên một giải pháp thường được đưa ra đó là sử dụng một hệ điều hành thời gian thực. Lúc đó, các nhà lập trình có thể tập trung vào việc phát triển các chức năng của thiết bị chứ không cần quan tâm đến các dịch vụ của hệ điều hành nữa.

5) Embedded Linux:

Embedded Linux là một hệ điều hành linux nhúng hoàn chỉnh được sử dụng cho một thiết bị, bao gồm Linux kernel và các ứng dụng kèm theo (kết hợp này gọi là một distributio). Cụm từ "embedded" thường được đề cập trong kernel nhưng thật chất không có một phiên bản Linux kernel nào dành riêng cho hệ thống nhúng. Linux kernel source code được sử dụng chung để compile cho mọi thiết bị, từ các thiết bị nhúng, đến máy PC và cả các server lớn, đối với mỗi platform sẽ có những option hiệu chỉnh phù hợp và đặc biệt dành cho platform này.

Các Embedded Linux distributions có thể mua được từ các nhà sản xuất (Monta Vista, Wind River System...),các distribution này đã được phát triển hoàn chỉnh có thể cài đặt như cài đặt một hệ điều hành cho một máy PC thông thường, đi kèm với distribution là các công cụ phát triển (toolchain, debugger, project management software và image builder...). Tuy nhiên, việc xây dựng một hệ thống Embedded Linux từ các phần từ rời rạc ban đầu sẽ giúp chúng ta có sự thấu hiểu sâu về hoạt động của hệ thống cũng như không phải tốn chi phí chi trả cho nhà cung cấp. Các thành phần cấu tạo nên hệ thống Embedded Linux bao gồm boot loader, Linux kernel, các ứng dụng. Tất cả các thành phần này đều có thể tìm thấy phiên bản open source và chúng ta có thể tự mình chỉnh sửa, thay đổi cho phù hợp với thiết bị của mình.

Embedded Linux không thể chạy trên các processor có kiến trúc nhỏ hơn 32-bit. Tuy nhiên trong thời gian gần đây, công nghệ system-on-chip (SOC) phát triển mạnh, dẫn đến việc hạ giá thành sản xuất các microprocessor, đồng thời bộ nhớ RAM và flash cũng rẻ và có dung lượng lớn hơn tạo nên thuận lợi cho việc chuyển sang phát triển hệ thống nhúng có sử dụng Embedded Linux. Các hệ thống nhúng ngày nay bên cạnh các chức năng cần thiết còn có thể hỗ trợ thêm các chức năng phụ (web server, firewall, nghe nhạc...) thông qua hệ thống Embedded Linux.

Việc sử dụng Embedded Linux cho hệ thống nhúng còn giúp giảm thời gian thiết kế và phát triển, do bản thân Linux kernel được thiết kế theo module. Chúng ta có thể dễ dàng tìm được nhiều module có sẵn và hiệu quả như TCP/IP stack, X-server cho ứng dụng GUI, hoặc có thể tìm thấy driver cho thiết bị nhúng của mình đã được viết sẵn trong kernel.

Một điểm mạnh khác của Embedded Linux là open source, điều này cho phép người thiết kế can thiệp sâu hơn vào các dịch vụ và module mà hệ điều hành cung cấp. Người thiết kế có thể hiểu rõ hơn về những hàm mà họ gọi và thậm chí có thể thay đổi, tối ưu các hàm này cho thiết bị mình sử dụng. Người thiết kế còn có thể dựa vào các module driver có sẵn để tham khảo cho các driver mà họ sắp viết. Tính open source còn giúp code hỗ trợ bởi kernel có tính tin cậy cao, trước khi được đưa vào kernel source tree, code này đã được test rất nhiều trong cộng đồng và thậm chí nếu có lỗi xảy ra cũng sẽ có patch thay thế trong thời gian ngắn.

Tính sẵn sàng của Embedded Linux là rất cao. Ít có hệ điều hành nào hỗ trợ được nhiều platform và device driver như Linux. Ngoài hỗ trợ phần cứng Linux còn hỗ trợ các giao thức tiêu chuẩn (wifi, bluetooth, ...).

Tất cả những điều trên cho chúng ta thấy tính năng hiệu quả của hệ thống Embedded Linux và cũng những điểm mạnh đó làm cho xu hướng phát triển Embedded Linux ngày càng trở nên quan trọng, càng nhiều công ty đầu tư vào lĩnh vực này và càng nhiều người yêu thích Embedded Linux hơn.

1.2. CÔNG NGHỆ FPGA

1.2.1. Cấu trúc FPGA

Có một xu hướng phát triển khác dựa trên công nghệ mảng cổng, đó là mảng cổng có thể lập trình được dạng trường, FPGA (Field-Programmable Gate Array). Từ 1980, các công ty sản xuất PLD hàng đầu đã đẩy mạnh quá trình nghiên cứu về FPGA và nhanh chóng cho ra các thế hệ FPGA với số lượng cổng và tốc độ ngày càng cao.các FPGA hiện nay có số lượng cổng đủ lớn để có thể thay thế cả một hệ thống bao gồm lõi CPU, Bộ điều khiển bộ nhớ (Memory Controller), các ngoại vi như SPI,Timer, I2C, GPIO, PWM, Video/Audio Controller… (nghĩa là tương đương với các SoC hiện đại).

FPGA gồm có(hình 1.1):

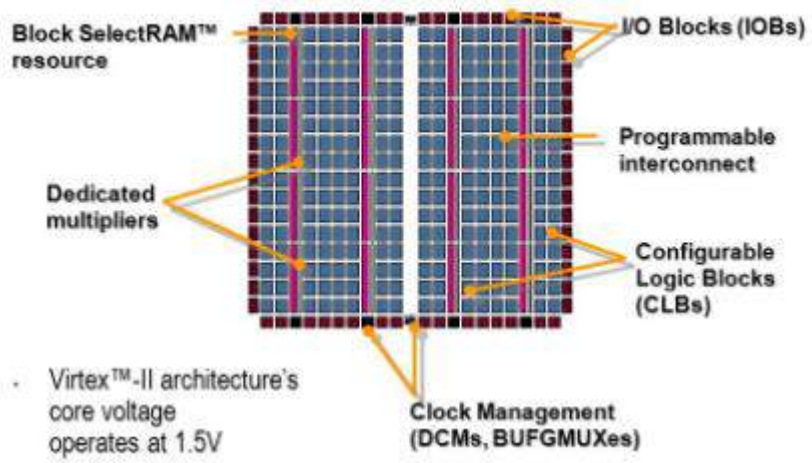
- CLBs (configurable Logic Blocks): các khối logic có thể cấu hình được, là các thành phần tiêu chuẩn. Trong hầu hết các FPGA, mỗi một CLB chứa một số các mảnh, mà mỗi mảnh lại chứa một số (thường là 2 hoặc 4) ô logic (logic cell) với một số thành phần nhớ (Flip-Flop) hoặc bộ dồn kênh (Mux) nếu không dùng FF. Mỗi ô logic có thể được cấu hình để thực hiện các chức năng logic cơ bản (như AND, OR,

NOT) trên các tín hiệu số nhờ sử dụng bảng LUT (look-up Table). Các CLB liên kết với nhau qua mạng liên kết có thể lập trình được (Programmable Interconnect hay routing).

- Interconnect hay Routing: mạng liên kết hay định tuyến, là các ma trận chuyển mạch có thể lập trình được - PSM (Programmable Switch Matrix) để hình thành các đơn vị thực hiện các chức năng phức tạp hơn.

- IOBs (Input/Output Blocks): các khối vào/ra nằm bao xung quanh của miếng FPGA và nối với các chân tín hiệu vào/ra (I/O pin). Như vậy từng chân I/O của FPGA có thể được lập trình để đảm bảo các giao tiếp điện cần thiết cho kết nối FPGA với hệ thống mà nó là thành phần.

- Block RAM: khối RAM, là các băng nhớ bên trong FPGA.



Hình 1.1: Cấu trúc của FPGA

Ngoài ra các thành phần trên, FPGA còn các logic nhỏ khác, như:

MAC (Multiply-accumulate circuits): các khối logic nhân tích lũy, để thực hiện các phép nhân và cộng hiệu quả.

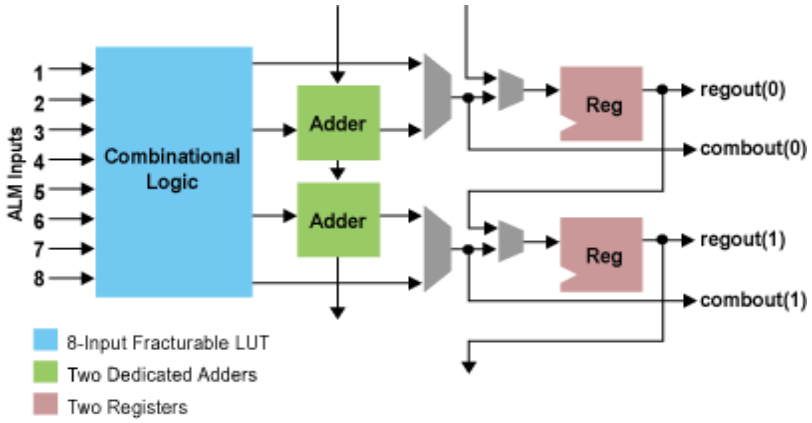
Các khối thực hiện các chức năng đặc biệt: xử lý tín hiệu số và tương tự, ví dụ các bộ biến đổi tương tự-số ADC (Analog-to-Digital Converter) và các bộ biến đổi số-tương tự DAC (Digital-to-Analog Converter), cho phép FPGA vận hành như là một SoC. Một FPGA chứa từ 64 đến hàng chục ngàn khối logic và các flip-flop.

LUT giống như một RAM nhỏ, cũng được gọi là các bộ tạo chức năng, FG (Function generator), được sử dụng để thực hiện các chức năng logic nhờ cất giữ trạng thái logic ra đúng ở trong một vùng nhớ, mà trạng thái logic ra tương ứng với từng tổ hợp của các biến vào. LUT thường có 4 đầu vào có thể thực hiện bất kỳ chức năng logic 4-đầu vào.

Các thành phần nhớ trong CLB có thể được cấu hình hoặc thành các các Flip-flop hoặc thành các mạch chốt (Latch). Mỗi CLB thường có bố trí các chân tín hiệu vào và hai chân tín hiệu ra, như chỉ ra ở hình 1.1. Như vậy, có thể kết nối chân tín hiệu đầu từ các phía tương ứng của CLB, trong khi chân tín hiệu đầu ra có thể kết nối với các dây dẫn định tuyến ở cả kênh phía bên phải và kênh phía dưới của CLB. Mỗi một chân tín hiệu đầu ra của CLB có thể kết nối với bất kỳ đoạn nối dây nào trong các kênh kề cận với nó. Tương tự, hộp đệm vào/ra (I/Opad) ở đỉnh của chip FPGA có thể nối với bất kỳ dây dẫn nào ở kênh nằm ngang phía dưới.

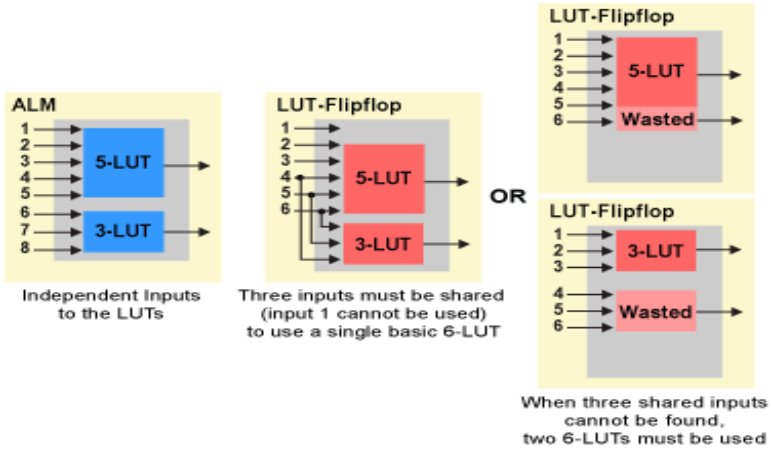
Các nhà sản xuất có xu hướng thiết kế các khối logic của FPGA thực hiện các chức năng lớn hơn để giảm liên kết cục bộ, đồng nghĩa với số lượng chân tín hiệu đầu vào của khối logic tăng lên, và nó cũng cho phép lập trình các khối logic linh hoạt hơn.

Kiến trúc của họ Altera Stratix FPGA đạt được hiệu năng cao nhờ đưa vào module logic thích ứng hiệu quả vùng - ALM (Adaptive logic Module). ALM gồm có logic tổ hợp, 2 thanh ghi, và 2 bộ cộng, như chỉ ra ở hình 1.2. logic tổ hợp có 8 đầu vào một bảng LUT (Lookup Table).



Hình 1.2: Altera Stratix IV FPGA ALM kiến trúc của Altera ALM

Cả LUT-6 đầu vào và ALM đều là những logic cơ bản xây dựng các khối của các kiến trúc FPGA và chúng tương đồng nhau (hình 1.3).



Hình 1.3: Sự thực hiện chức năng 5- đầu vào và
3- đầu vào trong Stratix IV ALM và Virtex-5 LUT-cặp FF

Các FPGA khác nhau có số lượng các ô logic, kích cỡ và số lượng các block RAM, các MAC khác nhau. Các FPGA sử dụng trong các hệ thống lai (hybrid system) thường có khoảng 100K-200K ô logic, 500KB của RAM bên trong và 100 MACs. Hệ thống lai có thể sử dụng FPGA với 1000 khối I/O tương ứng với 1000 I/O pin để đảm bảo các giao tiếp với hệ thống chủ, cũng như với bộ nhớ cục bộ nối trực với FPGA.

Các FPGA thường được lập trình sau khi đã hàn gắn trên bảng mạch in, tương tự như các CPLD lớn. Nhưng dữ liệu cấu hình trong FPGA bị mất khi ngừng cấp nguồn (mất điện) giống như RAM trong máy tính vậy. Do đó, muỗi lần ngắt nguồn và bật lại thì ta phải nạp lại tệp cấu hình vào FPGA. Muốn lưu giữ lại cấu hình đã lập trình cho FPGA thì ta phải mắc thêm PROM hay EPROM ngoài. Bộ nhớ ngoài này có nhiệm vụ lưu tệp cấu hình ở dạng nhị phân (bitstream hay bit file) và tự động nạp dữ liệu cấu hình lại cho FPGA mỗi khi bật nguồn, như vậy dù có ngắt nguồn FPGA vẫn "không bị mất" dữ liệu. Các phiên bản EEPROM có thể có thể lập trình được trong hệ thống (hay trong mạch), thường thông qua giao tiếp JTAG. Tệp cấu hình chứa các thiết lập cho từng CLB, PSM, MAC, I/O và các thành phần có thể cấu hình khác của FPGA. Các FPGA được sử dụng trong các hệ thống máy tính lai có thể được lập trình lại vô số lần. Thời gian tải cấu hình mới thường chỉ chưa đến 1 giây. Một số FPGA hiện nay

có khả năng trong khi đang hoạt động chuyển đến cấu hình mới đã được nạp trước vào thiết bị. Một số FPGA cũng cho phép cầu hình lại từng phần của thiết bị.

FPGA và CPLD có những điểm khác biệt đó là: FPGA bên trong dựa trên các bảng look-up (LUTs), trong khi các CPLD hình thành các chức năng logic bằng các nhiều mạch cổng (ví dụ tổng các tích); FPGA và CPLD đều cấu tạo từ các khối logic (các ô logic) là sự kết hợp của một khối logic và Flip-Flop. Nhưng, FPGA có số lượng lớn các khối logic (đến hàng trăm ngàn) hơn nhiều so với CPLD; FPGA giống như RAM, phải nạp lại dữ liệu cấu hình mỗi khi bật nguồn. CPLD giống như EEPROM chỉ cần nạp một lần và không bị mất chức năng sau khi ngắt nguồn;

Do FPGA có số lượng rất lớn các khối logic nên có nhiều tài nguyên để thực hiện nhiều chức năng toán học chuyên dụng và phức tạp.Vì vậy các FPGA phù hợp cho các thiết kế phức tạp hơn so với CPLD. Nhìn chung các CPLD là sự lựa chọn tốt cho các ứng dụng tổ hợp, trong khi các FPGA phù hợp hơn cho các máy trạng thái lớn (như các vi xử lý).

FPGA có các phần tử logic chạy theo dạng song song. Còn vi điều khiển dựa trên cấu trúc CPU thực thi theo mã lệnh theo dạng tuần tự.

FPGA dùng ngôn ngữ lập trình phần cứng (Verilog, VHDL) và lập trình trên FPGA gọi là lập trình phần cứng. Lập trình vi điều khiển là lập trình phần mềm phần cứng có sẵn.

1.2.2. Các kiến trúc của FPGA

Có hai loại kiến trúc cơ bản của FPGA: kiến trúc mặt độ thưa (Coarse-grained) và kiến trúc mật độ cao (fine-grained).

1) Kiến trúc mật độ thưa:

Kiến trúc nàyco các khối logic lớn, mỗi khối logic thường chứa hai hoặc nhiều bảng look-up (LUTs) và hai hoặc nhiều flip-flop. Trong FPGA kiến trúc này bảng LUT 4-đầu vào (như là 16x1 ROM) làm thành một logic cụ thể. Loại kiến trúc này sử dụng công nghệ cầu chì đối ngẫu CMOS (anti-fuse CMOS), chỉ cho phép lập trình một lần, nhưng dữ liệu không bị thay đổi khi bị mất nguồn. để lập trình cần phải có thiết bị lập trình chuyên dụng (do nhà sản xuất hay nhà phân phối cung cấp).

2) Kiến trúc mật độ cao:

Kiến trúc nàycó số lượng lớn các khối logic đơn giản. Khối logic đơn giản hoặc chứa chức năng logic hai đầu vào hoặc bộ dồn kênh 4-to-1 và một flip-flop. Chúng sử dụng công nghệ bộ nhớ SRAM, tương tự như các bộ vi xử lý. Như vậy chúng có thể được lập trình lại không hạn chế trong hệ thống, nhưng đòi hỏi phải có bộ nhớ PROM.EPROM, EEPROM hay Flash bên ngoài (gọi là bộ nhớ cấu hình) để lưu trữ chương trình xác định các chức năng như thế nào của từng khối logic, các khối I/O nào là các cổng vào và các cổng ra, và các khối được liên kết với nhau như thế nào. FPGA hoặc là tự nạp bộ nhớ cấu hình của nó hoặc bộ xử lý bên ngoài tải nội dung của bộ nhớ cấu hình vào FPGA. Khi thực hiện tự nạp, FPGA địa chỉ các byte của bộ nhớ cấu hình giống như bộ xử lý địa chỉ bộ nhớ PROM lưu cấu hình khởi tạo (boot PROM), hoặc sử dụng PROM tuần tự truy nhập liên tiếp. Khi bộ xử lý tải vào FPGA, FPGA thể hiện như là bộ xử lý ngoại vi chuẩn. Thời gian cấu hình thường nhỏ hơn 200 ms, phụ thuộc vào kích thước của FPGA và phương pháp cấu hình.

Bảng 1.1: Các kiến trúc FPGA, công nghệ và các nhà cung cấp

Kiến trúc	Static Memory	Anti-Fuse	Flash
Mật độ thưa (Coarse-grained)	**Altera**: (FLEX, APEX) **Atmel**: (AT40K) **DynaChip** **Lucent**: (ORCA) **Vantis**: (VF1) **Xilinx**: (XC3000,XC4000xx,Spartan,Virtex)	QuickLogic: (pASIC)	.
Mật độ cao (Fine-grained)	**Actel**: (SPGA) **Atmel**: (AT6000)	**Actel**: (ACT)	Gatefield

1.2.3. So sánh FPGA với các nghệ IC khác

1) FPGA và ASIC:

FPGA chiếm 15% của công nghiệp ASIC về khối lượng và doanh thu. Các ASIC có hiệu năng cao hơn, dung lượng cao hơn, nguồn nuôi thấp hơn, tích hợp các tín hiệu, và hiệu quả cao hơn so với các FPGA. Các thiết kế của FPGA tiêu thụ nhiều nguồn hơn so với các ASIC. Nếu sản xuất công nghiệp với số lượng lớn, thì ASIC cho chi phí trên một đơn vị thấp hơn các FPGA. Nếu ta biết rằng có nhiều thiết bị và các máy tính cá nhân chạy ở tốc độ vài Gigaherz, thì các FPGA lại chạy với tốc độ thấp ở (vài

trăm Megaherz). Trong khi đó các ASIC có thể tốc độ cao hơn FPGA nhiều.FPGA hiệu quả cho các ứng dụng nhỏ bởi vì nó có chi phí thiết kế thấp. Nhưng nếu sản xuất công nghiệp với số lượng lớn, thì ASIC lại có giá rẻ hơn nhiều. Quá trình thiết kế FPGA đơn giản và thời gian ngắn hơn so với quá trình thiết kế ASIC, bởi vì không cần phải sắp xếp linh kiện, không cần các mặt mạ hoặc các quá trình sau-cuối (back-end processes). ASIC có thể có các thiết kế xử lý hỗn hợp các tín hiệu, hoặc chỉ là các thiết kế tương tự. Nhưng không thể thiết kế ASIC sử dụng các chip FPGA.

ASIC có thể có các thiết kế hoàn toàn cho các ứng dụng riêng và phức tạp, ví dụ vi xử lý chẳng hạn, nhưng FPGA thì không thể.Bởi vì FPGA có thể được lập trình cấu hình lại vô số lần nên nó phù hợp cho các thiết kế mẫu cho ASIC. Như vậy với FPGA ta có thể tự thiết kế cả mẫu CPU theo mong muốn.Các thiết kế ASIC phải tốn chi phí NRE (Non Recurring Engineering), đó là chi phí cho một lần nghiên cứu, thiết kế, và kiểm thử sản phẩm mới, trong khi đó thì các thiết kế FPGA lại không cần.Các công cụ được sử dụng cho thiết kế FPGA thường rẻ hơn so với các công cụ thiết kế của ASIC.Một FPGA có thể được sử dụng cho các ứng dụng khác nhau, nhờ lập trình lại FPGA. Nhưng với ASIC thì không thể.

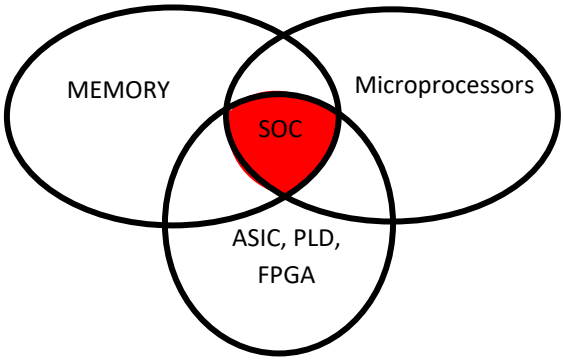
2) FPGA và CPLD:

FPGA chứa hơn 100000 khối logic nhỏ trong khi CPLD chỉ chứa tối đa vài nghìn khối.Về kiến trúc, các FPGA được xem như các thiết bị có mật độ cao (fine-grain devices), trong khi CPLD là các thiết bị mật độ thưa (coarse-grain devices).Các FPGA được sử dụng cho các ứng dụng phức tạp, trong khi các CPLD phù hợp cho các ứng dụng đơn giản và ít linh hoạt so với FPGA.Các FPGA gồm có các khối logic nhỏ, trong khi các CPLD được làm ra từ các khối logic lớn.FPGA là chip logic số dựa vào RAM, trong khi CPLD là chip dựa vào EEPROM.Các FPGA dựa vào các bảng Look-up (LUTs) bên trong, trong khi các CPLD lại hình thành các hàm logic nhờ các mạch cổng sea-of-gates.Hầu hết các FPGA có các mạch logic mức cao, ví dụ, các bộ cộng, bộ nhân và các bộ nhớ nhúng, và các khối logic thực hiện các bộ giải mã hoặc các hàm toán học, trong khi đó CLPD thì không. Bình thường, các FPGA đắt hơn so với các CPLD.Các trễ (Delay) trong các CLPD lớn hơn so với các trễ trong các FPGA.

3) FPGA và vi điều khiển:

Vi điều khiển là hệ thống tính toán, có kiến trúc Havard bên trong với đầy đủ các khối chức năng của một máy tính nhỏ và tập lệnh máy riêng. Trong khi đó FPGA là chip chưa định hình cấu trúc của một thiết bị thực hiện một chức năng cụ thể.Hoạt động của vi điều khiển phải được lập trình và phải cài đặt mã chương trình vào trong bộ nhớ code bên trong chip vi điều khiển. Lập trình cho vi điều khiển là lập trình phần mềm bằng bất kỳ ngôn ngữ lập trình nào. Chương trình không làm thay đổi cấu trúc bên trong của vi điều khiển. Trong khi đó, để tạo FPGA thành một thiết bị theo yêu cầu ứng dụng riêng cần phải lập trình bằng một ngôn ngữ mô tả phần cứng. Chương được dịch ra một file dạng bit hay chuỗi bit (BitStream) và được nạp vào trong FPGA cấu hình các liên kết các thành phần logic đã có sẵn bên trong FPGA tạo ra một thiết bị thực hiện chức năng cụ thể. Như vậy, lập trình FPGA làm thay đổi phần cứng trong FPGA. Đó là lập trình phần cứng.Các FPGA được ứng dụn g trong các hệ thống nhúng, có thể kết hợp với các vi điều khiển.

Xét chung về công nghệ mạch tích hợp bán dẫn: bộ nhớ, các bộ vi xử lý, ASIC, PLD, FPGA, và SoC, ta có sơ đồ mối liên quan của chúng với nhau như mổ tả ở hình 1.4 dưới đây:



Hình 1.4: Quan hệ giữa các công nghệ IC

1.3. KẾT LUẬN CHƯƠNG:

Trong chương này đã giới thiệu được tổng quan về hệ thống nhúng, phân tích được các đặc điểm, cấu trúc cũng như phân loại được hệ thống nhúng. Các đặc điểm của FPGA được nêu ra ở đây với các so sánh với các công nghệ PLD, ASC và vi điều khiển là sơ sở cho việc lựa chọn giải pháp triển khai hệ thống nhúng trên FPGA mà chương 2 sẽ trình bày.

CHƯƠNG 2. HỆ THỐNG NHÚNG TRÊN FPGA

2.1. LỰA CHỌN CÔNG NGHỆ ALTERA FPGA

Vớinhững phân tích và đánh giá về FPGA ở chương 1, có thể nhận thấy rằng, lựa chọn thiết kế hệ thống nhúng trên FPGA là một giải pháp thích hợp, tiết kiệm chi phí về thiết kế, đầu tư, và thiết kế có thể thay đổi linh hoạt tùy theo yêu cầu của hệ thống. Các nhà sản xuất chip FPGA như Xilinx, Altera đã có nhiều sản phẩm cung cấp tại Việt Nam, do đó sẽ là thuận tiện lựa chọn các bảng phát triển của các hãng này để thực hiện thiết kế, đặc biệt, bảng phát triển Altera DE2 khá phổ biến cho đào tạo và nghiên cứu thiết kế các hệ thống nhúng ứng dụng cho các thiết bị xử lý tín hiệu Audio, video.

Như đã giới thiệu và phân tích về các hệ thống nhúng trong Chương 1, embedded linux là một hệ thống mã nguồn mở hoàn chỉnh được sử dụng cho một thiết bị, bao gồm Linux kernel và các ứng dụng kèm theo. Với các tính năng và tiện ích vượt trội, embedded linux giúp cho nhà thiết kế tối ưu về thời gian cũng như chi phí thiết kế, phát triển và dễ dàng thay đổi để thích nghi được với các yêu khác nhau của từng hệ thống. uClinux là một hệ điều hành thuộc hệ thống nhúng embedded linux nên có được tất cả những ưu điểm vượt trội đã nêu, hơn nữa với kích thước nhỏ gọn, linh hoạt nên uClinux rất phù hợp để thiết kế cho các hệ thống nhúng sử dụng bộ vi điều khiển không có đơn vị quản ý bộ nhớ MMU.

Đó là cơ sở để tác giả lựa chọn giải pháp hệ thống nhúng trên FPGA là hệ điều hành nhúng uClinux trên bảng phát triển Altera DE2.

2.1.1. Hệ phát triển Altera DE2 Cyclone II 2C35 FPGA

1)Gói Hệ phát triển Altera DE2 Cyclone II 2C23FPGA:

Được cung cấp gồm (hình 2.1):

- Bảng phát triển Altera DE2 Cyclone II 2C35 FPGA

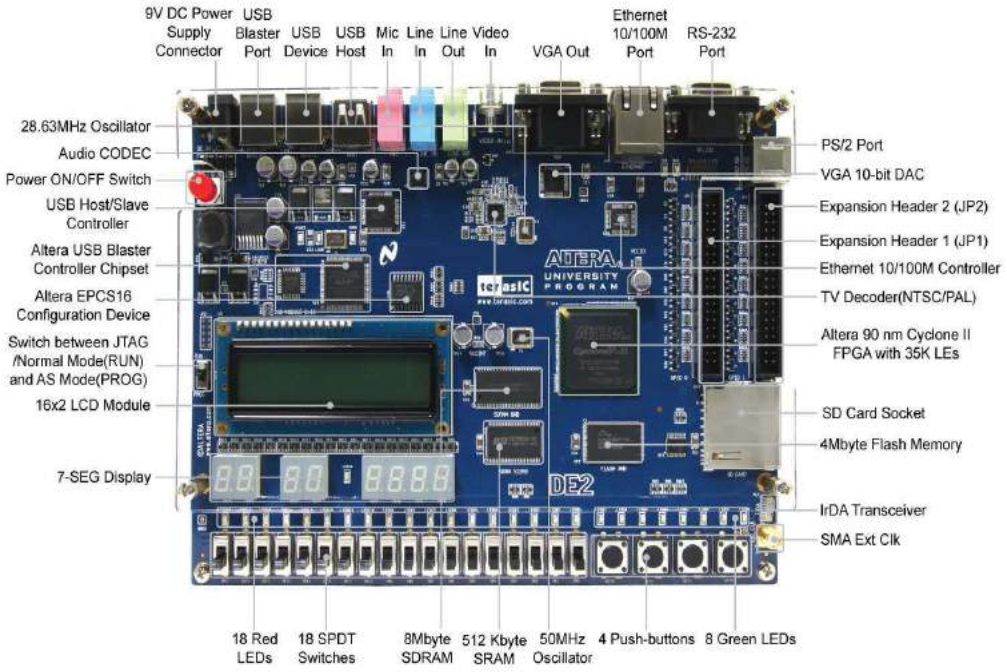
- Cáp USB để lập trình và điều khiển FPGA
- CD-ROM chứa tài liệu DE2 và các vật tư hỗ trợ, gồm hướng dẫn sử dụng, tiện ích của phiến điều khiển (Control Panel), các thiết kế và ví dụ tham chiếu, dữ liệu kỹ thuật của thiết bị, các bài học, và tập hợp các bài thí nghiệm.
- CD-ROMs chứa phần mềm Altera’s Quartus® II Web Edition và Nios® II Embedded Design Suit Evaluation Edition.

- Túi gồm 6 vỏ cao su (rubber silicon) cho bảng DE2 stands. Túi cũng có một số chân mở rộng có thể được dùng với thiết bị kiểm tra các đầu mở rộng I/O của bảng DE2.
- Vỏ nhựa cho bảng
- Bộ nguồn 9V DC gắn tường

Hình 2.1: Hộp đựng sản phẩm Altera DE2Cyclone II 2C35 và các phụ kiện đi kèm

2) Bảng phát triển Altera DE2 Cyclone II 2C35 FPGA:

1) Đặc điểm phần cứng:



Hình 2.2: Bảng Altera DE2 Cyclone II 2C35 FPGA

Bảng phát triển Altera DE2 2C35 FPGA gồm:

- Altera Cyclone® II 2C35 FPGA
- Thiết bị cấu hình tuần tự Altera EPCS16

- USB Blaster (trên bảng) để lập trình và điều khiển API của người dùng; hỗ trợ các chế độ lập trình cho cả JTAG và Active Serial (AS)
- 512-Kbyte SRAM
- 8-Mbyte SDRAM
- 4-Mbyte Flash memory (1 Mbyte trên một số bảng)
- Khe cắm thẻ SD
- 4 nút nhấn (pushbutton switches)
- 18 nút gạt (toggle switches)
- 18 LEDs đỏ
- 9 LEDs xanh
- 50-MHz oscillator and 27-MHz (từ TV decoder) cho các nguồn đồng hồ bên ngoài
- 24-bit CD-quality audio CODEC với line-in, line-out, và microphone-in jacks
- VGA DAC (10-bit high-speed triple DACs) với VGA-out connector
- TV Decoder (NTSC/PAL) và TV-in connector
- 10/100 Ethernet Controller với RJ-45 connector
- USB Host/Slave Controller với USB connectors loại A và loại B
- RS-232 transceiver và 9-pin connector
- PS/2 mouse/keyboard connector
- IrDA transceiver
- Hai 40-pin Expansion Headers với bảo về diode

2) Sơ đồ khối:

Hình 2.3 là sơ đồ các vị trí các khối phần cứng trên bảng Altera DE2 2C35 FPGA. Hình 2.4 cho sơ đồ khối của bảng Altera DE2. Để cung cấp cho người dùng độ linh hoạt tối đa, tất cả các kết nối được làm thông qua thiết bị Cyclone II FPGA. Như vậy, người dùng có thể cấu hình FPGA để thực thi bất kỳ thiết kế nào. Các đặc điểm kỹ thuật chi tiết của các khối như sau:

Cyclone II 2C35 FPGA:

- 3,216 Les
- 105 M4K RAM blocks

- 483,840 total RAM bits
- 35 embedded multipliers
- 4 PLLs
- 475 user I/O pins
- FineLine BGA 672-pin package

Thiết bị cấu hình tuần tự và mạch USB Blaster:

- Thiết bị cấu hình tuần tự Altera's EPCS16
- USB Blaster trên bảng để lập trình và điều khiển API của người dùng
- Hỗ trợ các chế độ lập trình JTAG và AS

SRAM:

- 512-Kbyte SRAM memory chip
- Tổ chức như 256K x 16 bits
- Có thể truy nhập như bộ nhớ cho Nios II processor và nhờ DE2 Control Panel

SDRAM:

- 8-Mbyte Single Data Rate Synchronous Dynamic RAM memory chip
- Tổ chức như 1M x 16 bits x 4 banks
- Có thể truy nhập như bộ nhớ cho Nios II processor và nhờ DE2 Control Panel

Flash memory:

- 4-Mbyte NOR Flash memory (1 Mbyte on some boards)
- 8-bit data bus
- Có thể truy nhập như bộ nhớ cho Nios II processor và nhờ DE2 Control Panel

SD card socket:

- Đảm bảo SPI và 4-bit SD mode chp truy nhập thẻ SD
- Có thể truy nhập như bộ nhớ cho Nios II processor với DE2 SD Card Driver

Các nút nhấn (Pushbutton switches):

- 4 nút gạt
- Được chặn bởi mạch Schmitt trigger

- Bình thường nhả; tạo xung tích cực thấp khi nhấn nút

Các nút gạt (Toggle switches):

- 18 nút gạt cho người dùng đặt vào
- Nút gạt tạo logic 0 khi gạt xuống (DOWN) (về phía cạnh của bảng DE2) và logic 1 khi gạt lên (UP) về phía trong của bảng DE2

Các đầu vào đồng hồ (Clock inputs):

- 50-MHz oscillator
- 27-MHz clock input
- SMA external clock input

Audio CODEC:

- Wolfson WM8731 24-bit sigma-delta audio CODEC
- Line-level input, line-level output, và microphone input jacks
- Tần số lấy mẫu: 8 to 96 KHz
- Các ứng dụng cho MP3 players và recorders, PDAs, smart phones, voice recorders, v.v.

VGA output:

- Sử dụng ADV7123 140-MHz triple 10-bit video DAC tốc độ cao
- Với 15-pin D-sub connector
- Hỗ trợ đến 1600 x 1200 ở tốc độ làm tươi 100-Hz
- Có thể dùng với Cyclone II FPGA để thực hiện TV Encoder hiệu suất cao

Mạch giải mã NTSC/PAL TV:

- Sử dụng ADV7180 Multi-format SDTV Video Decoder
- Hỗ trợ giải điều chế mầu NTSC/PAL/SECAM
- Một 10-bit ADC, 4X over-sampling cho CVBS
- Hỗ trợ Composite Video (CVBS) RCA jack input.
- Hỗ trợ digital output formats (8-bit/16-bit): ITU-R BT.656 YCrCb 4:2:2 output + HS, VS, and FIELD
- Các ứng dụng: DVD recorders, LCD TV, Set-top boxes, Digital TV, Portable video devices

10/100 Ethernet controller:

- MAC và PHY được tích hợp với giao tiếp bộ xử lý chung
- Hỗ trợ các ứng dụng 100Base-T và 10Base-T
- Hỗ trợ thao tác full-duplex ở 10 Mb/s và 100 Mb/s, với auto-MDIX
- Đáp ứng đầy đủ IEEE 802.3u Specification
- Hỗ trợ tạo tổng kiểm tra IP/TCP/UDP và kiểm tra
- Hỗ trợ back-pressure mode cho điều khiển luồng half-duplex mode

USB Host/Slave controller:

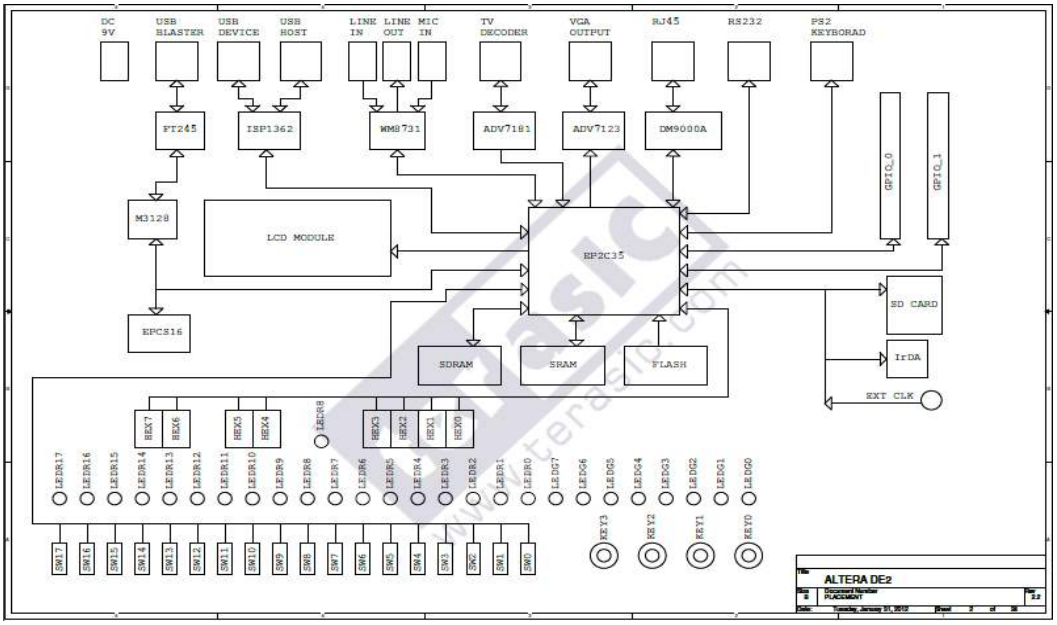
- Phù hợp đầy đủ với Universal Serial Bus Specification Rev. 2.0
- Hỗ trợ truyền dữ liệu tốc độ đầy đủ và tốc độ thấp
- Hỗ trợ cả USB host và thiết bị
- Hai cổng USB (một loại A cho host và một loại B cho thiết bị)
- Đảm bảo giao tiếp tốc độ cao cho hầu hết các processors; hỗ trợ Nios II với Terasic driver
- Hỗ trợ Programmed I/O (PIO) và Direct Memory Access (DMA)
- Serial ports
- Một cổng RS-232
- Một cổng PS/2
- DB-9 serial connector cho cổng RS-232
- PS/2 connector để nối PS2 mouse hoặc keyboard với bảng DE2

IrDA transceiver:

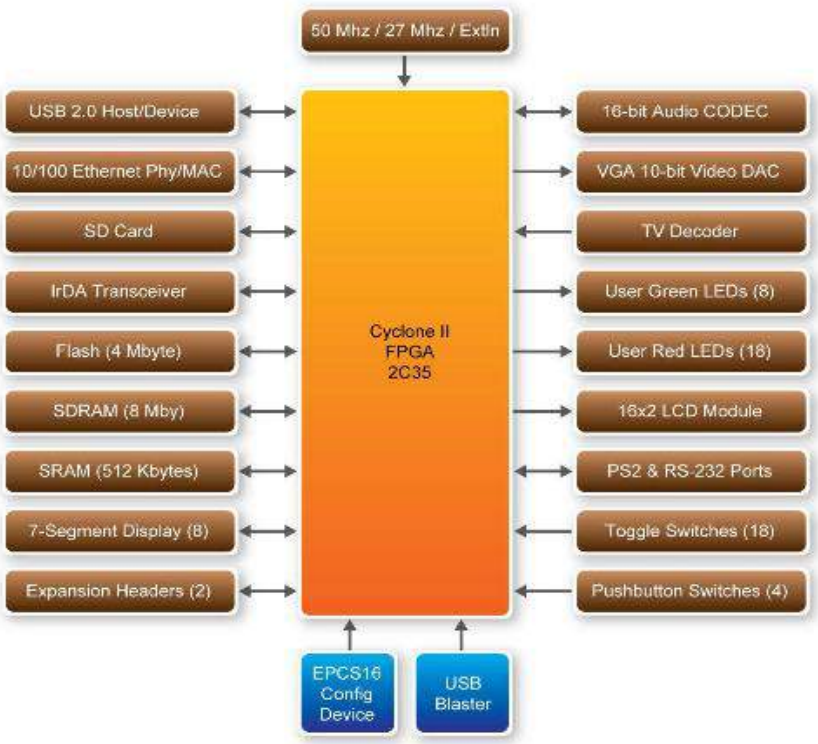
- Có 115.2-kb/s infrared transceiver
- Dòng điều khiển LED 32 Ma
- Tích hợp EMI shield
- IEC825-1 Class 1 eye safe
- Edge detection input

Hai đầu mở rộng 40-pin (expansion headers):

- 72 Cyclone II I/O pins, 8 đường nguồn và đất, được đưa tới 2 connectors mở rộng 40-pin
- 40-pin header được thiết kế để nhận cáp bẹt (ribbon) 40-pin được sử dụng cho các drivers ID
- Bảo vệ bằng Diode hoặc resistor [5]



Hình 2.3: Sơ đồ vị trí các khối bảng phát triển Altera DE2 2C35 FPGA



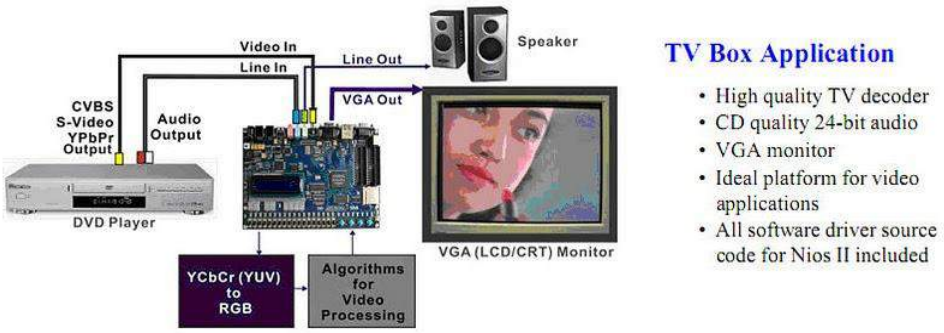
Hình 2.4: Sơ đồ khối bảng phát triển Altera DE2 2C35 FPGA

2.1.2. Các ứng dụng demo của Altera DE2

1) TV Box Application:

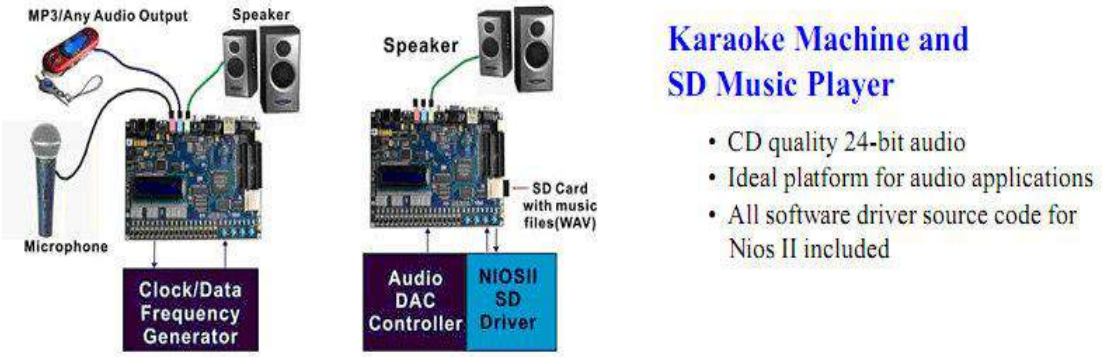
Có thể cấu hình tạo bộ xử lý mềm Nios II như là CPU với các giao tiếp phù hợp cho các ứng dụng trên bảng Altera DE2. Ví dụ Kết nối các thiết bị audio video: input:

microphone, camera hay DVD player; audio: speaker, VGA monitor hoặc tạo Web server nhúng điều khiển qua mạng LAN.



Hình 2.5: Sơ đồ demo TV Box application

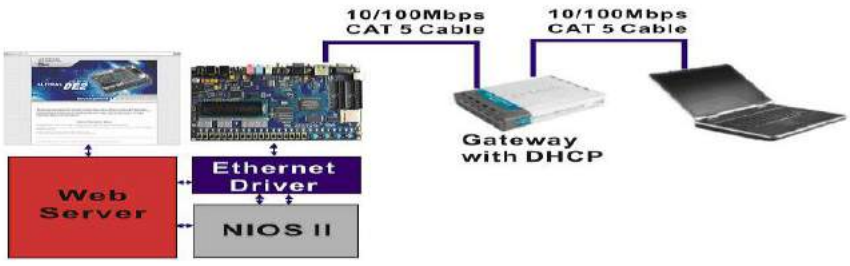
2) Karaoke Machina:



Hình 2.6: Sơ đồ demo Karaoke machine

3) Hệ thống web server:

Có thể cấu hình một Web server nhúng trên bảng FPGA trên c

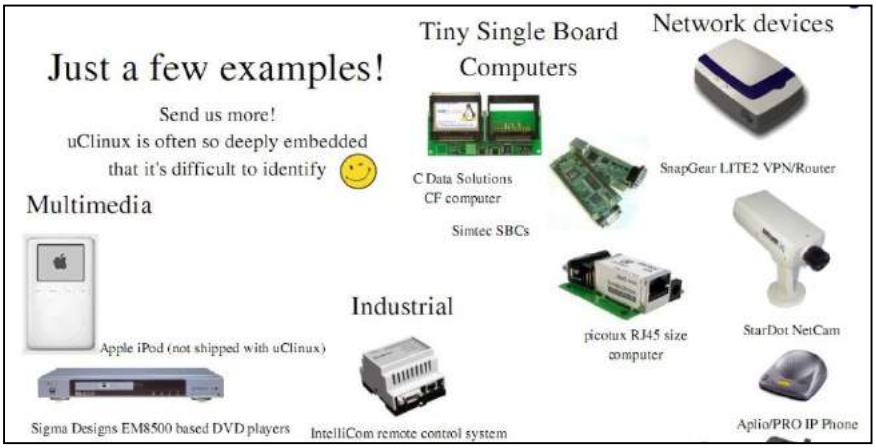


Hình 2.7. Sơ đồ demo Web server

2.2. HỆ ĐIỀU HÀNH NHÚNG uCLINUX

2.2.1. Tổng quan hệ điều hành nhúng uClinux

uClinux (viết tắt của từ MicroControllerLinux) là một phiên bản hệ điều hành sử dụng nhân Linux được thiết kế cho các hệ thống nhúng sử dụng bộ vi điều khiển không có đơn vị quản ý bộ nhớ MMU. uClinux có kiến trúc, cách thức hoạt độngvà phương pháp phát triển ứng dụng như hệ điều hành Linux. uClinux làhệ điều hành mã nguồn mở nên khi xây dựng để biên dịch người phát triển có thể cấu hình theo để tạo ra một hệ điều hành phù hợp với hệ thống cần phát triển.



Hình 2.8: Một số thiết bị ứng dụng hệ điều hành uClinux

1) Lịch sử phát triển uClinux:

uClinux được phát triển từ năm 1997. Mục đích tạo ra hệ điều hành này là phát triển một phiên bản nhân hệ điều hành Linux 2.0 để nhúng vào các vi điều khiển. Nó được Jeff Dionne, Kenneth Albanowski và nhóm các nhà phát triển khác đặt vấn đề là khả năng nhúng Linux vào mạng vi điều khiển không có đơn vị quản lý bộ nhớ MMU (Memory Management Unit), khả năng truyền thông giữa mạng đó với các hệ thống truyền thông. Phiên bản phát hành đầu tiên của hệ điều hành này được phát hành cùng với các vi xử lý Motorola 6800, nó được triển khai trong bộ điều khiển SCADA năm 1997/98. Phiên bản này đã được cộng đồng nguồn mở phát triển và một phiên bản khác đã được sử dụng cho Palm Pilot vào tháng 2 năm 1998.

Trong hệ điều hành uClinux có một số thay đổi so với hệ điều hành Linux, thư viện uC-libc được thiết kế để thay cho thư viện libc và glibc trong hệ điều hành Linux. Một cải tiến khác đã được thực hiện bởi SnapGear là thêm một định dạng mới Binary

Flat – bFLT. Đây là một định dạng file thực thi tương đối đơn giản và nhỏ gọn xây dựng trên cơ sở file a.out.

Hệ điều hành uClinux là một hệ điều hành đa nhiệm, các chương trình có thể chạy ở nhiều mức khác nhau của hệ thống và cho phép chạy các ứng dụng đa luồng. Việc nhúng hệ điều hành uClinux vào các vi điều khiển đã giúp cho các nhà phát triển tạo ra các ứng dụng dễ dàng vì phương pháp lập trình giống như trên môi trường Linux. uClinux là một hệ điều hành thời gian thực, nhà phát triển có thể chạy các ứng dụng đa luồng trên môi trường hệ điều hành.

Nhiệm vụ chính khi phát triển hệ điều hành uClinux là cấu hình nhân hệ điều hành cho phù hợp với hệ thống cần phát triển, biên dịch nhân, phát triển các driver cho các ngoại vi và phát triển các ứng dụng cho hệ thống nhúng.

2) Kiến trúc hệ điều hành uClinux:

uClinux là một hệ điều hành dùng phổ biến cho các hệ thống nhúng Linux. Nó thường được dùng cho các vi điều khiển không có đơn vị quản lý bộ nhớ (MMU). Ngày nay nhân hệ điều hành này hỗ trợ cho rất nhiều loại nền tảng CPU khác nhau như ColdFire, Axis ETRAX, ARM, Atari 68k... Giống như Linux, uClinux cũng hỗ trợ giao thức TCP/IP và các giao thức giao tiếp mạng khác. Nó cũng hỗ trợ các hệ thống file khác nhau và thêm vào một số dạng file đặc biệt được thiết kế cho các hệ thống nhúng.

Để có thể chạy trên các vi điều khiển không có MMU thì nhân của hệ điều hành có một số thay đổi. Toàn bộ mã nguồn và các chức năng điều khiển của MMU được loại bỏ ra khỏi mã nguồn của nhân hệ điều hành. Một số chức năng khác cũng được điều chỉnh cho phù hợp.

Sự thuận lợi chính của nhân uClinux mang lại so với Linux chạy trên PC là kích thước của nhân. Khi biên dịch nhân, nhà phát triển phải thiết lập các lựa chọn biên dịch như hỗ trợ loại vi xử lý, hệ thống file và các driver của nhân để kích thước giảm xuống còn khoảng 400 KB. Tuy nhiên vào lúc khởi động thì nhân của hệ điều hành sẽ yêu cầu không gian bộ nhớ khoảng 1MB. Trong thực tế, kích thước bộ nhớ cần khoảng 2MB vì còn cần cho các ứng dụng. Với uClinux, kích thước ảnh nhân hệ điều hành được điều chỉnh khoảng từ 500 tới 900 KB.

Tương tự như hệ điều hành Linux, mã nguồn của uClinux cũng có thể tải miễn phí với giấy phép bản quyền GNU GPL. Từ website www.uclinux.org, các nhà phát triển có thể tải gói phân phối của hệ điều hành bao gồm mã nguồn nhân, các thư viện và một số ứng dụng đã được phát triển, những thông báo lỗi và những lỗi đã được sửa.

2.2.2. Các đặc điểm của hệ điều hành uClinux

1) Ưu điểm và nhược điểm của hệ điều hành uClinux:

Với đặc điểm mã nguồn mở, các nhà phát triển có thể tải và chỉnh sửa theo ý muốn. Tuy đã được phát triển từ lâu nhưng bên cạnh những ưu điểm, uClinux vẫn còn tồn tại một số nhược điểm.

Ưu điểm:

- uClinux tích hợp sẵn nhiều IP của hệ điều hành Linux chuẩn, các file hệ thống, các phần mềm miễn phí mang tính ổn định cao.
- Nhân Linux 2.6 có kích thước dưới 300KB.
- Nhiều ứng dụng của hệ thống nhúng có thể thực hiện mà không cần MMU.
- Ứng dụng người dùng có thể truy cập vào toàn bộ hệ thống, đến từng thanh ghi của thiết bị.
- uClinux có thể sử dụng hầu hết các system calls của hệ điều hành Linux chuẩn.
- uClinux hỗ trợ chạy đa nhiệm full multi-tasking với hạn chế tương đối nhỏ.
- uClinux hỗ trợ nhiều bộ vi xử lý mà Linux chuẩn không hỗ trợ.

Nhược điểm:

-uClinux phát triển chậm hơn so với Linux.

-Tài liệu và tài nguyên online ít hơn so với Linux.

-Nhiều đường link cũng như source code trên trang chủ đã quá cũ.

-uClinux không có MMU nên sẽ không có VM và không có sự bảo vệ bộ nhớ.

- uClinux không có sự phân trang theo yêu cầu, cần phải tải toàn bộ mã nguồn chương trình vào trong RAM, thay vì chỉ cần tải những trang mà chương trình thực sự truy cập đến. Trong hệ điều hành uClinux, kích thước bộ nhớ cho các process được cố định nên không thể phân mảnh bộ nhớ, dẫn đến tốn bộ nhớ hơn.

2) Runtime linker and loader:

Đây là một phần quan trọng của hệ điều hành có nhiệm vụ tải và chạy chương trình. Nó có tác dụng định vị mã của chương trình ứng dụng trong các bộ nhớ thứ cấp

và cài đặt vào không gian bộ nhớ hệ thống. Mã chương trình ứng dụng có thể yêu cầu điều chỉnh phù hợp với vị trí nơi mà nó được cài đặt vào.

Trong các hệ thống có MMU, ứng dụng có thể sử dụng lợi thế VM nên các lệnh và các dữ liệu chỉ đọc có thể được chia sẻ giữa các ứng dụng. Trong hệ thống không có MMU, kiến trúc flat memory model sẽ cố định từng ứng dụng vào các vùng bộ nhớ khác nhau trong quá trình thực thi. Linker/loader sẽ sắp xếp các ứng dụng và đưa nó vào không gian nhớ hệ thống. Các ứng dụng chỉ có thể làm việc trên phân vùng mà nó được quy định. Về cơ bản, Linker và loader thực thi ba nhiệm vụ chính:

- Tải chương trình bằng cách copy chương trình vào bộ nhớ chính.
- Gán địa chỉ riêng cho mã nguồn và dữ liệu.
- Phân tách các địa chỉ, gán con trỏ địa chỉ tới chương trình con.

Loader và linker trải qua ba bước như trên để tạo ra một thực thi. Gói thư viện dùng để biên dịch (toolchain) của uClinux sử dụng ELF (Execution and Linking Format) đã được sửa đổi chút ít cho phù hợp. Đây là một định dạng file chuẩn của các file thực thi, mã đối tượng, các thư viện được chia sẻ. ELF nguyên bản được phát triển bởi Unix System Laboratories và được sử dụng rộng rãi trong nhiều hệ điều hành. Năm 1999, nó được công nhận như một định dạng file nhị phân chuẩn trong Unix. Không giống như nhiều định dạng file thực thi độc quyền, ELF rất linh hoạt, có thể mở rộng và không phụ thuộc vào vi xử lý hoặc kiến trúc.

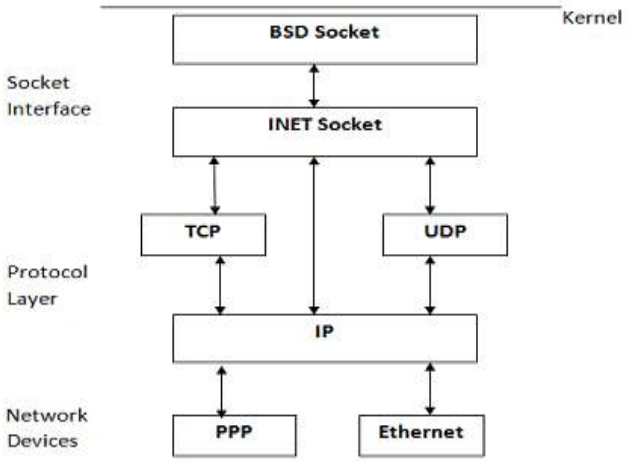
ELF định nghĩa liên kết và thực thi của một file. Một file ELF bao gồm bốn phần chính:

- ELF headers: chứa thông tin về file.
- Program header: header của chương trình.
- Section header: nội dung của file.
- Data: dữ liệu.

uClinux cung cấp bộ nhớ như một không gian địa chỉ duy nhất. Điều này có nghĩa là stack tiếp giáp vật lý với dữ liệu tĩnh. Các stack phải có một kích thước cố định và không được chồng lên dữ liệu tĩnh cũng như vùng mã nguồn. Do không có MMU nên uClinux sẽ không có bảo vệ bộ nhớ. Vì kích thước stack là cố định nên sẽ dẫn tới lãng phí một phần không gian bộ nhớ.

3) Ethernet:

uClinux hỗ trợ nhiều ứng dụng khác nhau, trong đó có ứng dụng mạng bao gồm TCP/IP stack và các giao thức khác liên quan đến mạng như TCP/UDP, IP, Ethernet. Khi một ứng dụng mạng được tạo ra và truyền nhận trong mạng, đầu tiên nó sẽ truyền đi gói tin thông qua lớp socket đến lớp transport (TCP hoặc UDP) sau đó chúng sẽ đi đến lớp IP, tại đây kernel sẽ phân giải địa chỉ mà gói tin cần truyền đến. Cấu trúc của ứng dụng mạng trong Linux được biểu diễn trên hình 2.9.



Hình 2.9. Ethernet trong uClinux

Hai kiểu socket chính được cung cấp trong Linux là BSD (Berkeley Socket Distribution) và INET socket. BSD socket được thực thi dựa trên lớp INET socket và cung cấp giao diện cho người sử dụng.

4) Các thư viện được sử dụng để phát triển hệ điều hành uClinux:

Để phát triển uClinux, nhà phát triển có thể lựa chọn giữa hai thư viện libC là uC-libc và uClibc của uClinux. Thư viện uClibC giống như thư viện glibc phát triển cho Linux, các hàm của glibc cũng được áp dụng cho uClibc. Ngoài ra gói mã nguồn của uClinux còn một số thư viện khác như:

- libgmb: GNU library

- libg: gồm có gtermcap, được thiết kế để hỗ trợ truyền đi các chuỗi điều khiển đến terminal

- libcap: cung cấp giao diện hệ thống độc lập ở mức user.

- libatm: hỗ trợ cho việc truyền bất đồng bộ (ATM).

- libjpeg: hỗ trợ hiển thị hình định dạng JPEG.
- libm: hỗ trợ cho các phép tính toán học.
- libnet: cung cấp thư viện API cho các giao thức mạng.
- libpam: thư viện giao diện phân cấp.
- libpng: thư viện giao diện đồ họa cho các ứng dụng mạng.

5) Mã nguồn hệ điều hành uClinux:

Có thể tải về các phiên bản khác nhau của hệ điều hành uClinux từ địa chỉ www.uclinux.org. Toàn bộ mã nguồn uClinux chứa trong một file nén được đặt tên theo một định dạng chuẩn như sau nios2-linux-YYYYMMDD.tar trong đó YYYY là năm, MM là tháng, DD là ngày. Khi được giải nén nằm trong thư mục chính là uClinux-dist. Công việc của người phát triển là biên dịch tạo ra một file ảnh để đưa vào hệ thống.

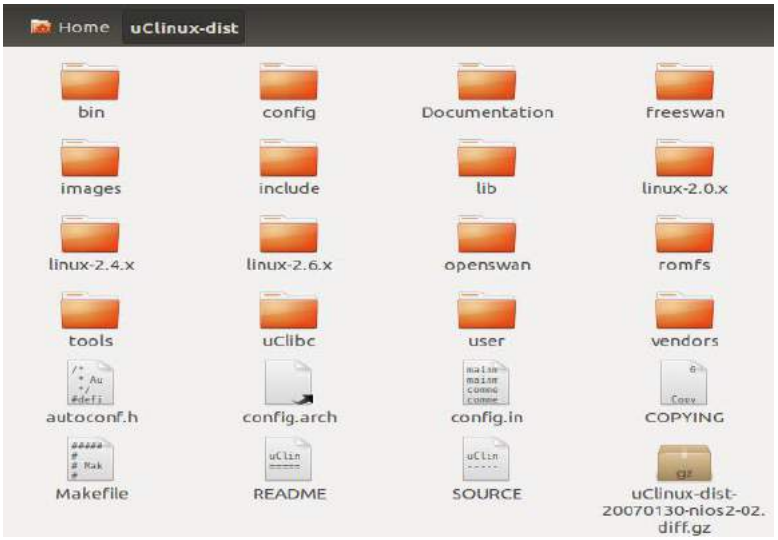
Trong luận văn này tôi sử dụng mã nguồn hệ điều hành uClinux với phiên bản nhân linux-2.6 (uClinux-dist-20070130). Sau khi giải nén, thư mục chính uClinux-dist gồm các thư mục sau:

- bin: gồm các tiện ích để tạo ra file flash.bin.
- config: các tệp cấu hình hệ thống file uClinux.
- Documention: tài liệu chi tiết của uClinux.
- freeswan: các chương trình bảo mật, mã hóa.
- images: chứa nhân dạng nhị phân của hệ điều hành sau khi biên dịch.
- lib: chứa các thư viện cho ứng dụng.
- linux-2.0.x: mã nguồn nhân uClinux.
- linux-2.4.x: mã nguồn nhân uClinux.
- linux-2.6.x: mã nguồn nhân uClinux.

-romfs: cấu trúc hệ thống file của ROM (Read-Only Memory), bao gồm ứng dụng, các file thiết bị. Thư mục này được tạo ra khi biên dịch.

- tools: các công cụ để biên dịch.
- uClibc : thư viện C.
- user: nơi lưu các ứng dụng của người dùng.
- vendor: gồm các lõi xử lý mà hệ điều hành hỗ trợ.

Thư mục của gói mã nguồn được giải nén ra được minh họa ở hình 2.10.



Hình 2.10. Cấu trúc thư mục của gói mã nguồn uClinux-dist

2.2.3. Công cụ phần mềm thiết kế Altera Quartus II Web Edition

Altera Quartus II [12] là phần mềm thiết kế thiết bị logic có thể lập trình được. Nó cho phép phân tích và tổng hợp các thiết kế của ngôn ngữ mô phỏng phần cứng HDL, cho phép nhà phát triển biên dịch các thiết kế của họ, thực hiện phân tích thời gian, kiểm tra các sơ đồ thiết kế mở mức RTL, mô phỏng phản ứng của thiết kế đối với các kích hoạt, và cấu hình thiết bị nhờ lập trình. Quartus gồm một thực hiện của VHDL hay Verilog cho mô tả phần cứng, soạn thảo hiển thị các mạch logic, và mô phỏng bằng đồ thị dạng sóng các tín hiệu của thiết kế.

Quartus II bao gồm:

- SOPC Builder: công cụ trong phần mềm Quartus II, thực hiện tự động tạo ra logic liên kết và tạo một tiến trình kiểm tra testbench chức năng của thiết kế. SOPC Builder tích hợp thư viện các linh kiện (gồm bộ xử lý mềm nios II, các bộ điều khiển bộ nhớ, các giao tiếp, và các ngoại vi) và một giao tiếp để tích hợp tùy nhu cầu những linh kiện này. Các liên kết được tạo ra thông qua Avalon Bus.

- Qsys, công cụ tích hợp hệ thống, là thế hệ sau của SOPC Builder. Nó sử dụng kiến trúc mạng trên chip của FPGA (NoC) cho các hệ thống nhúng FPGA (SoCFPGA).

- SoCEDS, tập hợp tạo cầu nối liền mạch giữa công cụ MATLAB/Simulink và phần mềm Quartus II, như vậy, các nhà thiết kế có khả năng phát triển thuật toán, mô phỏng, và so sánh của các công cụ thiết kế mức hệ thống MATLAB/Simulink.

- Bộ công cụ giao tiếp bộ nhớ ngoài, nó xác định chính xác các kết quả và các số đo các giới hạn của từng tín hiệu DQS.

- Tạo các file JAM/STAPL cho cho lập trình thiết bị trong mạch của JTAG.

Web Edition:là phiên bản của Quartus II có thể được tải miễn phí từ Web site Altera.com và có thể thực hiện các thiết kế với giới hạn một số chức năng nhưng đủ cho đào tạo không cần có bản quyền. Trong đó, có bao gồm các thư viện thiết bị FPGA của Altera.

Subscription Edition: là phiên bản có thể tải miễn phí, nhưng để chạy được cần phản mua bản quyền. Web Edition có thể được sử dụng trên phần mềm nay với một số thiết bị hạn chế có thể được sử dụng.

Các phiên bản của Quartus có thể cài đặt trên các hệ điều hành:

- Microsoft Windows Vista (32-bit and 64-bit)
- Microsoft Windows XP (32-bit and 64-bit)
- Microsoft Windows 2000
- Solaris 8 and 9 (32-bit and 64-bit)
- SUSE Linux Enterprise 9 (32-bit and 64-bit)
- Red Hat Enterprise Linux 5 (32-bit and 64-bit)
- Red Hat Enterprise Linux 4 (32-bit and 64-bit)
- Red Hat Enterprise Linux 3 (32-bit and 64-bit)

2.3. KẾT LUẬN CHƯƠNG

Nội dung chương này trình bày lý do lựa chọn thiết bị FPGA và hệ điều hành nhúng để thiết kế hệ thống nhúng. Đó là lựa chọn bảng phát triển Altera DE2 và uClinux. Vì vậy, ở đây đã nêu ra các đặc tính kỹ thuật, sơ đồ khối, lõi xử lý Nios II, chip FPGA, và các giao tiếp, các khả năng ứng dụng của bảng Altera DE2, và công cụ thiết kế Altera System Design Suite. Kiến trúc và các đặc điểm chính của hệ điều hành nhúng uClinux : mã nguồn mở, các thư viện, các thư mục của uClinux sau khi gỡ nén, tạo file ảnh zImage, và khởi động nhân uCLinux trên PC-linux cũng được nêu ra cụ thể.

CHƯƠNG 3: THIẾT KẾ HỆ VI ĐIỀU KHIỂN LÕI MỀM NIOS II 32-BIT VÀ CÀI ĐẶT uCLINUX

3.1. VI ĐIỀU KHIỂN LÕI MỀM NIOS II

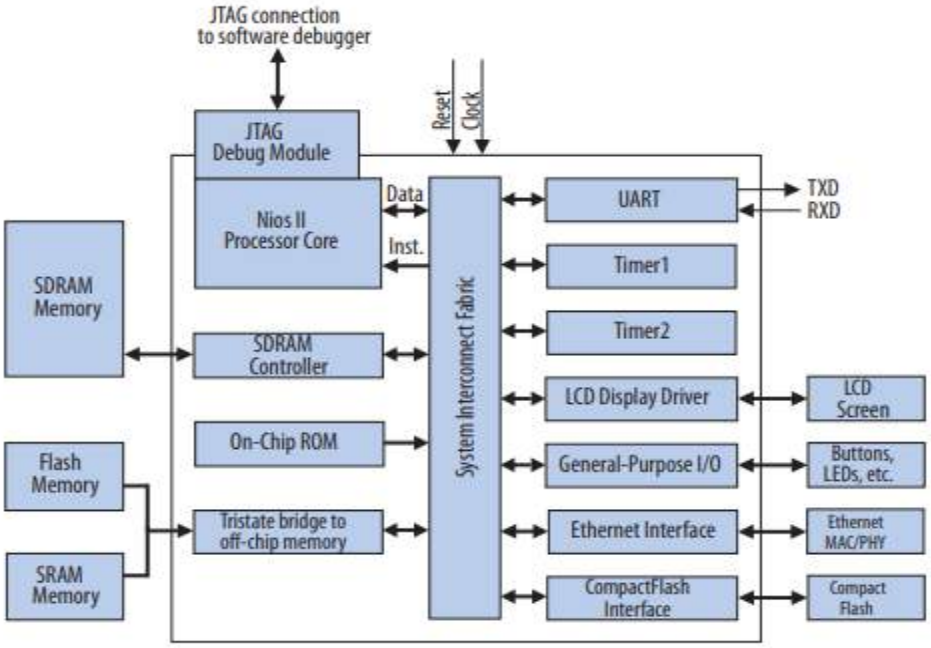
3.1.1. Sơ đồ khối của lõi mềm Nios II

Vi xử lý lõi mềm (soft microprocessor)[17],cũngđược gọi gọi là softcore microprocessor hay soft processor, được thiết kế bằng sự tổng hợp các mạch logic trên các thiết bị bán dẫn có khả năng lập trình được như ASIC, FPGA, CPLD.

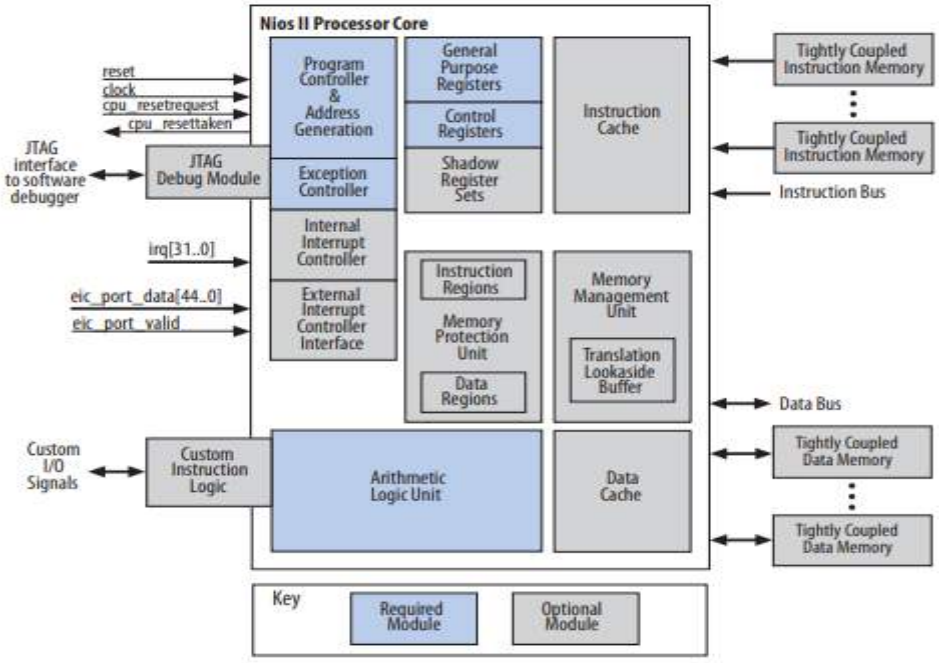
Sự khác biệt giữa vi xử lý cứng và vi xử lý lõi mềm là: vi xử lý lõi cứng là vi mạch tổ hợp tích hợp rất lớn đã được đóng vỏ cố định không thể thay đổi và được thương mại hóa, trong khi đó lõi mềm có thể thay đổi chức năng bằng lập trình (cấu hình lại) tùy thuộc vào ứng dụng.

Vi xử lý lõi mềm trên FPGA là loại phổ biến cho các thiết kế thử, cho đào tạo, và rất thích hợp cho các nghiên cứu ứng dụng khoa học kỹ thuật.

Bộ xử lý lõi mềm Nios II có thể được lập trình và cấu hình trên các thiết bị FPGA của Altera qua các bảng phát triển Altera DE2 Cyclone. Một hệ thống Nios II trên chip FPGA có thể có các giao tiếp như ở hình 3.1, và sơ đồ khối của Nios II cho ở hình 3.2 [13].



Hình 3.1: Hệ thống Nios II được thực hiện trên bảng DE2



Hình 3.2: Sơ đồ khối của Nios II

Nios II có thể được cấu hình nhờ SOPC Builder (các phiên bản Quartus II cũ) hoặc Qsys (các phiên bản Quartus II mới) ở một trong ba loại cấu hình sau:

- Nios II/f: là phiên bản nhanh (fast) cho các hệ thống Nios II hiệu năng cao. Nó có thể được lựa chọn nhiều cấu hình.
- Nios II/s: là phiên bản chuẩn (standard) yêu cầu ít tài nguyên trong FPGA cho các hệ thống Nios II không cần hiệu năng cao.
- Nios II/e: là phiên bản tiết kiệm (economy) yêu cầu ít tài nguyên trong FPGA, nhưng hạn chế cấu hình cho người dùng.

3.1.2. Ví dụ hệ thống xử lý Nios II

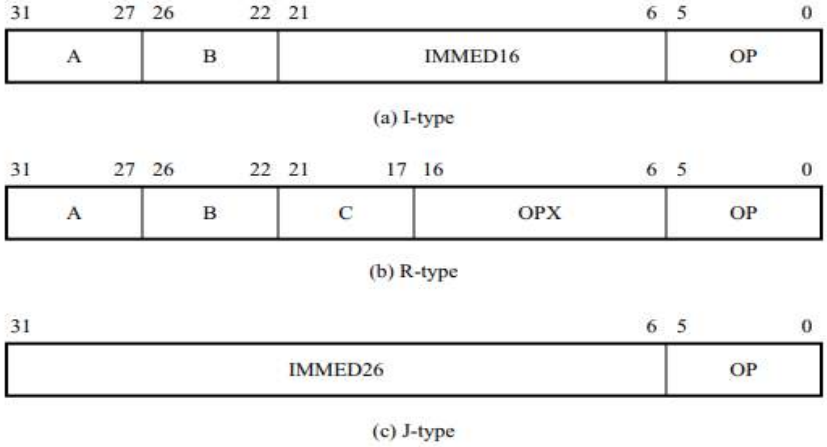
Có nhiều ví dụ thiết kế hệ thống Nios II cho các ứng dụng: xử lý tín hiệu: xử lý tiếng nói, xử lý ảnh, giám sát qua Web server nhúng như các sơ đồ kết nối ứng dụng demo cho ở mục 2.1.2. hay đã đưa ra trong [18][19], nhận dạng vân tay [20], hệ thống giam sát phương tiện giao thông [21], ứng dụng của Ethernet trên đường dây điện [22].

FPGA nói chung và Nios II nói riêng có nhiều ứng trong nhiều lĩnh vực.

3.1.3. Đặc điểm tập lệnh của Nios II

Nios II là chip xử lý có kiến trúc tập lệnh rút gọn (RISC) như hầu hết các chip vi điều khiển cứng hóa khác. Tập lệnh này có không nhiều lệnh và 5 chế độ địa chỉ: Trực

tiếp (immediate mode), chế độ thanh ghi (regisster mode), chế độ bước nhảy (displacement mode), chế độ gián tiếp thanh ghi (register indirect mode), và chế độ tuyệt đối (absolute mode). Nios II có tập thanh ghi gồm 32 thanh ghi chung 32-bit (r0 - r31) và các địa chỉ cũng 32-bit [11]. Tập lệnh gồm các lệnh với 3 loại định dạng: I-type, R-type, và J-type. Có thể tham khảo chi tiết trong tài liệu [11]. Với tất cả các định dạng trường mã lệnh (OP) có 6 bits (hình 3.3). Với R-type: có 3 trường toán hạng 5-bit (A, B, C) , trường OPX 11-bit là mở rộng của trường OP. Với I-type: có hai trường toán hạng 5-bit (A, B), trường toán hạng giá trị trực tiếp IMMED16 16-bit. Với J-type: chỉ có một trường toán hạng giá trị trực tiếp IMMED26 26-bit.



Hình 3.3: Các định dạng của các lệnh của Nios II

3.2. THIẾT KẾ HỆ NHÚNG NIOS II VỚI HỆ ĐIỀU HÀNH uCLINUX

3.2.1. Công cụ phần mềm thiết kế Altera quartus II

Phần mềm Quartus II Web Edition 13.0sp1 có phiên bản chạy trên Windows và phiên bản chạy trên Linux.

Để thiết kế hệ nhúng Nios II, có thể chạy phiên bản trên linux. Để thiết kế hệ nhúng có uClinux nên chạy phiên bản cho Linux trên hệ hệ điều hành Centos hoặc ubuntu.

Có thể tải miễn phí Altera Quartus II Web Edition 13.0sp1 bản nén .zip chạy trên Windows 7 Pro.

Trên PC với ubuntu 12.04 LTS có thể tải Altera Quartus II Web Edition 13.0sp1 bản nén tar cho linux.

3.2.2. Các bước thiết kế hệ nhúng Nios II và hiển thị kết quả

1) Tạo hệ nhúng Nios II

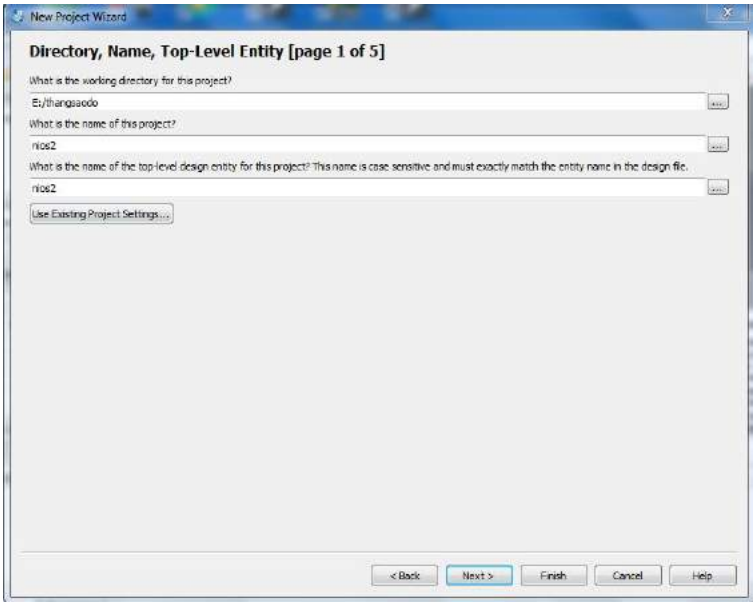
-Cài đặt Quartus II Web Edition 13.0sp1:

Có thể cài trên bất kỳ ổ đĩa nào, học viên đặt thư mục cho cài đặt:

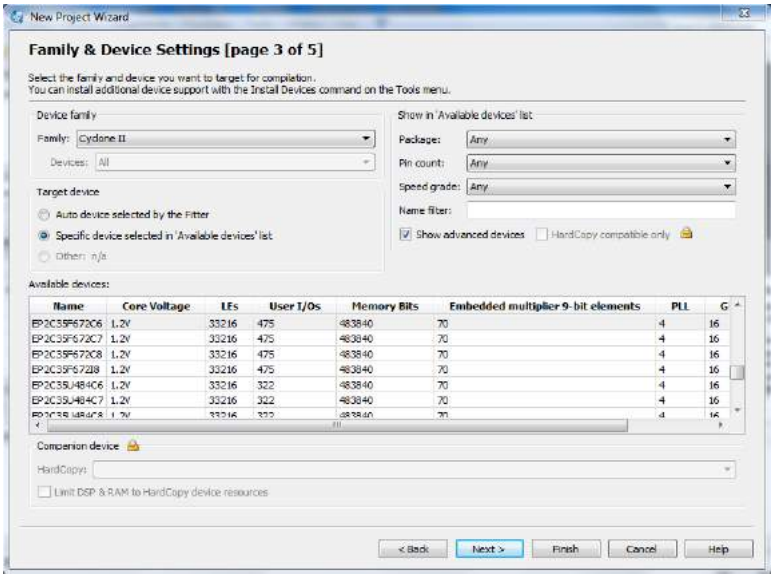
E:\altera\13.0sp1\...

-Chọn biểu tượng Quartus II 13.0sp1 trên Desktop chạy Quartus

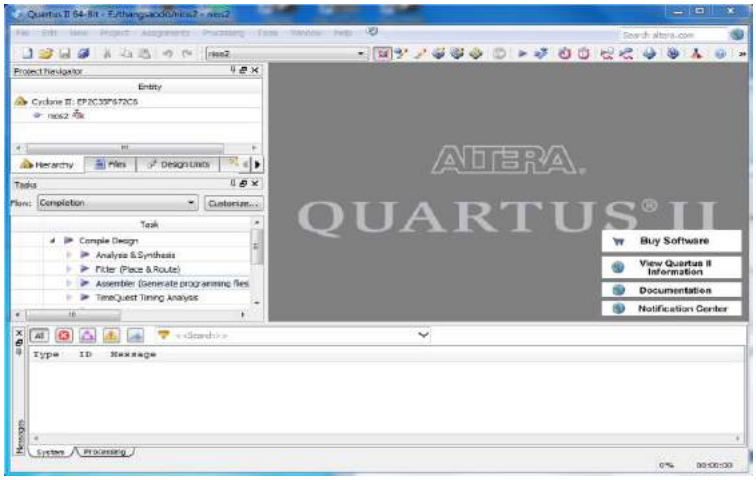
-Tạo tên project nios2:



Hình 3.4: Tạo project nios trong thue mục E:\thangsaodo



Hình 3.5: Chọn loại chip FPGA EP2C35F672C6 cho Nios2

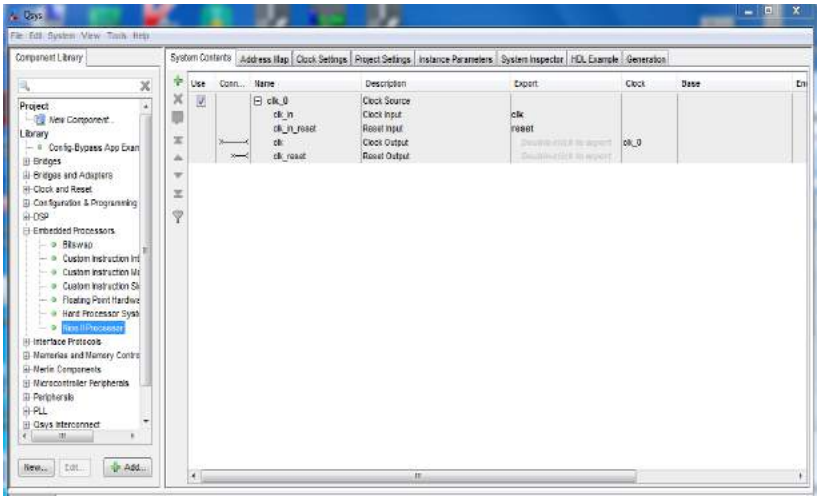


Hình 3.6: Kết quả tạo tên Project nios2

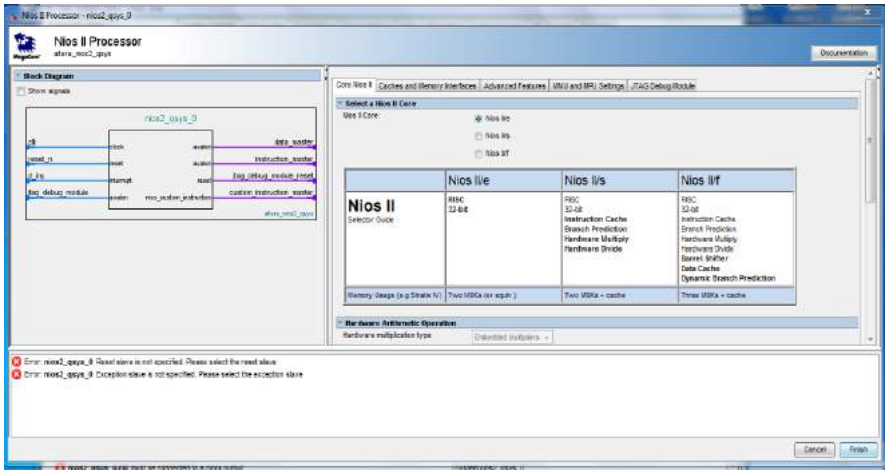
- Chọn Tools -> Qsys:

Hình 3.7: Chạy Qsys

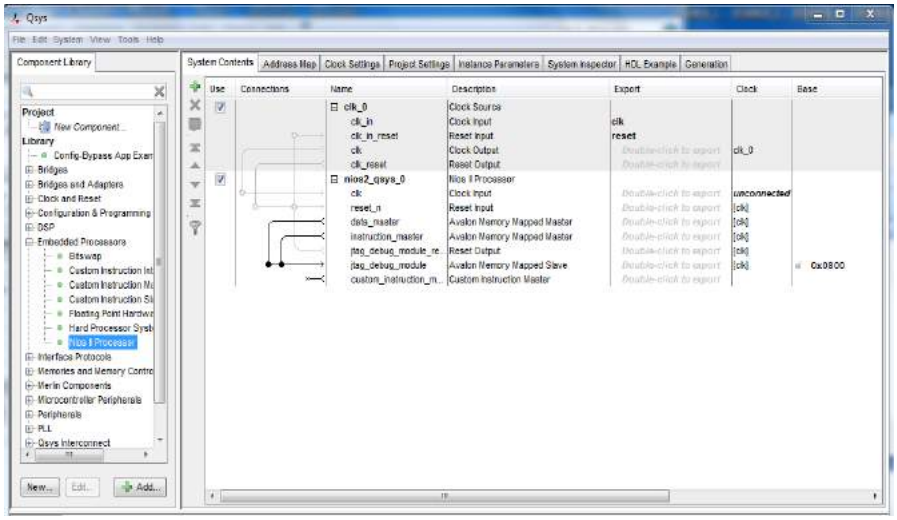
- Trên cửa sổ Qsys, chọn Component Library -> Embedded Processors -> Nios II Processor -> add: tạo nios2:



Hình 3.8: Chọn tạo Nios2

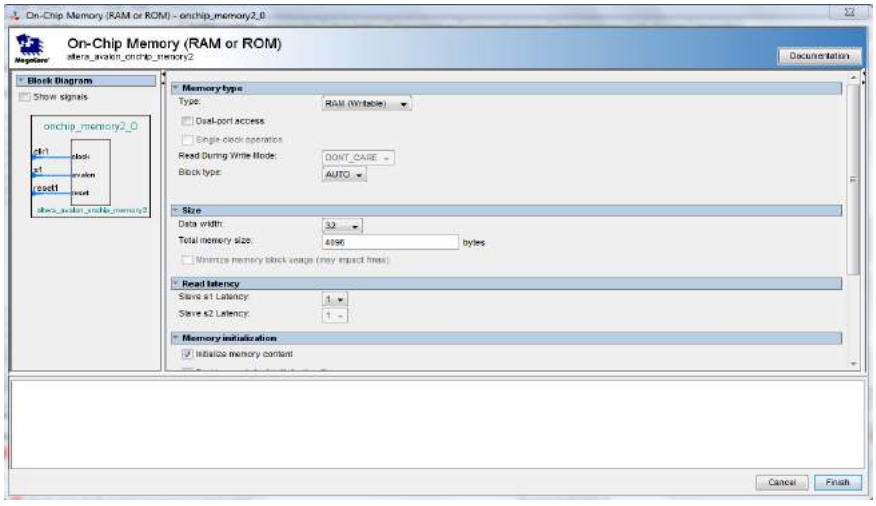


Hình 3.9: Chọn loại cấu hình tiết kiệm Nios II/e cho Nios2



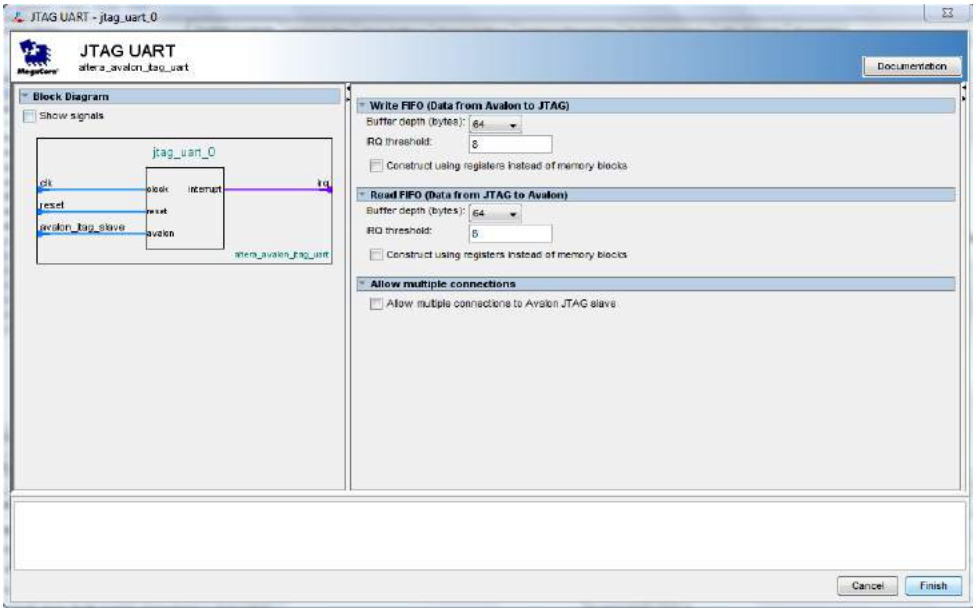
Hình 3.10: Kết quả chọn Nios2

- Chọn: Component Library -> Memory and Memory Controllers -> On-Chip -> On-Chip Memory (RAM or ROM) -> add: tạo Memory trong chip



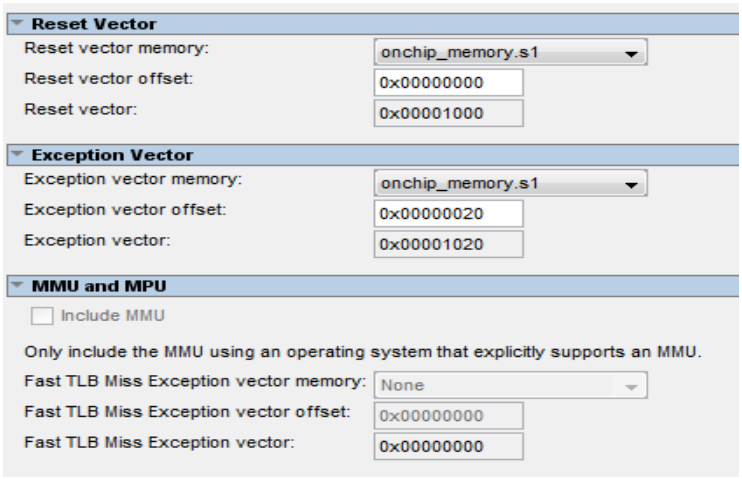
Hình 3.11: Chọn cấu hình cho Memory On Chip

- Chọn: Component Library -> Interface Protocols -> Serial -> JTAGUART -> add



Hình 3.12: Chọn cấu hình cho JTAGUART

- Chọn lại Nios 2 để đặt lại : Reset vector memory và Exception vector memory: onchip_memory.s1:



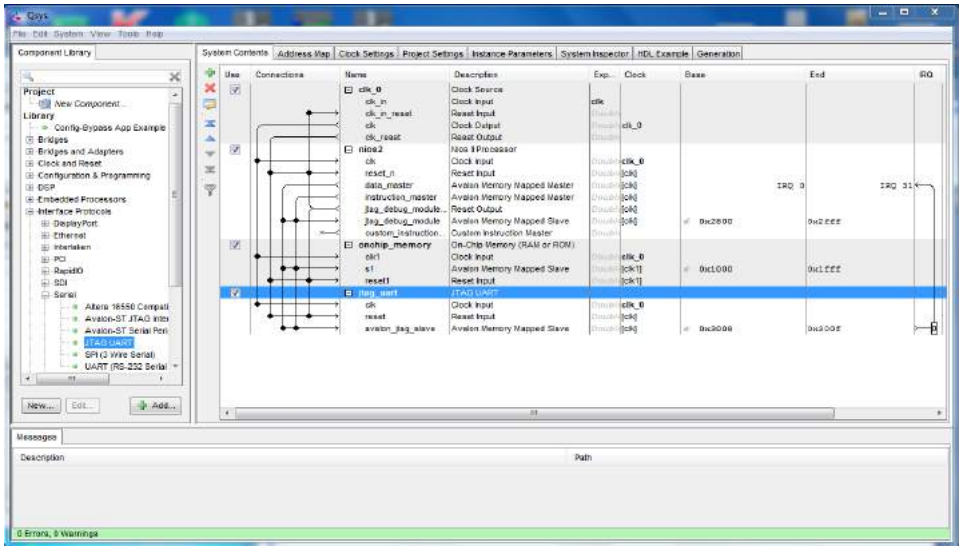
Hình 3.13: Đặt lại Nios2

- Chọn System -> Assign Base Address

-> Assign Interrupt Numbers

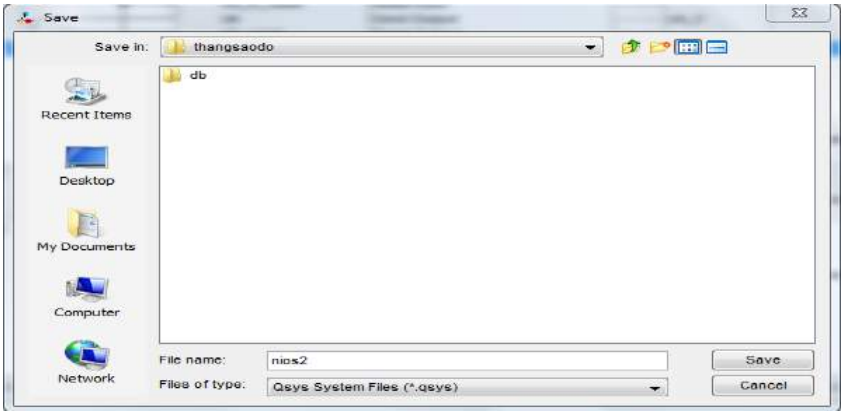
-> Create Global Reset Network

Có cấu hình hệ nhúng Nios II như sau:



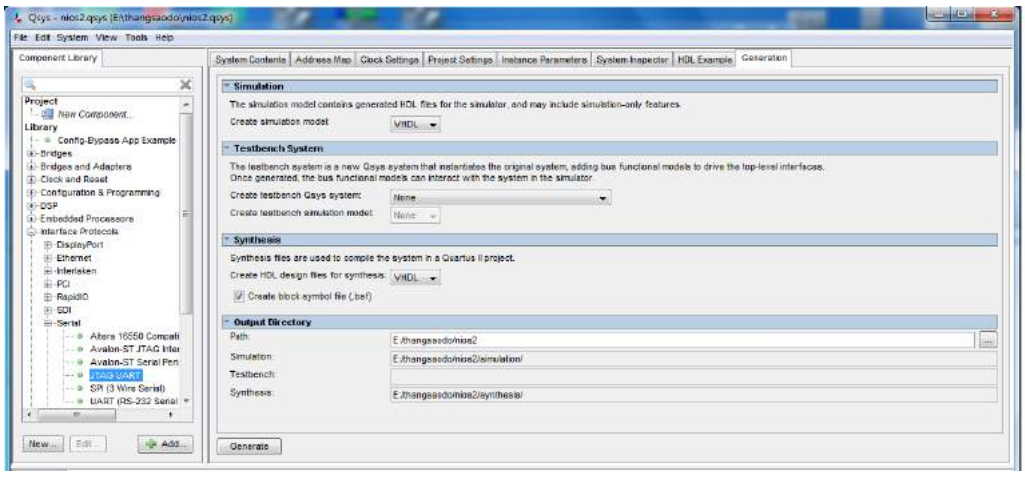
Hình 3.14: Cấu hình hệ nhúng Nios2

- Chọn File -> Save As: ghi file nios2.qsys vào thư mục của project: E:\thangsaodo\:

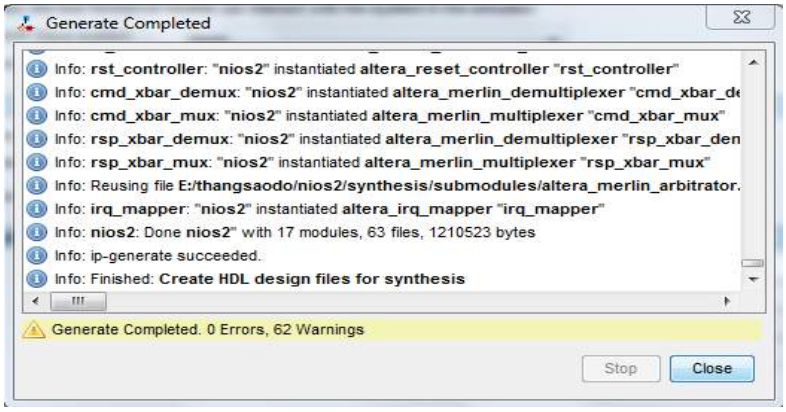


Hình 3.15: Ghi vào file nios2.qsys

- Chọn Generate để tạo project nios2

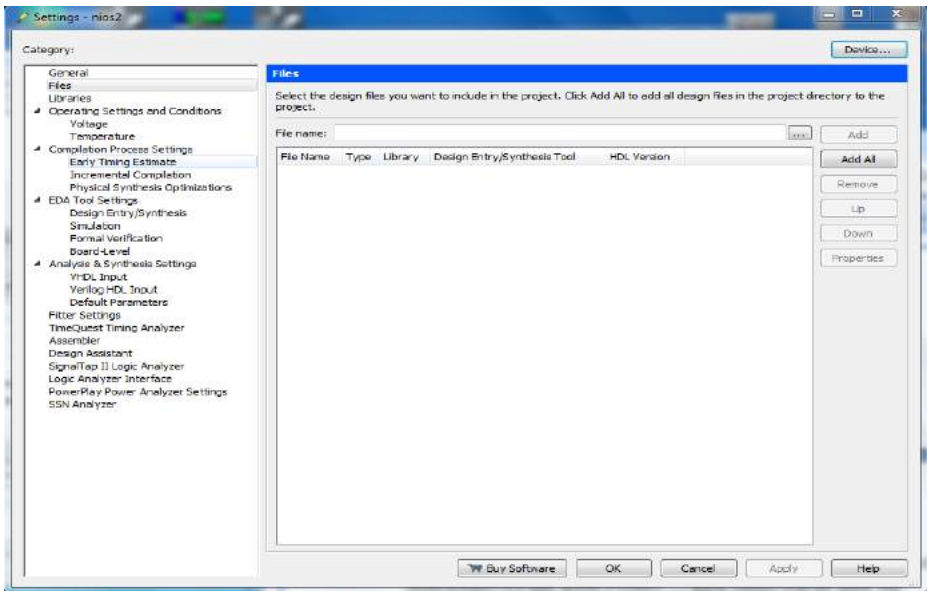


Hình 3.16: Chọn Generate để tạo project nios2

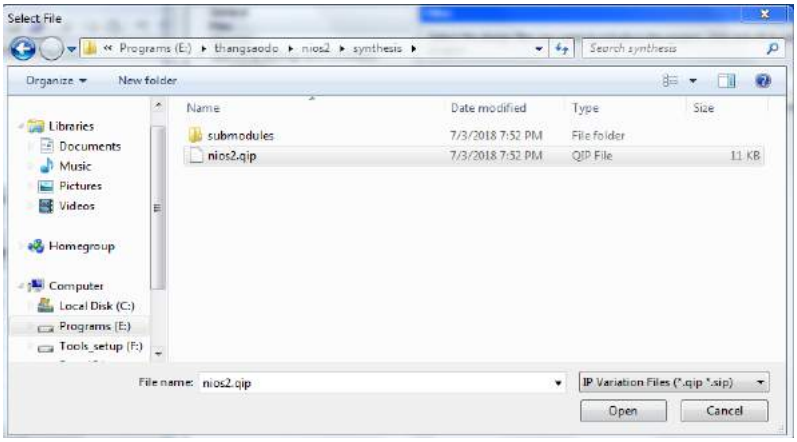


Hình 3.17: Tạo thành công project nios2

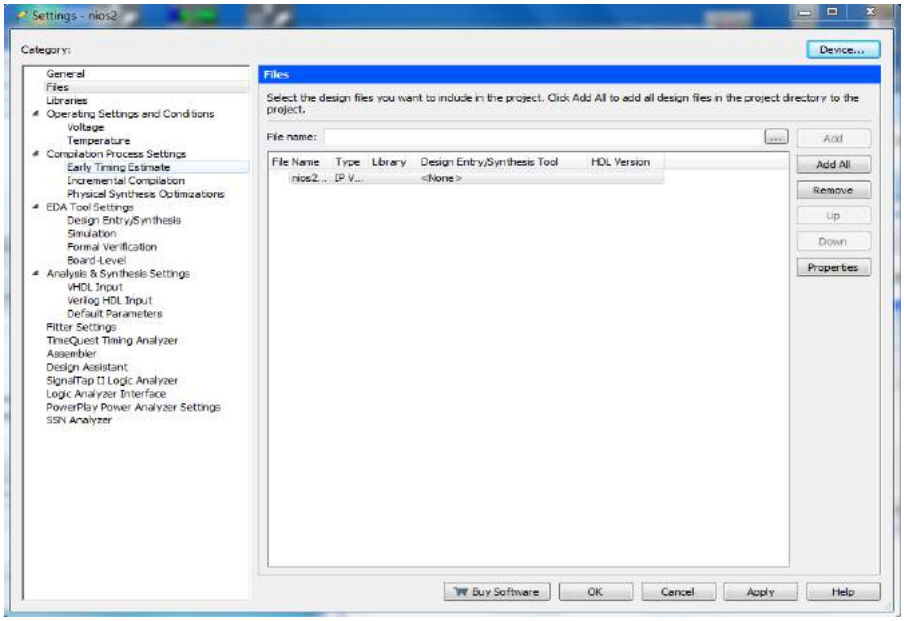
- Trên cửa sổ Quartus II, chọn: Project Navigator -> files -> kích chuột phải: Add/Remove Files from Project ..., kích chuột trái để hiển thị cửa sổ sau:



Hình 3.18: Chọn và bổ xung file cho project nios2

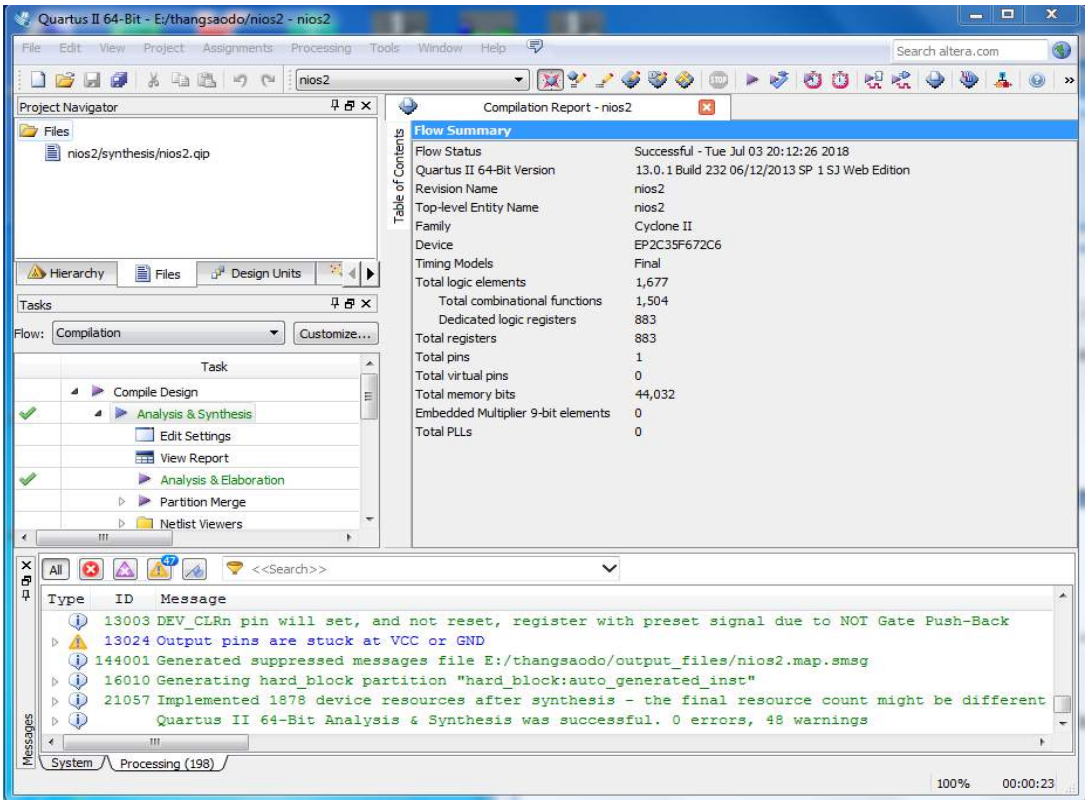


Hình 3.19: Chọn file nios2.qip



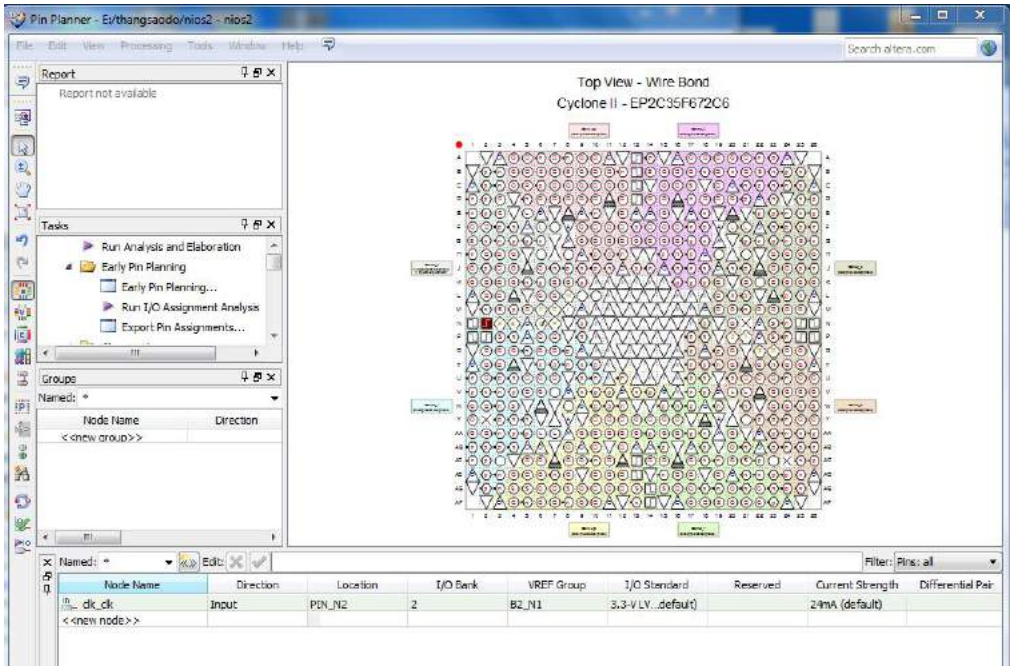
Hình 3.20: Bổ xung file nios2.qip

- Trên cửa sổ Quartus II, chọn Compile Design -> Analysis & Synthesis -> Start



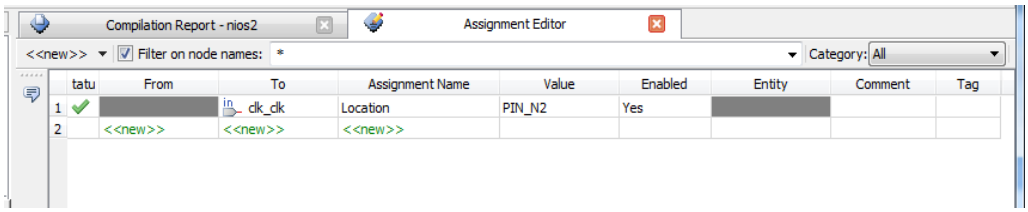
Hình 3.21: Kết quả Analysis & Synthesis thành công

- Trên cửa sổ Quartus II, chọn Assignments -> Pin Planner để gán chân tín hiệu cho Nios II trong FPGA:



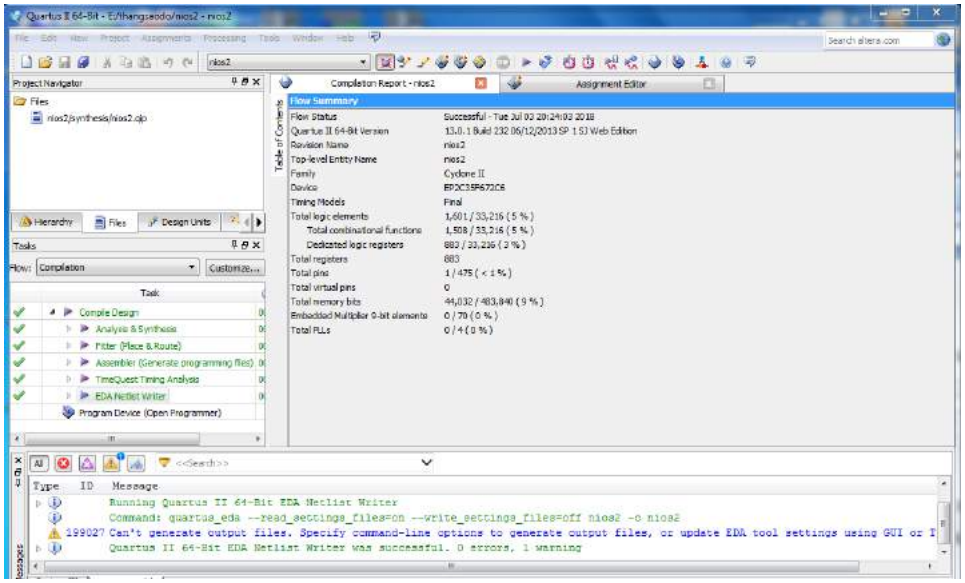
Hình 3.22: Đặt chân tín hiệu clk_clk với PIN_N2 cho clk_0 của nios2

Có thể chọn Assignments -> Assignment Editor để kiểm tra thiết lập pin:



Hình 3.23: Đã thiết lập chân nối tín hiệu clk_clk

- Chọn Compile Design -> Start để thực hiện biên dịch toàn bộ project nios2:

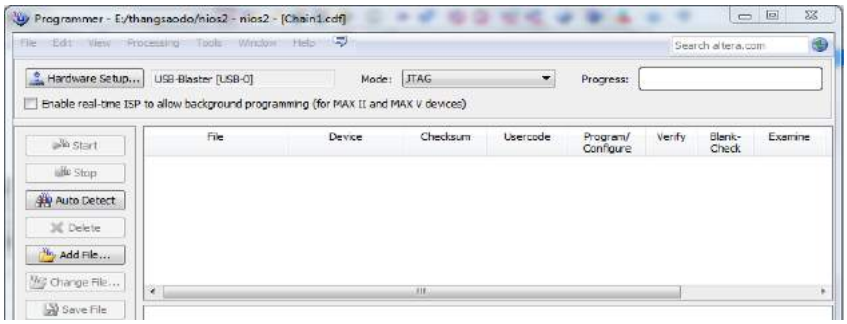


Hình 3.24: Biên dịch thành công project nios2

Chọn file -> Save Project

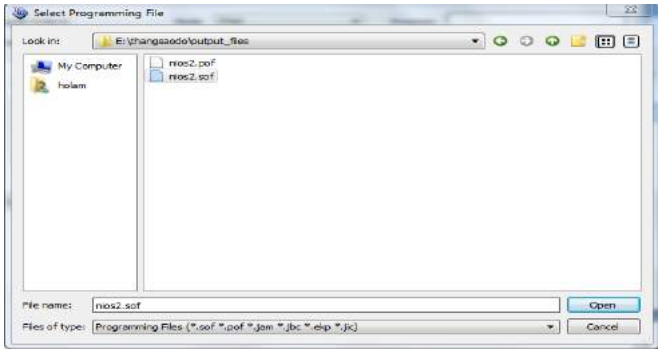
- Nối bảng Altera DE2 với PC bằng cáp lập trình Jtag-usb, bật nguồn cho bảng

- Trên cửa sổ Quartus II, chọn Tools -> Programmer:



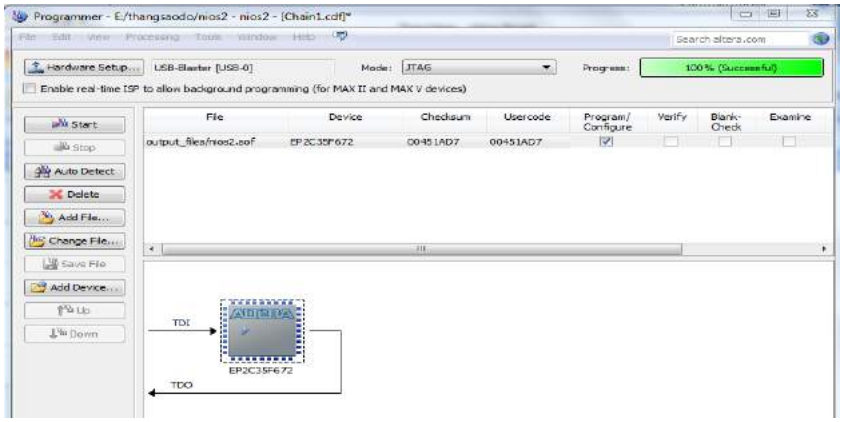
Hình 3.25: Cửa số lập trình cho bảng Altera DE2

- Trên cửa sổ lập trình (Programmer), chọn Add File ..., và trong thư mục \thangsaodo\output_files\ chon file nios2.sof



Hình 3.26: Chọn file nios2.sof

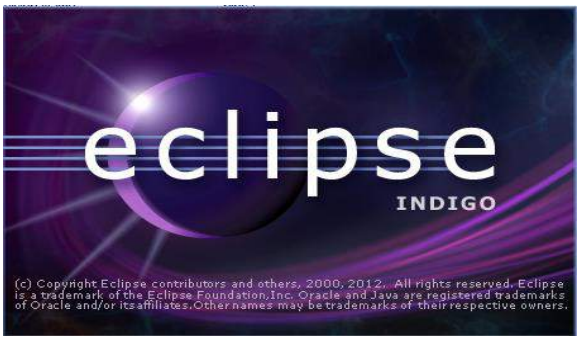
- file nios2.sof sẽ được đưa vào cửa sổ Programmer, tại đây, kích chọn start để nạp file cấu hình hệ thống nhúng nios2 này lên bảng Altera DE2. Các đèn trên bảng sẽ có thay đổi trạng thái khi nạp và không còn nhấp nhánh.



Hình 3.27: Nạp thành công file cấu hình hệ nios2 lên bảng

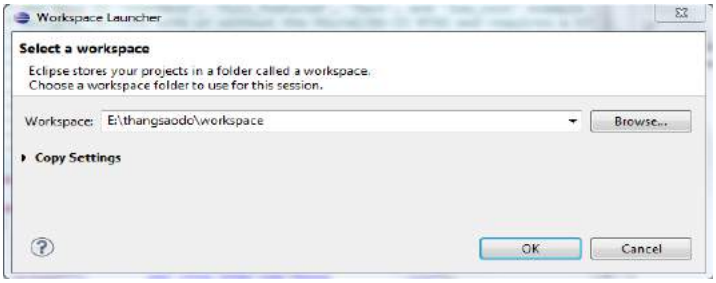
Như vậy, đến đây ta đã tạo thành công hệ nhúng Nios II. Tiếp theo cần tạo một ứng dụng trên C đơn giản để nạp và chạy trên hệ nhúng để kiểm tra hoạt động hệ nhúng này.

- Trên cửa sổ Quartus II, chọn Tools -> Nios II Software Build for Eclipse:



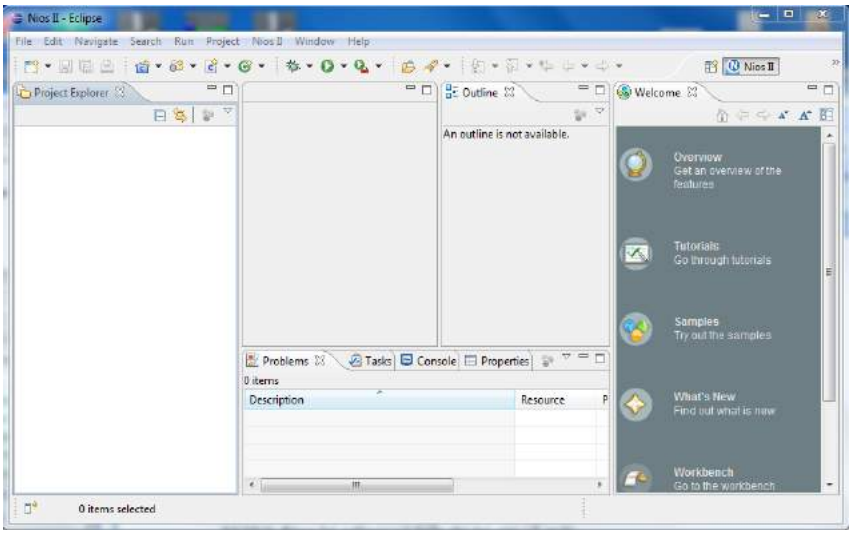
Hình 3.28: Chạy eclipse

- Trên cửa sổ eclipse, chọn file -> Switch Workspace, chọn thư mục \thangsaodo\workspace:



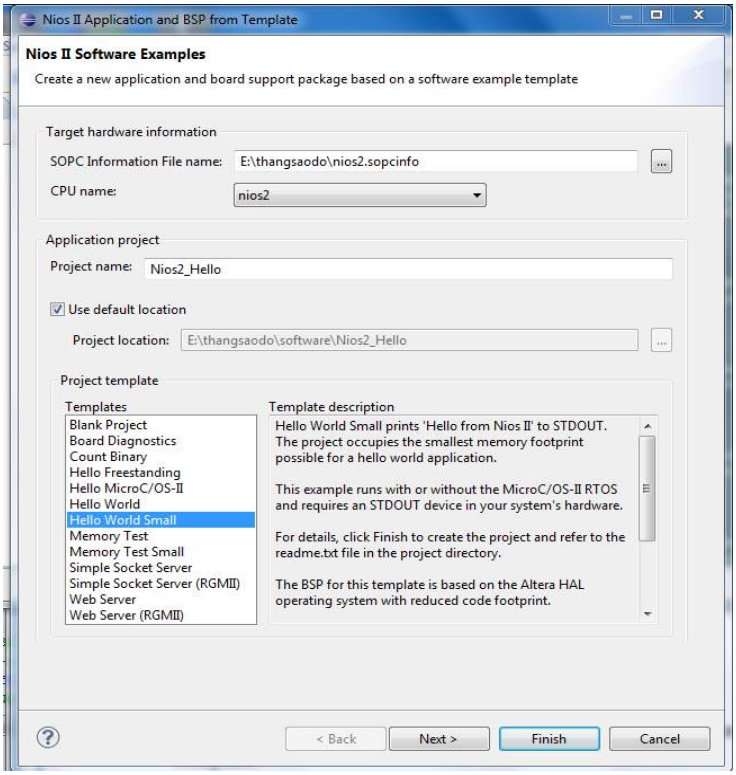
Hình 3.29: Chọn vùng làm việc cho ứng dụng của project nios2

Sẽ khởi động lại eclipse và hiển thị lại cửa sổ mới:



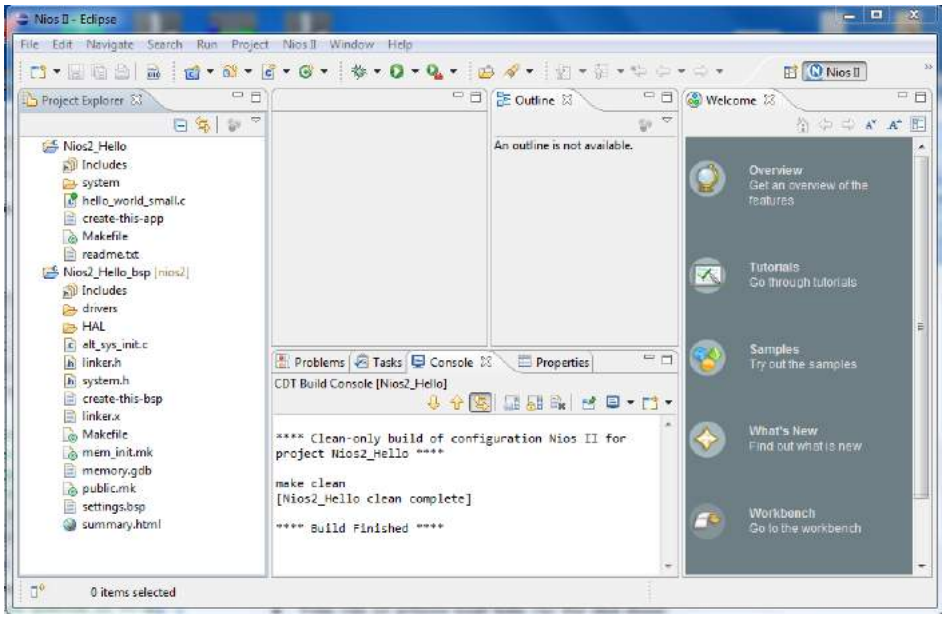
Hình 3.30: Cửa sổ eclipse cho tạo ứng dụng mới của project nios2

- Trên cửa sổ eclipse, chọn file -> New -> Nios II Application and BSP from Template. Trên cửa sổ Nios II Application and BSP from template, chọn file nios2.sopcinfo trong \thangsaodo, CPU name: nios2, và đặt tên cho phần mềm ứng dụng Nios2_Hello, kích chọn Next, tiếp chọn Finish



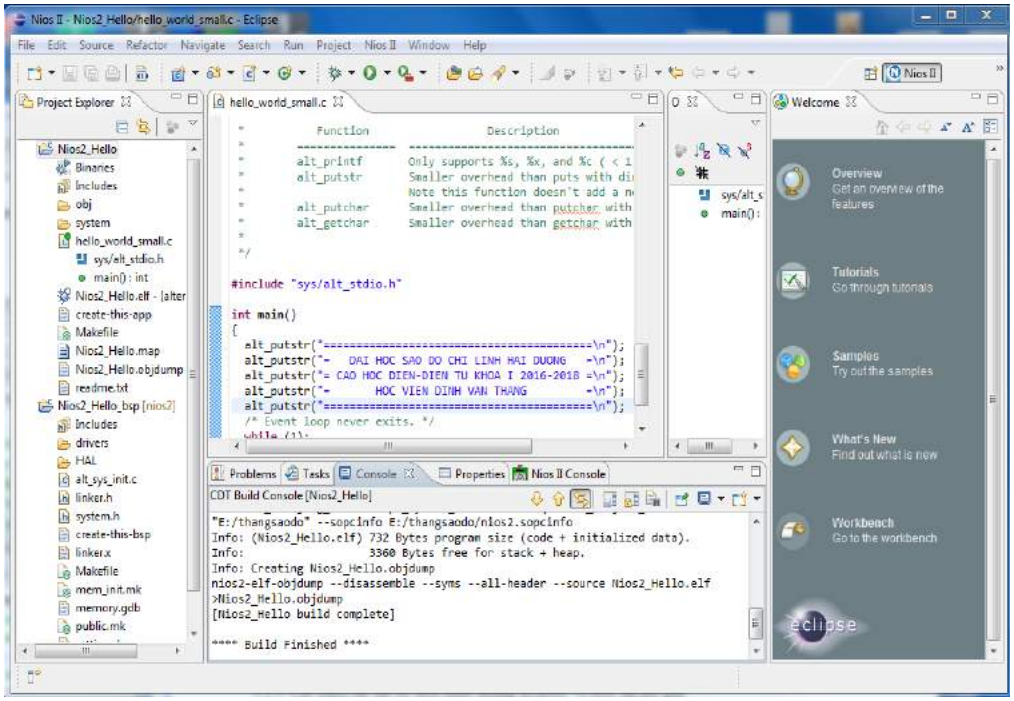
Hình 3.31: Chọn vào tạo ứng dụng Nios2_Hello

- Trên cửa sổ eclipse xuất hiện các file ứng dụng:



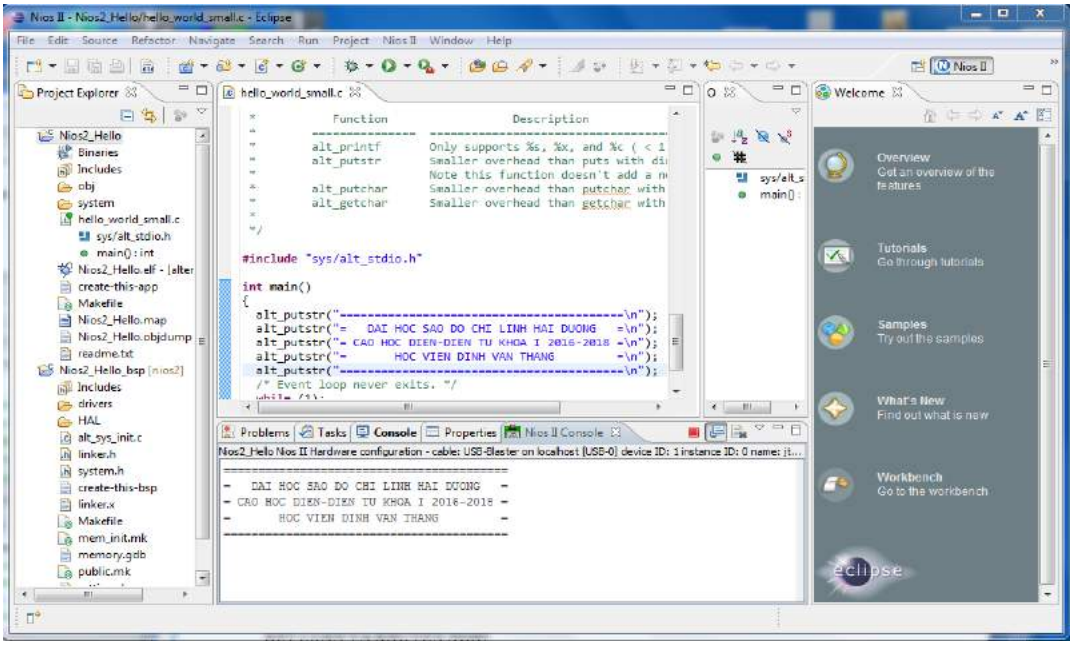
Hình 3.32: Các file của ứng dụng phần mềm Nios2_Hello

- Biên soạn lại nội dung phần mềm Nios2_Hello.c, trong cửa sổ Project Explorer, chọn Nio2_Hello bằng kích chuột phải, chọn Build Project để biên dịch file Nios2_Hello.c:



Hình 3.33: Biên dịch thành công

- Chọn Nios2_Hello -> Run As -> Nios II Hardware



Hình 3.34: Nạp ứng dụng Nios2_Hello lên hệ nhúng Nios2 và chạy thành công

Đến đây, ta có thể kết luận rằng thiết kế hệ nhúng Nios 2 trên bảng Altera DE2 2C35 FPGA và cài đặt một ứng dụng phần mềm nhỏ đa thành công.

3.2.3. Các bước cài đặt hệ điều hành nhúng uClinux và hiển thị kết quả

1) Cài đạt Altetra Quartus II Web Edition 13.0sp1:

- Trước hết cần cài ubuntu 12.04 LTS trên PC

- Tải Quartus II Web Edition 13.0sp1 và gỡ nén trong /home/thangsaodo

- Cài đặt Web Edition 13.0sp1 trong /home/thangsaodo/altera/13.0sp1:

Hình 3.35: Thư mục của Altera Quartus Web Edition 13.0sp1

2) Cài đặt nios2gcc:

- Tải nios2gcc-20080203.tar.bz2 về /home/thangsaodo

- Gỡ nén về /opt/nios2:

$ sudo jxf nios2gcc-20080203.tar.bz2 -C /

3) Cài đặt nhân uClinux:

- Tải các phần mềm của uClinux về /home/thangsaodo:

uClinux-dist-20070130.tar.bz2, uClinux-dist-20070130-nios2-02.diff.gz

- Gỡ nénuClinux-dist-20070130.tar.bz2 vào /home/thangsaodo/uClinux-dist:

$ tar jxf uClinux-dist-20070130.tar.bz2

- Sửa lỗi bản uClinux-dist:

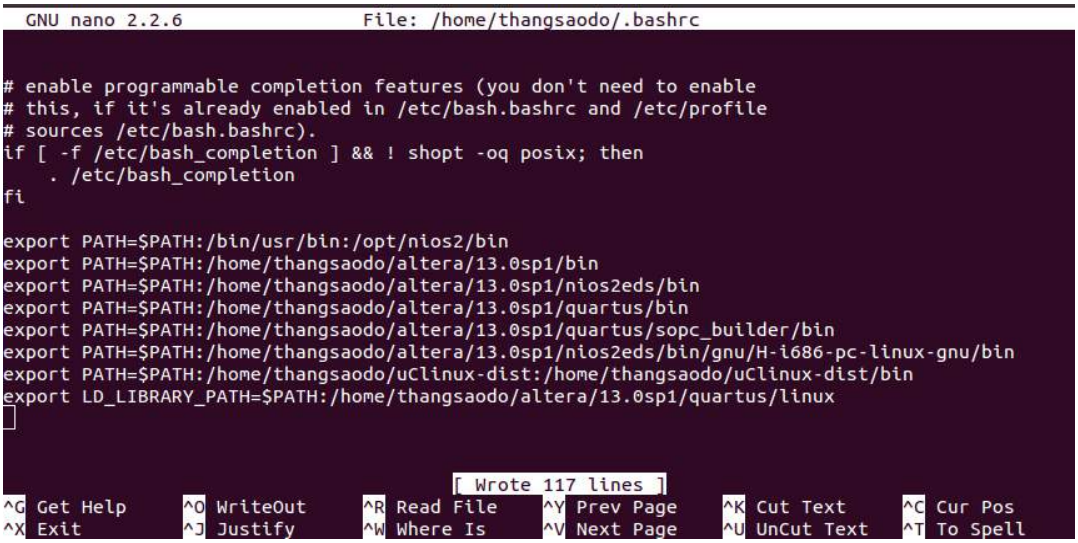
$ cp uClinux-dist-20070130-nios2-02.diff.gz uClinux-dist
$ cd uClinux-dist

$ gunzip -c uClinux-dist-20070130-nios2-02.diff.gz | patch -p0

4) Tạo môi trường:

- Sử dụng trình soạn thảo nano thường trú của ubuntu 12.04 LTS:

$ nano .bashrc
Bổ xung các dòng môi trường cho file .bashrc trong /home/thangsaodo:



Hình 3.36: Các dòng lệnh môi trường bổ xung trong .bashrc

5) Cài USB Blaster cho ubuntu:

- Tạo file 51-usbblaster.rules trong /etc/udev/rules.d

$ sudo nano /etc/udev/rules.d/51-usbblaster.rules với nội dung sau:

USB-Blaster

BUS=="usb", SYSFS{idVendor}=="09fb", SYSFS{idProduct}=="6001",

MODE="0666"

BUS=="usb", SYSFS{idVendor}=="09fb", SYSFS{idProduct}=="6002",

MODE="0666"

BUS=="usb", SYSFS{idVendor}=="09fb", SYSFS{idProduct}=="6003",

MODE="0666"

- Để có hiệu lực cho kết nối USB Blaster, cần phải khởi động lại PC ubuntu, hoặc vào lệnh:

$ sudo udevadm control --reload

6) Tạo file n2sdk để khởi tạo kết nối với bảng Altera DE2

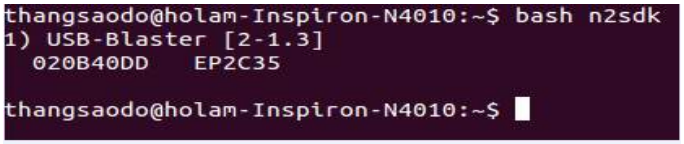
$ nano n2sdk và đưa vào dòng lệnh:

/home/thangsaodo/altera/13.0sp1/quartus/bin/jtagconfig

7) Kiểm tra kết nối PC ubuntu với bảng Altera DE2:

- Nối bảng Altera DE2 với PC qua cổng USB Blaster và cáp USB, bật nguồn cho bảng

- Gõ lệnh bash n2sdk: nếu kết nối tốt thì phải hiển thị trên ubuntu:



Hình 3.37: Hiển thị kết nối bảng Altera DE2

8) Bổ xung một số module cho ubuntu 12.04 LTS:

Để chạy file 32-bit trên ubuntu 12.04 LTS 64-bit cần phải bổ xung ba gói thư viện libc6:i386, libncurses5:i386, and libstdc++6:i386:

$ sudo apt-get update

$ sudo apt-get install libc6:i386libncurses5:i386libstdc++6:i386

Và bổ xung một số gói sau cho ubuntu 12.04 LTS:

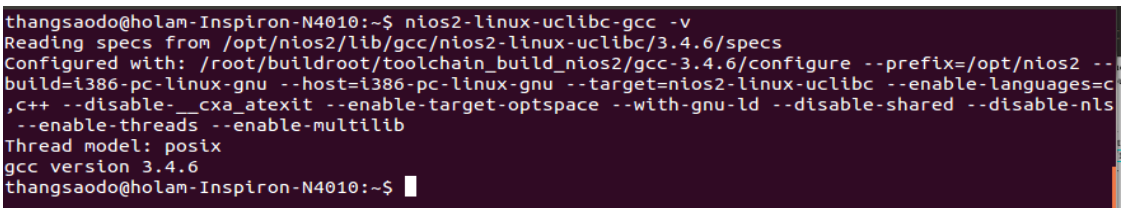
$ sudo apt-get install git-core git-gui make gcc ncurses-dev bison flex gawk \

gettext ccache zlib1g-dev libx11-dev texinfo liblzo2-dev pax-utils uboot-mkimage corkscrew

9) Kiểm tra cross gcc:

$ nios2-linux-uclibc-gcc -v

Trên ubuntu phải hiển thị:

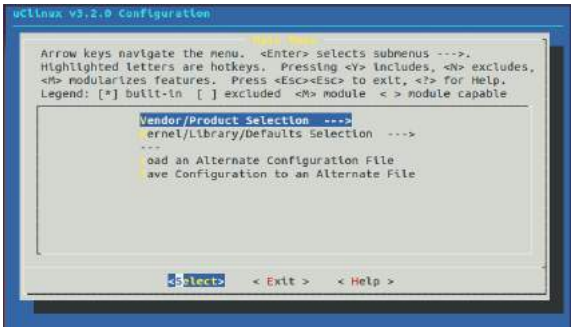


Hình 3.38: Hiển thị phiên bản gcc

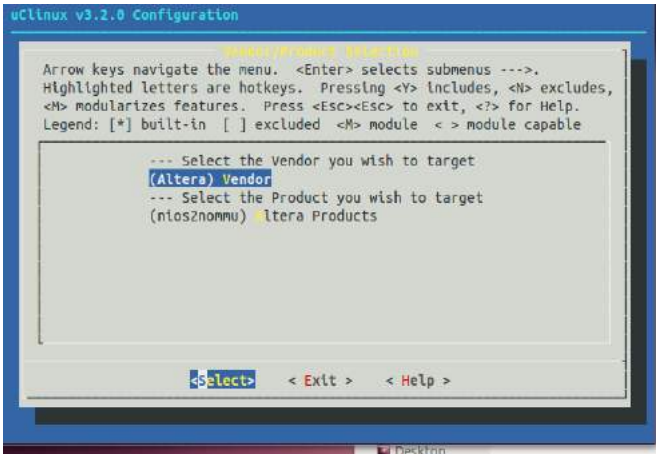
10) Cấu hình cho uClinux

$ cd uClinux-dist

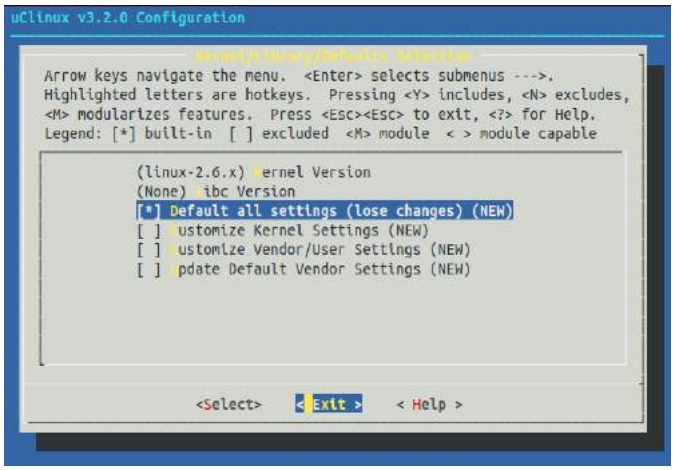
$ make menuconfig



Hình 3.39: chọn nhà sản xuất



Hình 3.40: chọn Altera với nios2 không có MMU



Hình 3.41: chọn cấu hình mặc định cho uClinux

$ make vendor_hwselect SYSPTF = /home/thangsaodo/DE2_System_v1.6_bk/DE2_demonstrations/DE2_NET/system_0.ptf

Sẽ phải lựa chọn: CPU (1), Flash (1), và SDRAM 8MB (2)

$ make romfs

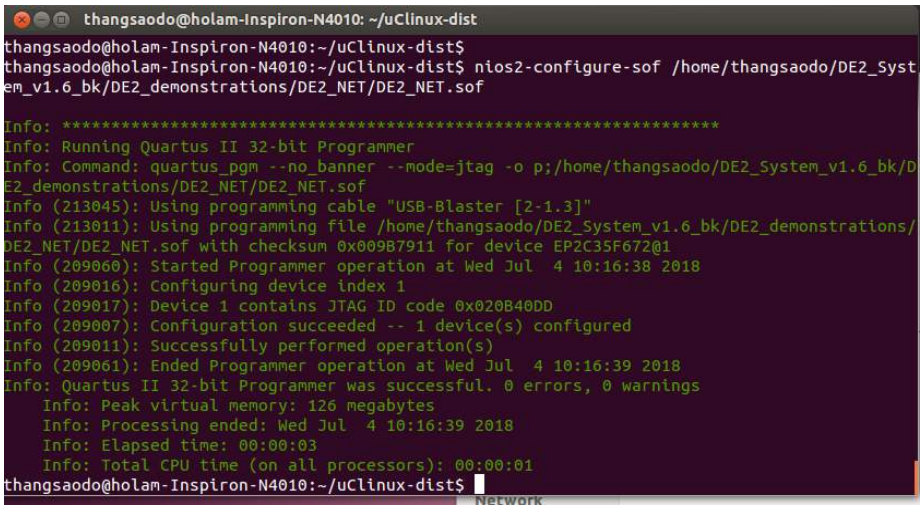
$ make

Tạo image

$ make linux image

$ nios2-configure-sof /home/thangsaodo/DE2_System_v1.6_bk/DE2_demonstrations/DE2_NET/DE2_NET.sof

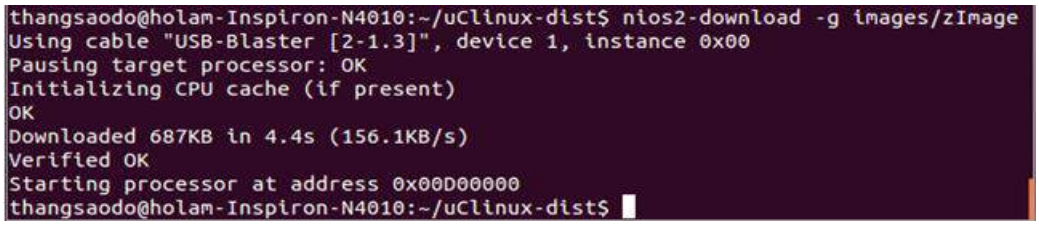
Tải cấu hình hệ thống nios2 lên bảng Altera DE2



Hình 3.42: Nạp cấu hình thành công lên bảng Altera DE2

$nios2-download -g images/zImage

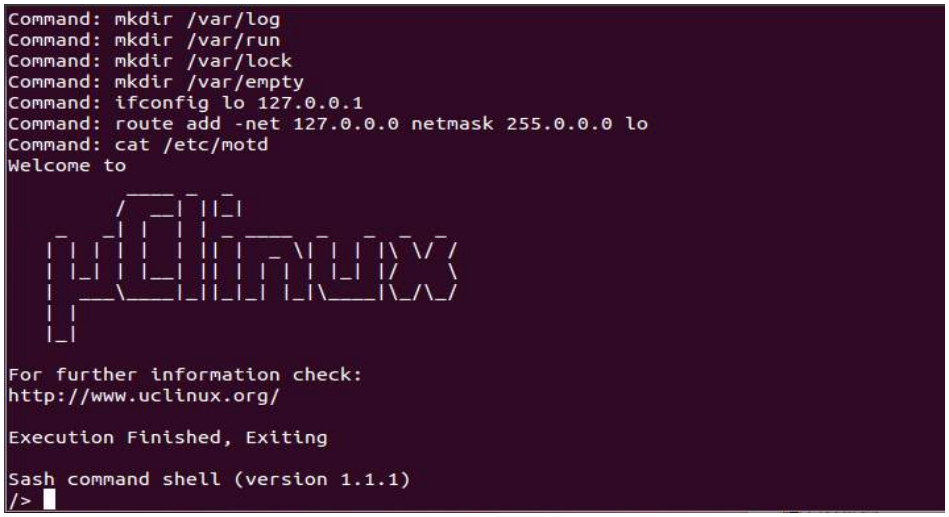
Tải file ảnh zImage của uClinux trong /home/thangsaodo/uClinux-dist/images lên bảng altera DE2 đã có system nios.



Hình 3.43: Nạp thành công uClinux lên bảng Altera DE2 với nios2

$ nios2-terminal

Kết nối với hệ thống nios2+uClinux trên bảng Altera DE2:



Hình 3.44: Đáp ứng trả về PC (terminal) từ Nios2+uClinux

Đến đây, quá trình tạo hệ thống nhúng Nios2 + uClinux trên bảng Altera DE2 2C35 FPGA đã đạt kết quả thành công.

Có thể xem uClinux có các thư mục nhờ lệnh ls:

Hình 3.45: Các thư mục của uClinux

Có thể soạn thảo một ứng dụng phần mềm trên PC ubuntu, biên dịch và bổ xung vào file ảnh của uClinux, sau đó tải lên bảng Altera DE2.

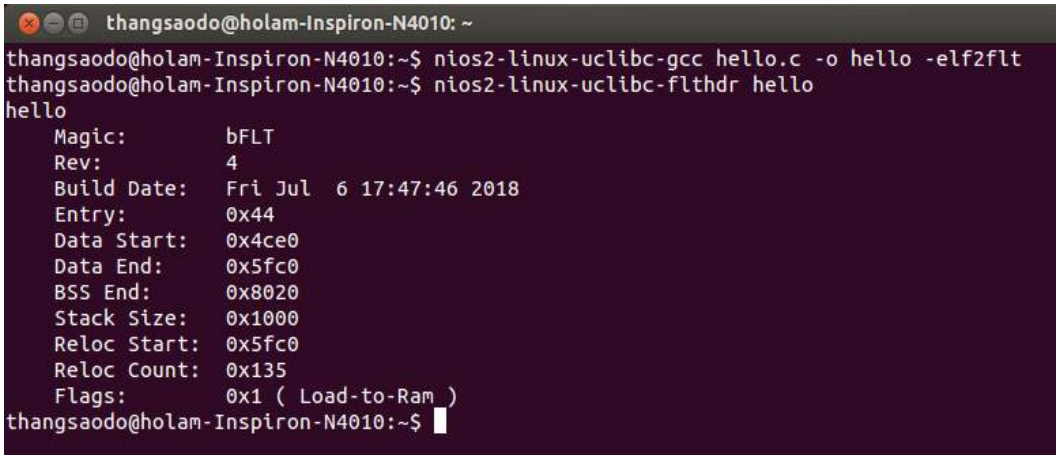
3.2.4. Tạo phần mềm ứng dụng để thử nghiệm hệ nhúng

1) Soạn một phần mềm đơn giản trên PC ubuntu:

$ nano hello.c với nội dung:

Hình 3.46: Nội dung file ứng dụng hello.c

2) Biên dịch và kiểm tra:



Hình 3.47: Biên dịch và kiểm tra kết quả biên dịch

3) Sao chép ứng dụng đã biên dịch ra nhị phân:

$ cp hello /home/thangsaodo/uClinux-dist/romfs/bin

Tạo lại ảnh cho uClinux

$ cd uClinux-dist

$ make linux image

Đến đây ta nạp lại ảnh uClinux (đã có kèm ứng dụng) lên bảng Altera DE2, và thực hiện lệnh nios2-terminal, sẽ nhận được hình ảnh đáp ứng hello từ uClinux.

3.3. KẾT LUẬN CHƯƠNG

Chương 3 đã trình bày tổng quan các bước thực hiện một hệ thống nhúng Nios II đơn giản với hệ điều hành uClinux. Altera cũng cung cấp một số ứng dụng về cài đặt hệ nhúng nios II và uClinux song có rất nhiều lỗi, bắt buộc khi cài đặt phải thực hiện vá lỗi các file nguồn tải về từ Altera. Các vá lỗi này nhiều và không thể trình bày hết được. Để thiết kế một hệ nhúng nios II phức tạp cần phải nhiều bước cấu hình các thiết bị và giao tiếp của nios II, đồng thời phải lựa chọn bản uClinux phù hợp. Trong khuôn khổ một chương thiết kế, học viên chỉ trình bầy thiết kế đơn giản nhưng phù hợp đề cương của luận văn đặt ra.

KẾT LUẬN VÀ KIẾN NGHỊ

Chúng ta đã biết đến các hệ thống nhúng với các họ vi điều khiển 8051,PIC, AVR, ARM, AVR32 hay pSoC. Luận văn đã đề cập đến FPGA và các hệ nhúng trên FPGA, cách thiết kế hệ nhúng trên FPGA như thế nào. FPGA có thể sẽ mạnh mẽ hơn và có lợi ích hơn nếu nó được ghép nối với chip vi điều khiển. Đó là một cách mà nhiều nhà thiết kế thường mong muốn đạt được cho nhiều ứng dụng cũng như các nghiên cứu, và gọi đây là co-design FPGA + vi điều khiển (MCU).Trong thực tế, có thể ghép nối FPGA với MCU ngoài bằng các chân giao tiếp thông thường. Một cách khác là sử dụng MCU nhúng có sẵn trong FPGA một hoặc nhiều MCU cứng hoặc mềm). Với Altera là Nios, Nios II (MCU mềm) hay Excalibur (MCU cứng với lõi ARM922T). Còn trong các FPGA của Xilinx là Picoblaze, Microblaze (MCU mềm) hay PowerPC (cứng). MCU cứng được thiết kế cứng sẵn trong FPGA, còn MCU mềm thực chất là một IP thường bằng VHDL và công cụ thiết kế sẽ triển khai IP đó xuống FPGA. Chính vì vậy MCU mềm sẽ tiêu tốn một phần tài nguyên của FPGA, nhưng bù lại nó có thể được cập nhật, thay đổi cấu hình và chức năng tuỳ biến theo các phiên bản khác nhau và ứng dụng khác nhau.

Trên cơ sở tìm hiểu cách tạo hệ nhúng trên FPGA với nhân là lõi vi xử lý mềm 32-bit mà nội dung luận văn đã trình bày, ta có thể kết hợp ghép nối với hệ vi điều khiển với các chip vi điều khiển, mở rộng bộ nhớ bán dẫn (SRAM, DRAM), các thiết bị ngoại vi xử lý các tín hiệu video, audio, data: camera, sensor, DVD, microphone, VGA monitor, v.v... tạo các hệ thống nhúng phức tạpcó các kết nối truyền thông: ethernet, wifi, mobile communication, Internet.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Hồ Khánh Lâm, "Giáo trình Lập trình VHDL thiết kế hệ thống số trên FPGA". NXB KHKT, 2015.

Tiếng Anh

[2] Prabhakar R, Thirumal Murugan J and Praveenkumar B," Efficient Method for Controlling Electric Power by Automated Monitoring System using FPGA". International Journal of Engineering Research & Technology (IJERT) ISSN:

2278- 0181 Vol. 3 Issue 11, November-2014.

[3] HAYASHI Toshifumi et. al."Application of FPGA to nuclear power plant I&C systems". Nuclear Safety and Simulation, Vol. 3, Number 1, March 2012. Received date: March 13, 2012 (Revised date: April 12, 2012).

[4] Lauri Lötjönen," Field-Programmable Gate Arrays in Nuclear Power Plant Safety Automation". School of Electrical Engineering. Thesis submitted for examination for the degree of Master of Science in Technology.

[5] IAEA Nuclear Energy Series No. NP-T-3.17," Application of Field Programmable Gate Arrays in Instrumentation and Control Systems of Nuclear Power Plants".

[6] Usama Bin Rehan et. al.,"Power Line Control and Monitoring Using FPGA". 2017 2nd International Electrical Engineering Conference (IEEC 2017) May 19th - 20th, 2017 at IEP Centre, Karachi, Pakistan.

[7] "The implementation of digital protection in power system using FPGA".ASICON 2001. 2001 4th International Conference on ASIC Proceedings (Cat.No.01TH8549)

[8] "Designing Home Appliances With FPGAs".www.altera.com,"Nios II Classic Processor Reference Guide".www.altera.com. WP-01027-1.0. 06/2007.

[9] Andrei Cozma and Eric Cigan**,**" FPGA-Based Systems Increase Motor-Control Performance".analog.com/analogdialogue. Analog Dialogue 49-03, March 2015

[10] "Hệ thống nhúng". https://vi.wikipedia.org/wiki/Hệ_thống_nhúng

[11] "Instroduction to the Altera Nios II Soft Processor".

https://people.ece.cornell.edu/land/courses/ece5760/DE2/tut_nios2_introduction.pdf

[12] "Altera Quartus". https://en.wikipedia.org/wiki/Altera_Quartus

[13] https://www.altera.com/en_US/pdfs/literature/hb/nios2/n2cpu-nii5v1gen2.pdf

[14] www.Altera.com,"Designing with the Nios II Processor and SOPC Builder: Exercise Manual".

[15] Nguyễn Thị Thảo," Thiết kế Web server an toàn, cấu hình lại được trên FPGA". Luận văn thạc sĩ CNTT ĐHSPKT Hưng Yên. 2017.

[16] "uClinux". http://www.alterawiki.com/wiki/UClinux

[17] https://en.wikipedia.org/wiki/Soft_microprocessor

[18] "Video and Image Processing Design Example". https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/an/an427.pdf

[19] L. Pyrgas, A. Kalantzopoulos* and E. Zigouris," Design and Implementation of an Open Image Processing System based on NIOS II and Altera DE2-70 Board". Journal of Engineering Science and Technology Review 9 (5) (2016) 51 - 55.

[20] Ji Wang, Liang Wu, Yong Liu," Nios II Processor-Based Fingerprint Identification System". Institution: College of Communication Engineering, Chongqing University.

[21] ATIBI MOHAMED et. al." IMPLEMENTATION OF A VEHICLE DETECTION SYSTEM IN THE FPGA EMBEDDED PLATFORM". Journal of Theoretical and Applied Information Technology. ISSN: 1992-8645 www.jatit.org E-ISSN: 1817- 3195.

[22] JAMI ADITYA and PRIYANKA PRIYADARSINI," APPLICATION OF ETHERNET OVER POWERLINE COMMUNICATION". Department of Electronics and Communication Engineering National Institute of Technology, Rourkela May, 2013.