**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN – ĐIỆN TỬ

**BỘ MÔN ĐIỆN TỬ**

---------------o0o---------------

****

**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

**THIẾT KẾ GAME BATTLE TANK TRÊN NỀN FPGA SỬ DỤNG DE1 BOARD**

**GVHD: TS. Trần Hoàng Linh**

**SVTH: Nguyễn Hữu Nghĩa**

**MSSV: 1512157**

**TP. HỒ CHÍ MINH, THÁNG 1 NĂM 2022**

ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

TRƯỜNG ĐẠI HỌC BÁCH KHOA Độc lập – Tự do – Hạnh phúc.

-----✩----- -----✩-----

Số: \_\_\_\_\_\_ /BKĐT

Khoa: **Điện – Điện tử**

Bộ Môn: **Điện Tử**

N**HIỆM VỤ LUẬN VĂN TỐT NGHIỆP**

1. HỌ VÀ TÊN : MSSV:

1. NGÀNH: **ĐIỆN TỬ - VIỄN THÔNG** LỚP :
2. Đề tài:
3. Nhiệm vụ (Yêu cầu về nội dung và số liệu ban đầu):

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

1. Ngày giao nhiệm vụ luận văn: ...............................
2. Ngày hoàn thành nhiệm vụ: ...................................
3. Họ và tên người hướng dẫn: Phần hướng dẫn

................................................................. .....................................

................................................................. .....................................

Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

*Tp.HCM, ngày…... tháng….. năm 20*

**CHỦ NHIỆM BỘ MÔN NGƯỜI HƯỚNG DẪN CHÍNH**

**PHẦN DÀNH CHO KHOA, BỘ MÔN:**

Người duyệt (chấm sơ bộ):.......................

Đơn vị:......................................................

Ngày bảo vệ : ...........................................

Điểm tổng kết: .........................................

Nơi lưu trữ luận văn: ...............................

***LỜI CẢM ƠN***

***Lời đầu tiên, em xin gửi lời cảm ơn chân thành đến thầy TS. Trần Hoàng Linh, người đã đồng hành cùng em trong suốt quá trình làm luận văn. Cảm ơn thầy đã định hướng, truyền đạt những kiến thức, kinh nghiệm quý báu, cũng như luôn tận tình quan tâm, giúp đỡ em để hoàn thành luận văn.***

***Em cũng xin chân thành cảm ơn quý thầy cô đang giảng dạy ở khoa Điện- Điện tử trường Đại học Bách Khoa ĐHQG tp HCM. Cảm ơn thầy cô đã luôn tận tâm trong việc truyền đạt kiến thức chuyên môn cũng như những chia sẻ kinh nghiệm quý giá để chúng em trang bị đủ tri thức bước vào đời. Cảm ơn nhũng món quà tri thức vô giá giúp em biết mình còn nhỏ bé đến thế nào để em tiếp tục cố gắng.***

***Cảm ơn trường Đại học Bách Khoa đã cho em một môi trường học tập và rèn luyện. Bên cạnh vốn kiến thức chuyên môn mà em đã có được, quan trọng hơn hết, đó là em đã học thêm được nhiều bài học quý giá về cách học tập, kỹ năng sống và rèn luyện tinh thần. Cảm ơn những mùa bài tập lớn, mùa thi, những bài thí nghiệm và cả những chương trình, chiến dịch tình nguyện của trường đã nuôi lớn tâm trí em từng ngày. Cảm ơn những kỉ niệm gắng bó với mái trường này, em sẽ không bao giờ có thể trải qua lần nữa.***

***Cảm ơn những người bạn đã cùng tôi đi qua năm tháng ở Bách Khoa, những người bạn sống cùng, học cùng. Những người bạn đã dang đôi tay giúp đỡ khi tôi cần, chong đèn cùng tôi hoàn thành bài tập. Những người bạn đi cùng tôi hay không còn bên cạnh tôi nữa. Cảm ơn các bạn.***

***Con cảm ơn ba mẹ đã hi sinh để con được đến trường, được học tập. Cảm ơn ba mẹ đã không quản ngại khó khăn nuôi dạy con nên người. Ba mẹ đã luôn yêu thương, dành mọi thứ tốt nhất, tạo mọi điều kiện để con theo đuổi ước mơ, chấp cho con đôi cách bay xa tận chân trời. Những năm xa nhà giúp con thêm yêu và quý trọng tình cảm gia đình, tình yêu thương của ba mẹ.***

***Lời cuối cùng, em kính chúc quý thầy cô trường Đại học Bách Khoa dồi dào sức khỏe, đạt được nhiều thành công trong cuộc sống và sự nghiệp, cũng như luôn giữ ngọn lửa đam mê trong nghiên cứu và giảng dạy.***

*Tp. Hồ Chí Minh, ngày 06 tháng 01 năm 2022 .*

**Sinh viên**

**TÓM TẮT LUẬN VĂN**

Luận văn này trình bày về quá trình thiết kế ra một hệ thống trò chơi có tên gọi là battle tank, trò chơi này là một trò chơi điện tử bắn súng đa hướng.

Người chơi điều khiển xe tăng, phải tiêu diệt xe tank của đối phương bằng mọi cách có thể, đồng thời cố gắng sống sót khỏi xe tăng của địch. Diện tích của trò chơi gói gọn trong một màn hình.

Có hai chế độ chơi là: chơi với người và chơi đơn. Với chế độ chơi với người: xe tăng của cả hai xuất hiện ở hai góc xa nhau và ai tới ba điểm trước sẽ thắng.

Với chế độ chơi đơn. Máy sẽ chuyển động tự do và bắn mọi hướng. Người chơi cố gắng sinh tồn và tiêu diệt máy.

Đồng thời trò chơi có nhạc nền và địa hình kèm theo. Với địa hình có các dạng như tường gạch ( có thể bị phá hủy bằng đạn xe tăng), thép ( không thể bị phá hủy bằng bất cứ gì), sông( đạn có thể đi qua nhưng xe tăng thì không).

Người chơi điều khiển xe tank và các tương tác có thể có thông qua bàn phím hoặc switch trên kit DE1.

**MỤC LỤC**

[1. GIỚI THIỆU. 1](#_Toc92250224)

[1.1 Tổng quan. 1](#_Toc92250225)

[1.1.1 FPGA là gì. 1](#_Toc92250226)

[1.1.2 Lịch sử ra đời fpga . 2](#_Toc92250227)

[1.1.3 Ứng dụng. 3](#_Toc92250228)

[1.1.4 Tổng quan về verilog. 3](#_Toc92250229)

[1.2 Tình hình nghiên cứu trong và ngoài nước. 3](#_Toc92250230)

[1.3 Nhiệm vụ luận văn. 4](#_Toc92250231)

[1.3.1 Yêu cầu 4](#_Toc92250232)

[1.3.2 Kết quả cần đạt được. 4](#_Toc92250233)

[2 LÝ THUYẾT. 4](#_Toc92250234)

[2.1 Lý thuyết phần cứng. 4](#_Toc92250235)

[2.1.1 Board DE1. 4](#_Toc92250236)

[2.1.2 PS/2 Keyboard. 8](#_Toc92250237)

[2.1.3 VGA. 10](#_Toc92250238)

[2.1.4 SD. CARD 14](#_Toc92250239)

[2.1.5 AUDIO CODEC 20](#_Toc92250240)

[2.2 Lý thuyết vẽ hình và di chuyển. 25](#_Toc92250241)

[2.2.1Vẽ xe tăng và viên đạn. 25](#_Toc92250242)

[2.2.2 Vẽ hình nền . 25](#_Toc92250243)

[2.2.3 Vẽ map. 25](#_Toc92250244)

[3 THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG 25](#_Toc92250245)

[4 . THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM. 29](#_Toc92250246)

[4.1 Sơ đồ khối tổng quát: 29](#_Toc92250247)

[4.2. Khối KEYBOARD. 30](#_Toc92250248)

[4.3. Khối SD card 33](#_Toc92250249)

[4.4. Khối PLL và giảm tần số 38](#_Toc92250250)

[4.5. Khối vẽ 38](#_Toc92250251)

[4.6. Khối trung tâm 38](#_Toc92250252)

[4.7 Khối FIFO 39](#_Toc92250253)

[4.8 Khối vẽ xe tăng và viên đạn và map. 40](#_Toc92250254)

[4.8.1 Sơ đồ điều hướng xe tăng. 40](#_Toc92250255)

[4.8.2 Tình trạng xe tăng. 42](#_Toc92250256)

[4.8.3 Sơ đồ điều hướng viên đạn 44](#_Toc92250257)

[4.8.4 Tình trạng viên đạn 46](#_Toc92250258)

[4.8.5 Hiển thị trên map và hình nền thông báo 47](#_Toc92250259)

[4.9 Khối Audio. 47](#_Toc92250260)

[4.9.1 Khối I2C. 47](#_Toc92250261)

[4.9.2 Khối phát nhạc. 48](#_Toc92250262)

[4.10 Khối VGA 49](#_Toc92250263)

[4.11 Khối hiển thị 50](#_Toc92250264)

[5. KẾT QUẢ THỰC HIỆN 52](#_Toc92250265)

[5.1 Kết quả biên dịch 52](#_Toc92250266)

[5.2 Kết quả khi chạy chương trình 53](#_Toc92250267)

[6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 54](#_Toc92250268)

[6.1 Cách thực hiện chương trình 54](#_Toc92250269)

[6.2 Kết luận 55](#_Toc92250270)

[6.3 Hướng phát triển 55](#_Toc92250271)

[7. TÀI LIỆU THAM KHẢO 55](#_Toc92250272)

[8. PHỤ LỤC 56](#_Toc92250273)

[8.1 Chuyển từ file ảnh sang định dạng mif 56](#_Toc92250274)

[8.2 Chuyển dữ liệu file nhạc vào thẻ nhớ SD card 59](#_Toc92250275)

[8.3 Một số bảng tra lệnh và số liệu 61](#_Toc92250276)

[8.3.1 Bảng lệnh SD card 61](#_Toc92250277)

[8.3.2 Bảng cấu hình WM8731 64](#_Toc92250278)

DANH SÁCH HÌNH MINH HỌA

[Hình 2.1 BOARD DE1. 5](#_Toc92250279)

[Hình 2.2 Sơ đồ khối DE1 Board 6](#_Toc92250280)

[Hình 2.3 Cổng PS/2 8](#_Toc92250281)

[Hình 2.4 Khung truyền dữ liệu PS/2 9](#_Toc92250282)

[Hình 2.5 Mã tương ứng của từng phím 10](#_Toc92250283)

[Hình 2.6 Sơ đồ chân VGA. 10](#_Toc92250284)

[Hình 2.7 Nguyên lý hiển thị 12](#_Toc92250285)

[Hình 2.8 VGA Horizontal Timing 13](#_Toc92250286)

[Hình 2.9 Chức năng của sd card 15](#_Toc92250287)

[Hình 2.10: Mô phỏng sơ đồ khối điều khiển của thẻ SD 16](#_Toc92250288)

[Hình 2.11Sơ đồ chân SD card 17](#_Toc92250289)

[Hình 2.12 Cấu hình tín hiệu SPI 17](#_Toc92250290)

[Hình 2.13 Cấu trúc câu lệnh SD card 18](#_Toc92250291)

[Hình 2.14 Tín hiệu phản hồi R1 19](#_Toc92250292)

[Hình 2.15 Sơ đồ khối âm thanh 21](#_Toc92250293)

[Hình 2.16 Pin Audio Codec 21](#_Toc92250294)

[Hình 2.17 I2C protocol 22](#_Toc92250295)

[Hình 2.18 Tín hiệu ACK. 23](#_Toc92250296)

[Hình 2.19 Gói thông tin được gửi đi. 23](#_Toc92250297)

[Hình 2.20 Chế độ Left Justified Mode 24](#_Toc92250298)

[Hình 3.1 Sơ đồ kết nối phần cứng. 27](#_Toc92250299)

[Hình 3.2 Các tín hiệu chính trong phần cứng. 28](#_Toc92250300)

[Hình 4.1 Sơ đồ khối tổng quát. 30](#_Toc92250301)

[Hình 4.2 Các tín hiệu khối Keyboard 31](#_Toc92250302)

[Hình 4.3 Sơ đồ giải thuật khối Keyboard. 33](#_Toc92250303)

[Hình 4.4 Các tín hiệu khối SD card. 33](#_Toc92250304)

[Hình 4.5 Sơ đồ giải thuật khởi động SD card. 36](#_Toc92250305)

[Hình 4.6 Sơ đồ giải thuật khối đọc đơn khối SD card 38](#_Toc92250306)

[Hình 4.7 Tín hiệu chính khối trung tâm. 38](#_Toc92250307)

[Hình 4.8 Khối FIFO. 39](#_Toc92250308)

[Hình 4.9 Điều hướng xe tăng. 40](#_Toc92250309)

[Hình 4.10 Sơ đồ các va chạm có thể có của xe tăng. 43](#_Toc92250310)

[Hình 4.11 Sơ đồ điều hướng viên đạn. 46](#_Toc92250311)

[Hình 4.12 Các va chạm có thể có của viên đạn. 46](#_Toc92250312)

[Hình 4.13 Một số các vật thể trên map. 47](#_Toc92250313)

[Hình 4.14 Các tín hiệu khối phát nhạc. 48](#_Toc92250314)

[Hình 4.15 Sơ đồ điều khiển các tín hiệu điều khiển VGA. 49](#_Toc92250315)

[Hình 4.16 Sơ đồ điều khiển hiển thị trên màn hình. 52](#_Toc92250316)

[Hình 5.1 Kết quả biên dịch. 52](#_Toc92250317)

[Hình 5.2 RTL schematic. 53](#_Toc92250318)

[Hình 5.3 Màn hình trò chơi. 54](#_Toc92250319)

[Hình 8.1 Tạo file wav từ chương trình Audacity. 60](#_Toc92250320)

[Hình 8.2 Tạo file nhạc ghi vào SD card. 60](#_Toc92250321)

**DANH SÁCH BẢNG SỐ LIỆU**

[Bảng 2.1 Mô tả chân VGA 11](#_Toc92250322)

[Bảng 2.2 VGA Horizontal Timing 13](#_Toc92250323)

[Bảng 2.3VGA Vertical timing 14](#_Toc92250324)

[Bảng 2.4 Một số lệnh cơ bản 20](#_Toc92250325)

[Bảng 2.5 Gán chân trên kit DE1 22](#_Toc92250326)

[Bảng 2.6 Các thanh ghi cấu hình. 24](#_Toc92250327)

[Bảng 8.1 Một số lệnh SD card. 61](#_Toc92250328)

[Bảng 8.2 Một số lệnh SD card. 62](#_Toc92250329)

[Bảng 8.3 Một số lệnh SD card. 63](#_Toc92250330)

[Bảng 8.4 Cấu hình Audio 64](#_Toc92250331)

[Bảng 8.5 Cấu hình Audio 65](#_Toc92250332)

[Bảng 8.6 Cấu hình Audio 66](#_Toc92250333)

[Bảng 8.7 Cấu hình Audio 67](#_Toc92250334)

[Bảng 8.8 Cấu hình Audio 68](#_Toc92250335)

# GIỚI THIỆU.

## Tổng quan.

### 1.1.1 FPGA là gì.

Field-programmable gate array (FPGA) 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 chỉ đến khả năng tái lập trình "bên ngoài" của người sử dụng, không phụ thuộc vào dây chuyền sản xuất phức tạp của nhà máy bán dẫn. Vi mạch FPGA được cấu thành từ các bộ phận:

* Các [khối logic](https://vi.wikipedia.org/w/index.php?title=Kh%E1%BB%91i_logic&action=edit&redlink=1) 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](https://vi.wikipedia.org/w/index.php?title=DSP_slice&action=edit&redlink=1), [RAM](https://vi.wikipedia.org/wiki/RAM), [ROM](https://vi.wikipedia.org/wiki/B%E1%BB%99_nh%E1%BB%9B_ch%E1%BB%89_%C4%91%E1%BB%8Dc), nhân vi xử lý...

So sánh FPGA với ASIC và các vi mạch bán dẫn khác:

ASIC (Application-Specific Integrated Circuit) là một vi mạch IC được thiết kế dành cho một ứng dụng cụ thể.

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 đựợc 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 chi phí giảm, 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, 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 .
* Kiến trúc của FPGA cho phép nó có khả năng chứa khối lượng lớn cổng logic. (logic gate), 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, 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 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).

### 1.1.2 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 1 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.

CPLD được cấu trúc từ số lượng nhất định các khối SPLD (Simple programable 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 (multication 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 hộ 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.

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...

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.

### Tổng quan về verilog.

**Verilog**, được tiêu chuẩn hóa thành **IEEE 1364**, là ngôn ngữ mô tả phần cứng (hardware description language, viết tắt: HDL) được sử dụng để mô hình hóa các hệ thống điện tử. Nó được sử dụng phổ biến nhất trong thiết kế và xác minh các mạch kỹ thuật số ở trừu tượng mức chuyển thanh ghi. Nó cũng được sử dụng trong việc xác minh các mạch tương tự và mạch tín hiệu hỗn hợp, cũng như trong thiết kế các mạch di truyền.  Vào năm 2009, tiêu chuẩn Verilog (IEEE 1364-2005) đã được hợp nhất vào tiêu chuẩn system verilog, tạo ra tiêu chuẩn IEEE 1800-2009. Kể từ đó, Verilog chính thức là một phần của ngôn ngữ SystemVerilog. Phiên bản hiện tại là tiêu chuẩn IEEE 1800-2017.

## Tình hình nghiên cứu trong và ngoài nước.

Hiên tại trong nước và trên thế giới có rất nhiều nghiên cứu được thực hiện trên kit DE. Các loại giao tiếp cơ bản như PS/2, VGA, cùng nhiều ứng dụng khác được thưc hiện trên kit DE. Về việc thực hiện mô phỏng ứng dụng paint trên FPGA đa phần sử dụng NIOS II và ngôn ngữ C. Trong luận văn này sẽ thực hiện game battle tank đơn giản hoàn toàn bằng ngôn ngữ Verilog trên kit DE1.

## Nhiệm vụ luận văn.

### Yêu cầu

Thực hiện các giao tiếp trên kit DE1:

* Giao tiếp bàn phím PS/2.
* Giao tiếp VGA.
* Giao tiếp SD Card.
* Giao tiếp module âm thanh WM8731.

### Kết quả cần đạt được.

Thiết kế cần đạt được:

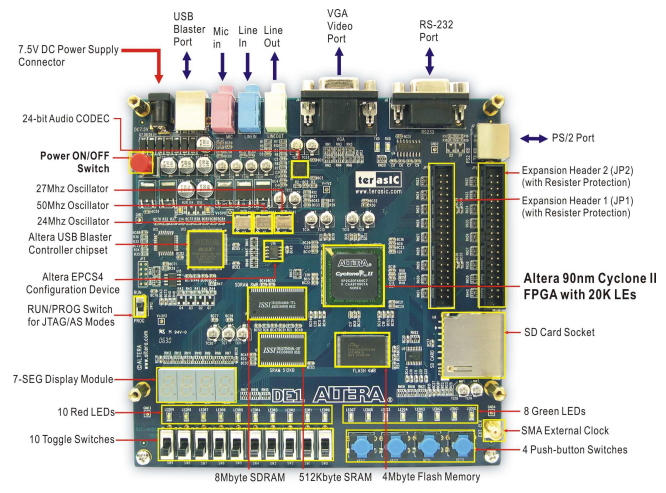
* Xe tăng chạy bình thường.
* Đạn đi bình thường.
* Tuân theo luật chơi có sẵn.
* Nhạc nền ổn.
* Màn hình hiện thị đúng những gì mong muốn.

# LÝ THUYẾT.

## Lý thuyết phần cứng.

### Board DE1.

Board DE1 được mô tả bởi hình sau:



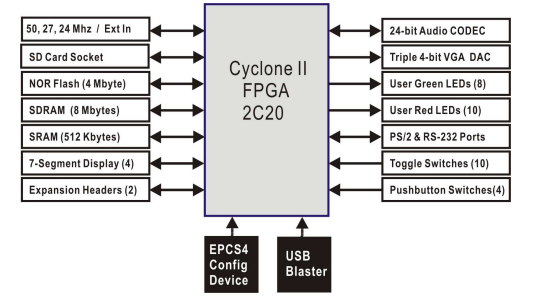
Hình 2.1 BOARD DE1.

DE1 Board có nhiều tính năng cho phép thực hiện được nhiều ứng dụng.

Phần cứng DE1:

1. Altera Cyclone® II 2C20 FPGA device.
2. Altera Serial Configuration device – EPCS4.
3. 512-Kbyte SRAM.
4. 8-Mbyte SDRAM.
5. 4-Mbyte Flash memory.
6. SD Card socket.
7. 4 pushbutton switches.
8. 10 toggle switches.
9. 10 red user LEDs.
10. 8 green user LEDs.
11. 50-MHz oscillator, 27-MHz oscillator and 24-MHz oscillator for clock sources.
12. 24-bit CD-quality audio CODEC with line-in, line-out, and microphone-in jacks.
13. VGA DAC (4-bit resistor network) with VGA-out connector.
14. RS-232 transceiver and 9-pin connector.
15. PS/2 mouse/keyboard connector.
16. Two 40-pin Expansion Headers with resistor protection.
17. Powered by either a 7.5V DC adapter or a USB cable.

Sơ đồ khối DE1 Board:



Hình 2.2 Sơ đồ khối DE1 Board

Chi tiết về các khối trong hình.

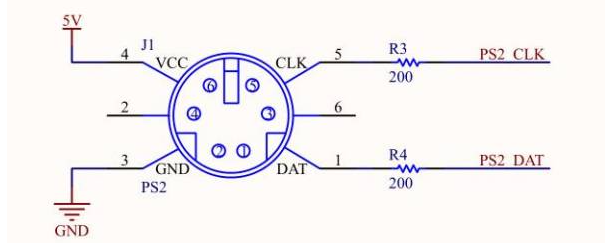
* Cyclone II 2C20 FPGA.
  + 18,752 Les.
  + 52 M4K RAM blocks.
  + 240K total RAM bits.
  + 26 embedded multipliers.
  + 4 PLLs.
  + 315 user I/O pins.
  + FineLine BGA 484-pin package.
* Serial Configuration device and USB Blaster circuit.
  + Altera’s EPCS4 Serial Configuration device.
  + On-board USB Blaster for programming and user API control.
  + JTAG and AS programming modes are supported.
* SRAM.
  + 512-Kbyte Static RAM memory chip.
  + Organized as 256K x 16 bits.
* SDRAM.
  + 8-Mbyte Single Data Rate Synchronous Dynamic RAM memory chip.
  + Organized as 1M x 16 bits x 4 banks.
* Flash memory.
  + 4-Mbyte NOR Flash memory.
  + 8-bit data bus.
* SD card socket.
  + Provides SPI mode for SD Card access.
* Pushbutton switches.
  + 4 pushbutton switches.
  + Debounced by a Schmitt trigger circuit.
  + Normally high; generates one active-low pulse when the switch is pressed.
* Toggle switches.
  + 10 toggle switches for user inputs.
  + A switch causes logic 0 when in the DOWN (closest to the edge of the DE1 board) position and logic 1 when in the UP position.
* Clock inputs.
  + 50-MHz oscillator.
  + 27-MHz oscillator.
  + 24-MHz oscillator.
  + SMA external clock input.
* Audio CODEC.
  + Wolfson WM8731 24-bit sigma-delta audio CODEC.
  + Line-level input, line-level output, and microphone input jacks.
  + Sampling frequency: 8 to 96 KHz.
  + Applications for MP3 players and recorders, PDAs, smart phones, voice recorders, etc.
* VGA output.
  + Uses a 4-bit resistor-network DAC.
  + With 15-pin high-density D-sub connector.
  + Supports up to 640x480 at 60-Hz refresh rate.
* Serial ports.
  + One RS-232 port.
  + One PS/2 port.
  + DB-9 serial connector for the RS-232 port.
  + PS/2 connector for connecting a PS2 mouse or keyboard to the DE1 board.
* Two 40-pin expansion headers.
  + 72 Cyclone II I/O pins, as well as 8 power and ground lines, are brought out to two 40-pin expansion connectors.
  + 40-pin header is designed to accept a standard 40-pin ribbon cable used for IDE hard drives.
  + Resistor protection is provided.

### PS/2 Keyboard.

* + - 1. Chuẩn PS/2.

Chuột PS/2 sử dụng chuẩn giao tiếp PS/2. PS/2 là loại cổng 6 chân dùng để kết nối bàn phim và chuột với hệ thống máy tính. Tên của nó xuất phát từ IBM Personal System/2 là hãng sản xuất máy tính cá nhân.

Sơ đồ chân và mô tả:



Hình 2.3 Cổng PS/2

Sơ đồ chân:

Pin 1 +DATA Data

Pin 2 Not connected

Pin 3 GND Ground

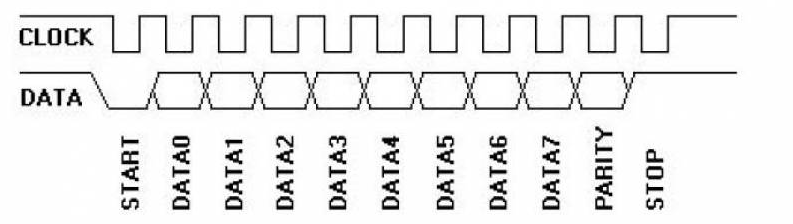
Pin 4 Vcc +5 V DC at 275 mA

Pin 5 +CLK Clock

Pin 6 Not connected

Dữ liệu được truyền nối tiếp bit theo khung truyền 11 bit gồm.

* 1 bit start mức thấp.
* 8 bit dữ liệu.
* 1 stop bit mức cao.
* 1 parity kiểm tra lẻ.

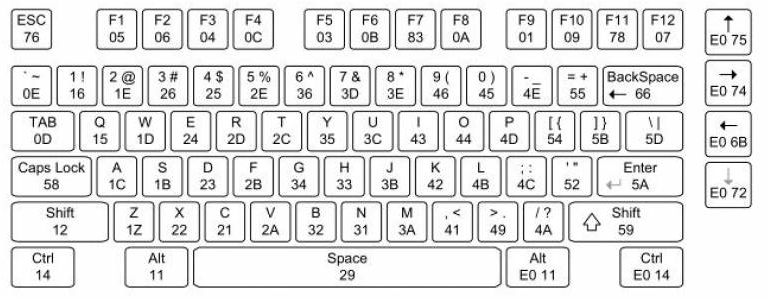


Hình 2.4 Khung truyền dữ liệu PS/2

* + - 1. Giao tiếp với bàn phím

Để có thể thu lại được dữ liệu thay vì phải thiết kế một bộ lấy mẫu ta sẽ dùng một mạch dò sườn xuống của tín hiệu ps2\_clk làm điểm tham khảo. Đầu vào của mạch bao gồm tín hiệu ps2\_data và ps2\_clk nhận từ keyboard, tín hiệu clock của host, một tín hiệu reset của hệ thống. Đầu ra của mạch sẽ là 8 bit dữ liệu và một tín hiệu có mã mới.

Mã của các phím khác nhau trong bàn phím như sau:

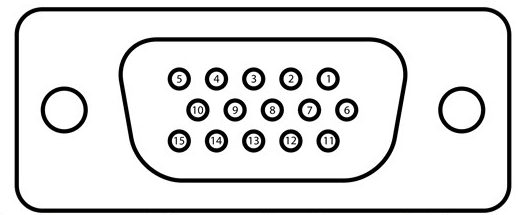


Hình 2.5 Mã tương ứng của từng phím

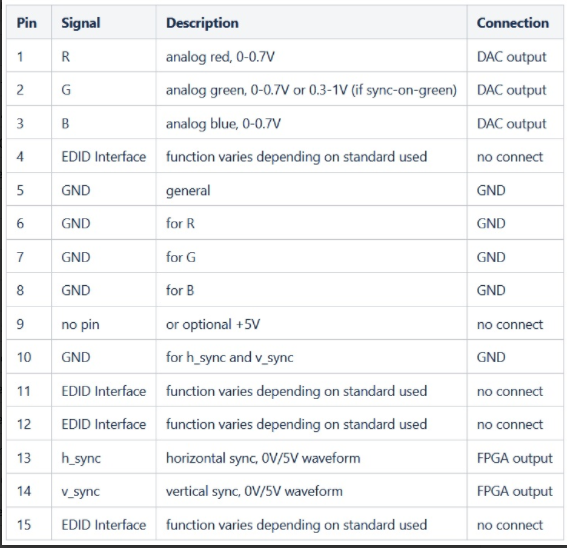
### VGA.

Video Graphics Array (VGA) là một bộ điều khiển màn hình và tiêu chuẩn đồ họa đi cùng với nó, được giới thiệu năm 1987 bởi IBM cùng dòng máy tính cá nhân PS/2.VGA là trở thành tiêu chuẩn chung tối thiểu mà hầu như mọi phần cứng đồ họa PC.

* Sơ đồ chân và mô tả:

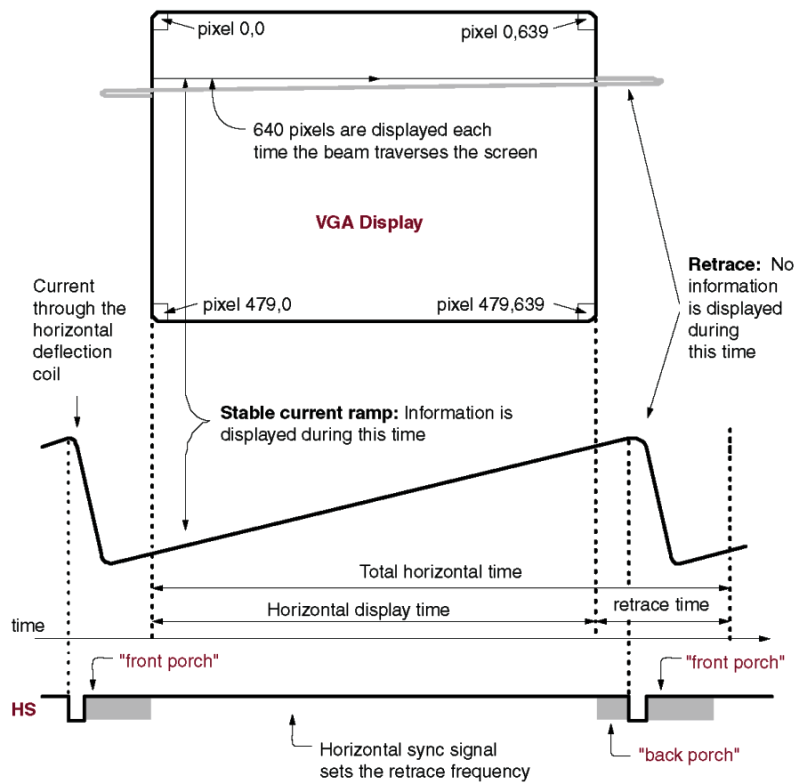


Hình 2.6 Sơ đồ chân VGA.



Bảng 2.1 Mô tả chân VGA

* Hiển thị:



Hình 2.7 Nguyên lý hiển thị

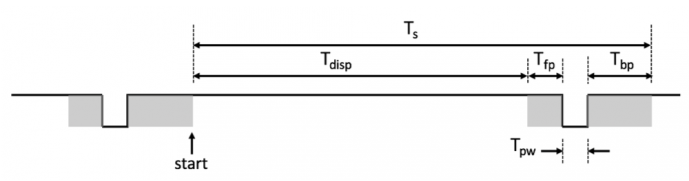
Để hiển thị hình ảnh trên màn hình VGA, bộ điều khiển VGA phải tạo ra tín hiệu thời gian thích hợp cho định dạng hiển thị VGA nhất định. Mỗi định dạng hiển thị có một bộ thông số thời gian khác nhau và mạch điều khiển VGA có thể tạo ra các màn hình với các định dạng khác nhau bằng cách thay đổi tín hiệu thời gian.  Tín hiệu định thời cho định dạng màn hình VGA 640 × 480. Định dạng này hiển thị 640 cột dọc, 480 dòng ngang và tổng số 307.200 pixel (640 cột × 480 dòng). Mỗi pixel trên màn hình có thể được tham chiếu bằng tọa độ x, y của nó trong đó x là cột (bắt đầu từ 0 ở bên trái và tăng dần khi di chuyển sang phải) và y là dòng (bắt đầu từ 0 ở trên cùng và tăng dần di chuyển xuống).

* **Đồng hồ pixel.**

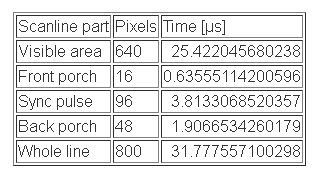
Thông số thời gian quan trọng nhất của bộ điều khiển VGA là tốc độ xung nhịp pixel. Tốc độ đồng hồ pixel xác định tốc độ hiển thị các pixel. Tốc độ xung nhịp pixel cho định dạng 640 × 480 là 25 MHz . . Tất cả các tín hiệu thời gian khác đều bắt nguồn từ đồng hồ pixel này.

* **VGA Horizontal Timing**

Bộ điều khiển VGA phải tạo ra tín hiệu “Đồng bộ hóa theo chiều ngang” (được gọi là HS). Tín hiệu đồng bộ ngang là một tín hiệu định kỳ cho biết màn hình bắt đầu một đường ngang mới, tần suất của đường ngang và thời gian mà màn hình sẽ được để trống. Tín hiệu đồng bộ ngang (giống như tín hiệu đồng bộ dọc) bao gồm bốn pha như thể hiện trong hình bên dưới.

****

Hình 2.8 **VGA Horizontal Timing**

****

Bảng 2.2 **VGA Horizontal Timing**

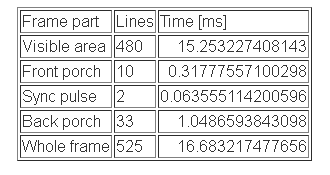
Bốn giai đoạn này như sau:

* **Thời gian hiển thị** (T disp ): Đây là thời gian mà màn hình VGA hiển thị các pixel trên một đường ngang. HS phải cao trong giai đoạn này. (640pixel).
* **Front Porch** (T­­­­­fp ): Đây là thời gian sau khoảng thời gian hiển thị mà ở đó bắt đầu phản hồi. HS cũng phải cao trong giai đoạn này. (16 pixel).
* **Xung đồng bộ** (Tpw ): Đây là xung đồng bộ hóa trong đó tín hiệu HS ở mức thấp.(96 pixel).
* **Back Porch** (Tbp ): Giai đoạn cuối cùng là backporch. Tín hiệu HS phải cao trong giai đoạn này.(48 pixel).

Tín hiệu HS phải liên tục tuần tự qua cả bốn giai đoạn này. Tổng của bốn giai đoạn này là 800 pixel đồng hồ .

* **VGA Vertical timing**

 Tín hiệu đồng bộ dọc cho biết màn hình kết thúc toàn bộ khung hình. Giống như tín hiệu đồng bộ ngang, tín hiệu đồng bộ dọc bao gồm bốn giai đoạn sau: thời gian hiển thị, hiên trước, xung đồng bộ và hiên sau.

****

Bảng 2.3 **VGA Vertical timing**

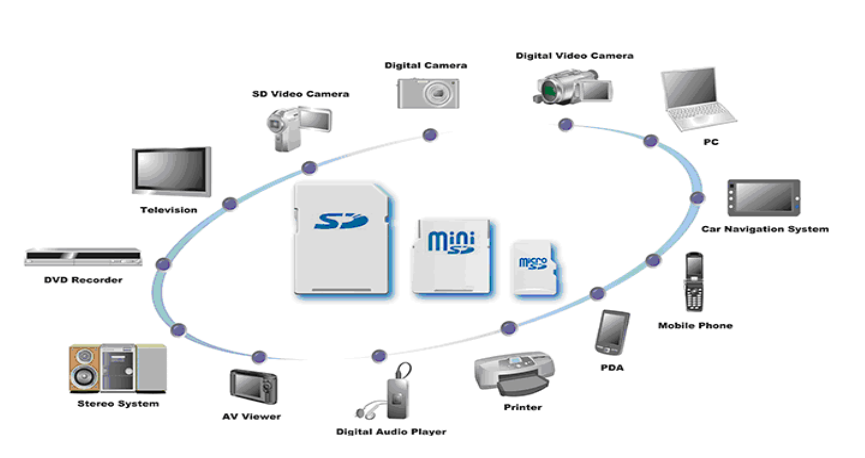
### SD. CARD

**SD Card (Secure Digital Card) là một định dạng thẻ nhớ được sử dụng để lưu trữ dữ liệu chủ yếu dùng trên các thiết bị di động.** Thẻ nhớ hoặc thẻ flash điện tử là một loại bộ nhớ được sử dụng để lưu trữ nội dung kỹ thuật số. Thẻ nhớ được sử dụng trong nhiều thiết bị điện tử như máy ảnh kỹ thuật số, máy mp3, tự điển điện tử, máy tính xách tay, máy điện thoại di động vv…Thể nhớ có kích thước nhỏ gọn dùng để nghi lại và có thể giữ lại dữ liệu khi không có điện.

SD Card, hay còn gọi là thẻ SD, được sản xuất và phát triển dựa trên các tiêu chuẩn kỹ thuật của hiệp hội thẻ SD (SDA). SDA được thành lập ngày 28/1/2000 bởi các tập đoàn Matsushita, Panasonic, SanDisk, Nintendo, Toshiba…).

Định dạng thẻ SD được chia làm 4 loại gồm: Hiệu suất tiêu chuẩn SDSC – Standard Capacity; Hiệu suất cao SDHC – High Capacity; Hiệu suất mở rộng SDXC - eXtended Capacity (SDXC); và loại thẻ kết hợp đầu vào/đầu ra với chức năng lưu trữ dữ liệu SDIO – input/output. Bốn định dạng này được được thiết kế dựa trên Ba hình thức, kích thước là: SD (tiêu chuẩn thông thường), miniSD (hình thức nhỏ hơn SD) và microSD (hình thức rất nhỏ). Tuy nhiên, định dạng thẻ SDXC không có hình thức mini còn định dạng SDIO không có hình thức micro.

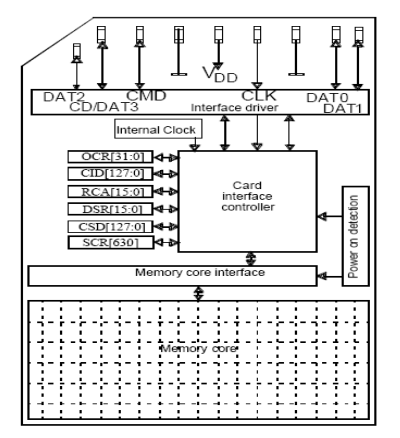
SD là những thẻ nhớ tích hợp cao, bộ nhớ flash với khả năng truy cập ngẫu nhiên và nối tiếp. Nó có thể truy cập thông qua một giao diện chuyên dụng nối tiếp tối ưu hóa cho viêc truyền dữ liệu nhanh và đáng tin cậy. Những thẻ SD là hoàn toàn tương thích với những tiêu chuẩn mới của người dùng, được gọi là hệ thống SD Card tiêu chuẩn xác định trong đặc tả kỹ thuật hệ thống thẻ SD. Hệ thống SD là một loạt hệ thống lưu trữ mới dựa trên sự đổi mới công nghệ bán dẫn. Nó đã được phát triển để cung cấp một bộ lưu trữ rẻ tiền vừa cơ học mạnh mẽ cho các úng dụng đa phương tiện của người tiêu dùng.



Hình 2.9 Chức năng của sd card

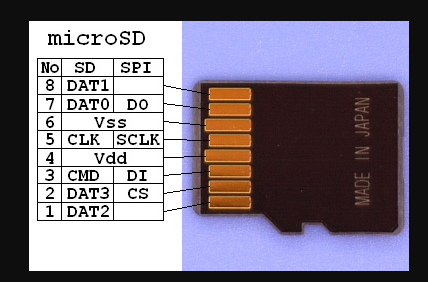
Thẻ SD tiêu chuẩn hoạt động ở phạm vi cung cấp điện từ 2,7-3,3V.

Bộ vi xử lý, bộ nhớ flash điều khiển (xóa, đọc, viết và kiểm soát lỗi) được tích hợp bên trong thẻ nhớ. Dữ liệu được chuyển giữa các thẻ nhớ và bộ điều khiển máy chủ lưu trữ tại đơn vị của mỗi khối 512 byte trong mặc định. Cấu trúc của thẻ SD bao gồm giao diện vi xử lý và giao diện bộ nhớ.



Hình 2.10: Mô phỏng sơ đồ khối điều khiển của thẻ SD

Thẻ SD được điều khiển bởi 6 dòng tín hiệu nằm trên giao diện thẻ: CMD, CLK, DAT0, DAT1, DAT2, DAT3 thông qua giao diện xử lý và điều khiển để truy xuất nội dung trong ô nhớ.

****

Hình 2.11Sơ đồ chân SD card

SD Card có 2 chuẩn giao tiếp: SD Protocol và SPI Protocol.

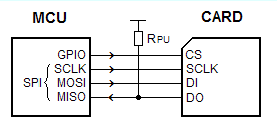
* SD Protocol: Sử dụng chân CMD để truyền lệnh và nhận phản hồi, chân DAT0 nhận dữ liệu ở chế độ 1 bit hay DAT0 – DAT3 nhận dữ liệu ở chế độ 4 bit.
* SPI Protocol: Sử dụng chân CMD để truyền lệnh, chân DAT0 nhận phản hồi và dữ liệu, DAT3 đóng vai trò như chân chip select (CS).

Thẻ SD hoạt động ở hai chế độ tốc độ. Tốc độ đồng hồ chế độ mặc định là 0-25MHz.

Trong luận văn này sử dụng chuẩn SPI để giao tiếp với SD Card. Chế độ đọc đơn khối.

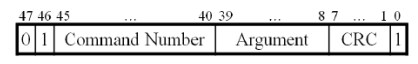
**Chế độ SPI:**

* Cấu hình tín hiệu SPI:

****

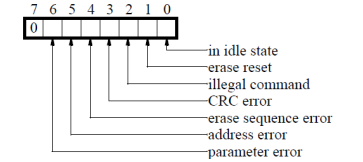
Hình 2.12 Cấu hình tín hiệu SPI

* **SPI Serial Peripheral Interface** là một chuẩn truyền thông nối tiếp đồng bộ dùng để chuyền dữ liệu.
* Giao thức SPI thường được sử dụng On Board hoặc các đường tín hiệu ngắn. Tốc độ của SPI khá cao thường phụ thuộc vào tốc độ xung truyền vào bộ SPI.
* Giao thức SPI là dao thức dạng Master –Slave trong đó Master giữ quyền điều khiển xung Clock và chọn Slave nào giao tiếp với mình.
* Bus SPI bao gồm 4 đường tín hiệu:
  + MOSI (Master Out Slave In): Đường truyền tín hiệu từ Master đến Slave
  + MISO (Master In Slave Out): Đường truyền tín hiệu từ Slave đến Master
  + CLK (Serial Clock): Đường phát xung đồng hồ được điều khiển bởi Master
  + CS (Chip Select): Đường chọn chip giao tiếp với Master, được kéo xuống 0 khi được chọn
* Nguyên lý hoạt động như sau:
  + Khi muốn truyền nhận dữ liệu tới các Slave, đầu tiên Master kéo đường CS kết nối từ Master tới Slave đó xuống 0.
  + Gửi xung Clock, tương ứng với mỗi Clock sẽ gửi DATA trên chân MOSI tại thời điểm Clock ở mức cao (hoặc thấp tùy người lập trình)
  + Slave cũng có thể gửi ngược lại DATA tại chân MISO tới Master
  + Sự truyền nhận dữ liệu là liên tục nên SPI thường có tốc độ rất cao.
* Một lệnh hợp lệ bao gồm 48 bits ( 6 bytes) .
  + Hai bit bắt đầu (“01”).
  + Theo sau là 6 bits lệnh.
  + Tiếp là 32 bits đối số.
  + Cuối cùng 7 bits CRC.
  + Kết thúc là bit stop (1).



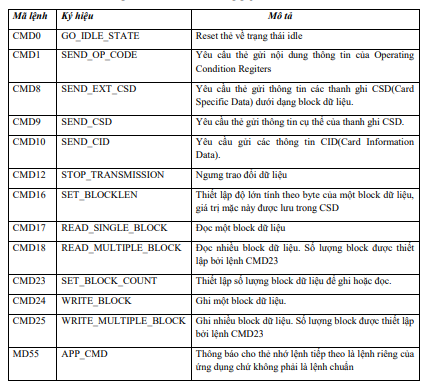
Hình 2.13 Cấu trúc câu lệnh SD card

* Quá trình gửi bắt đầu bằng cách đặt bit trên MOSI và chuyển đổi tín hiệu SD\_CLK từ 0 sang 1 rồi từ 1 về 0. Sau đó lần lượt từ bit từ hai cho đến khi hoàn thành 1 lệnh hoàn chỉnh.
* Khi nhận được lệnh sd sẽ xử lý. Để phản hồi một lệnh thẻ SD yêu cầu tín hiệu CLK chuyển đổi trong ít nhất 8 chu kỳ và MOSI phải ở mức cao để chờ phản hồi. Thời lượng tùy vào từng lệnh Hầu hết lệnh phản hồi dạng 8-bit
* Để giao tiếp với thẻ SD, phải đặt vào chế độ SPI. Đặt MOSI và CS ở mức cao trong ít 74 chu kỳ. Sau đó gửi CMD0: 0x400000000095.
* Đây là lệnh reset, thẻ sẽ phản hồi bằng 8 bits trên MISO. Nếu lệnh thành công ta sẽ nhận được (0000001)­­2.



Hình 2.14 Tín hiệu phản hồi R1

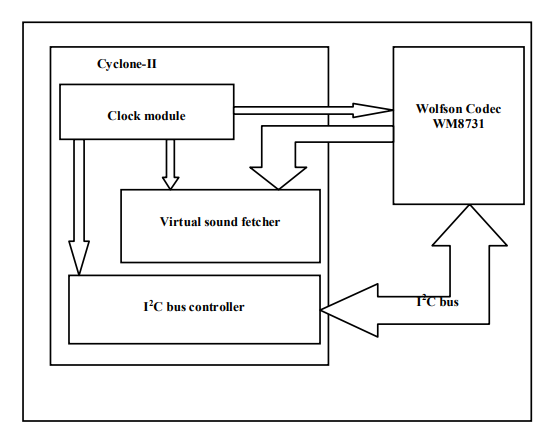
* Để nhận được tín hiệu liên tục chuyển đổi SD\_CLK. Dòng MISO dữ liệu đồng thời MOSI ở mức cao và CS mức thấp.
* Ở trạng thái nhàn rỗi (in idle state): Thẻ ở trạng thái không hoạt động và đang chạy quá trình khởi tạo.
* Đặt lại xóa (erase reset): Một trình tự xóa đã bị xóa trước khi thực thi vì không có trình tự xóa lệnh đã được nhận.
* Lệnh không hợp lệ (illegal command): Mã lệnh không hợp lệ đã được phát hiện.
* Lỗi CRC giao tiếp (com CRC error): Kiểm tra CRC của lệnh cuối cùng không thành công.
* Lỗi trình tự xóa (erase\_seq\_error): Xảy ra lỗi trong chuỗi lệnh xóa.
* Lỗi địa chỉ (address error): Địa chỉ bị lệch, không khớp với độ dài khối đã được sử dụng trong yêu cầu.
* Lỗi tham số (parameter error): Đối số của lệnh (ví dụ: địa chỉ, độ dài khối) nằm ngoài phạm vi cho phép cho thẻ này.
* Dạng tín hiệu phản hồi R2, R3 xem phần tài liệu tham khảo.
* Đọc đơn khối dữ liệu từ thẻ nhớ:
  + Gói dữ liệu ( data token): Các lệnh đọc có truyền dữ liệu đi kèm với chúng. Dữ liệu được truyền qua data token. Tất cả các byte dữ liệu được truyền MSB.
  + Data token dài từ 4 đến 515 byte và có định dạng sau:
    - Byte đầu tiên (0xFE): Bắt đầu block.
    - Bytes 2-513: Gói dữ liệu.
    - Hai bytes cuối CRC (Xác minh CRC không bắt buộc trong SPI).
* Một số lệnh SD Card:



Bảng 2.4 Một số lệnh cơ bản

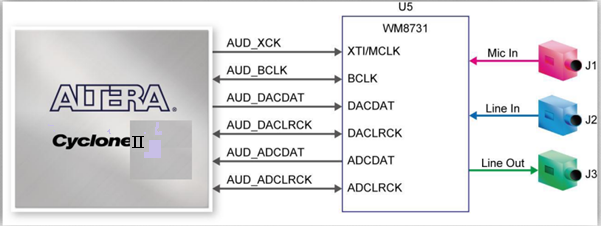
### AUDIO CODEC

* Tín hiệu âm thanh có thể coi như là một điện áp.
* Âm thanh phát ra trên đầu Line out của DE1 BOARD.
* Sơ đồ khối cơ bản:

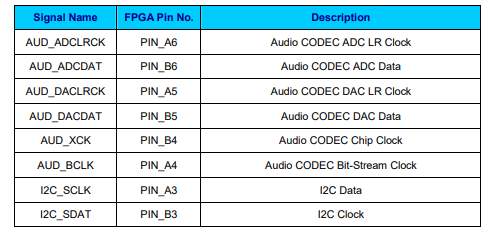


Hình 2.15 Sơ đồ khối âm thanh

* Chi tiết Pin trên DE1 Board:

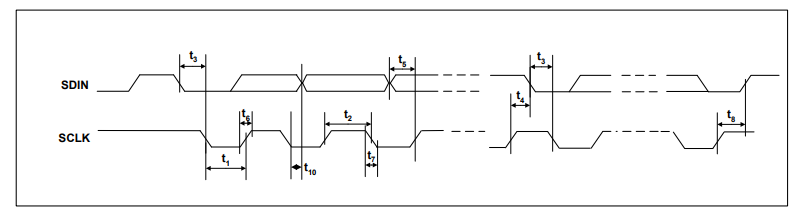


Hình 2.16 Pin Audio Codec



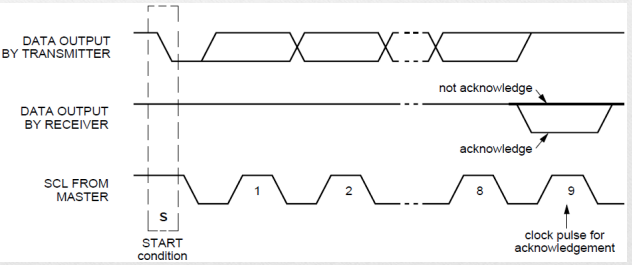
Bảng 2.5 Gán chân trên kit DE1

* Ta giao tiếp với audio thông qua I2C với hai chân I2C\_SCLK và I2C\_SDAT.
* Sau khi khởi tạo thành công ta có thể truyền âm thanh qua giao diện âm thanh kỹ thuật số .
* Ta dùng chế độ two-wire, slave mode với các chân AUD\_XCK,AUD\_BCLK, I2C\_SCLK được cấp từ bên ngoài.
* I2C protocol:



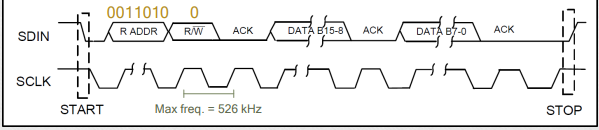
Hình 2.17 I2C protocol

* + Quá trình truyền dữ liệu bằng bit START(S) khi SDIN kéo xuống thấp trong khi SCLK ở mức cao.
  + Sau đó SDIN đặt bit được truyền trong khi SCLK thấp .
  + Dữ liệu lấy mẫu khi SCLK tăng.
  + Khi quá trình thành công bit STOP được gửi bằng cách cho SDIN kéo lên khi SCLK ở mức cao.
  + SDIN được thay đổi trên cạnh rơi và được bắt trên cạnh lên của SCLK.
  + Sau mỗi 8 bit dữ liệu một bit “ACK” được truyền.



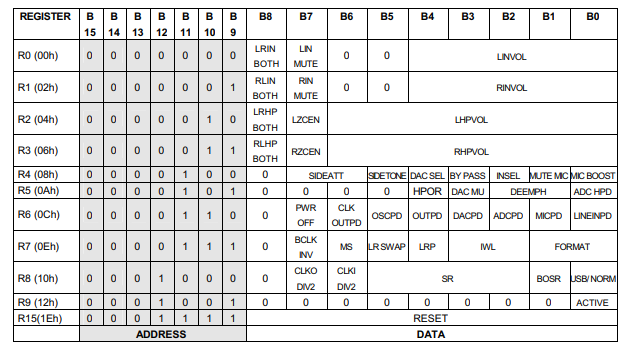
Hình 2.18 Tín hiệu ACK.

* 2-Wire Interface
  + Thiết bị hoạt động như một thiết bị phụ.
  + WM8731 có một trong hai địa chỉ phụ được chọn bằng cách thiết lập trạng thái của PIN CSB.
  + ADDR[6:0] là địa chỉ phụ (0011010)2.
  + R/W is ‘0’.
  + B[15:9] là địa chỉ thanh ghi.
  + B[8:0] là dữ liệu thanh ghi.



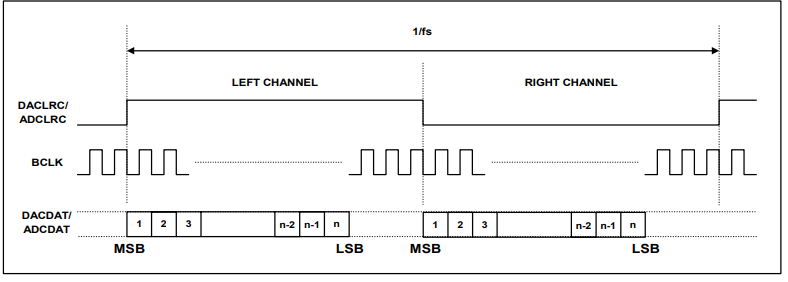
Hình 2.19 Gói thông tin được gửi đi.

* Một số lệnh thiết lập



Bảng 2.6 Các thanh ghi cấu hình.

* Tương quan giá trị của AUD\_BCLK,AUD\_DALRCK và AUD\_DACDAT:
  + Chế độ sử dụng trong đề tài Left Justified Mode



Hình 2.20 Chế độ Left Justified Mode

* + DACLRC tích cạnh lên mỗi 1/fs .
  + BCLK tích cạnh xuống số lần bằng số bit mỗi lần lấy mẫu âm thanh cho mỗi kênh (Ví dụ: 16 bits mỗi mẫu thì BCLK = 1/fs\*32).
  + Bits của mẫu tín hiệu ở mỗi lần BCLK tích cạnh lên.

## Lý thuyết vẽ hình và di chuyển.

### 2.2.1Vẽ xe tăng và viên đạn.

* Xe tăng và viên đạn là những vật thể chuyển động trên màn hình theo thời gian.
* Muốn hiển thị chính xác xe tăng và viên đạn cần có file ROM để lưu trước chính xác vị trí từng điểm ảnh của xe tăng và viên đạn để hiển thị.
* Ở đây ta chọn xe tăng và viên đạn là các khối có hình vuông và hình chữ nhận để dễ lập trình.
* Khi màn hình quét đến địa chỉ của xe tăng hoặc viên đạn cần vẽ thì địa chỉ trong file ROM cũng tăng theo để có giá trị chính xác từng điểm ảnh hiển thị lên màn hình.
* Địa chỉ giá trị lưu trong file ROM chỉ tăng lên khi địa chỉ quét tiến vào vùng giới hạn xe tăng hoặc viên đạn cho trước để có thể hiển thị đúng.
* Khi chuyển động ta phải cập nhật vị trí của xe tăng và viên đạn liên tục để có thể vẽ chính xác.

### 2.2.2 Vẽ hình nền .

* Tương tự như vẽ xe tăng và viên đạn ta cũng cần một file ROM để lưu giá trị từng bit màu để hiển thị.

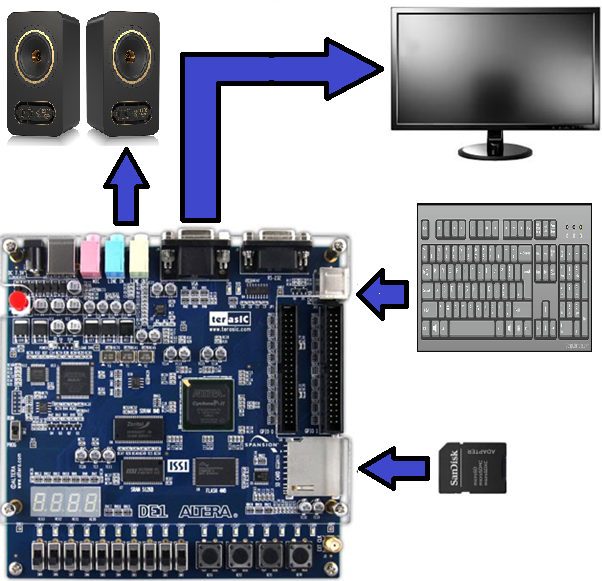
### 2.2.3 Vẽ map.

* Tương tự vẽ hình nền ta cũng cần một file ROM để lưu giá trị của các vật thể trên map.
* Vì bộ nhớ DE1 có hạn nên ta sẽ không lưu cả map để tiết kiệm bộ nhớ.
* Ta dùng file ROM để lưu hình ảnh các vật thể trên map để khi màn hình quét đến thì sẽ hiển thị chính xác vật thể đó.
* Ngoài ra ta tạo một file RAM để lưu các vị trí của các vật thể đó. Nên sẽ tiết kiệm dữ liệu đồng thời vẫn đảm bảo cơ chế game được tiến hành đúng như mong đợi.

# THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

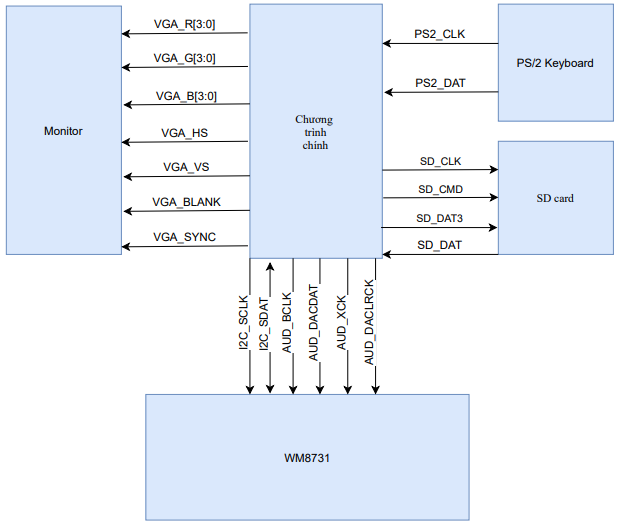
* Yêu cầu:
  + Hiển thị được kết quả trên màn hình có độ phân giải 640x480.
  + Giao tiếp được với DE1 BOARD và bàn phím để tương tác.
  + Bộ nhớ đủ để thực hiện các thao tác.
  + Giáo tiếp với các ngoại vi còn lại.
* Lý do chọn kit DE1:
  + Có giao tiếp PS/2.
  + Có cổng kết VGA để hiển thị với màn hình 640x480.
  + Có giao tiếp với module âm thanh.
  + Lập trình được với ngôn ngữ Verilog.
  + Có cổng giao tiếp với SD card.
* Ưu và nhược điểm:
  + Ưu điểm.
    - Có đầy đủ yêu cầu đáp ứng.
    - Dễ dàng sử dụng ngôn ngữ verilog.
    - Nguồn tài liệu hướng dẫn sử dụng dễ kiếm.
    - Được làm quen cơ bản ở trường học.
  + Nhược điểm.
    - Giá thành cao.

Sơ đồ phần cứng tổng quát:



Hình 3.1 Sơ đồ kết nối phần cứng.

Sơ đồ nối chân chi tiết:



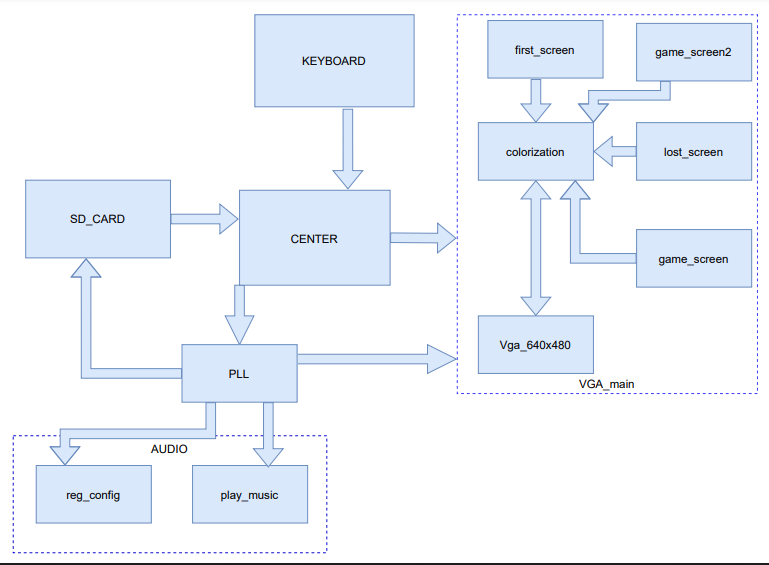
Hình 3.2 Các tín hiệu chính trong phần cứng.

Giải thích các tín hiệu:

* VGA\_HS(horizonal sync): tín hiệu quét ngang.
* VGA\_VS(vertical sync): tín hiệu quét dọc.
* VGA\_BLANK: không sử dụng.
* VGA\_SYNC: tín hiệu đồng bộ hiển thị.
* VGA\_R, VGA\_G, VGA\_B: các tín hiệu màu, 4 bit mỗi màu đỏ, xanh lá, xanh dương.
* PS2\_CLK: xung clock đồng bộ tín hiệu giữa kit và bàn phím.
* PS2\_DAT: tín hiệu nhận được từ bàn phím.
* SD Card bao gồm các tín hiệu:
* SD\_CLK: xung clock đồng bộ tín hiệu giữa thẻ nhớ và kit (ảnh hưởng trực tiếp đến tốc độ truyền dữ liệu).
* SD\_CMD: tín hiệu gửi lệnh từ kít đến thẻ nhớ.
* SD\_DAT: dữ liệu phản hồi từ thẻ nhớ đến kit.
* SD\_DAT3: trong trường hợp này đóng vai trò như tín hiệu cho phép truy cập(chip enable).
* I2C\_SDAT: đường truyền để gửi và nhận dữ liệu.
* I2C\_SCLK: đường mang tín hiệu xung nhịp.
* AUD\_DACDAT: dữ liệu DAC.
* AUD\_BCLK: xung nhịp điều khiển từng bit dữ liệu DAC..
* AUD\_DACLRCK: xung nhịp điều khiển tần số lấy mẫu..
* AUD\_XCK: xung nhịp điều khiển chip.

# . THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM.

## Sơ đồ khối tổng quát:

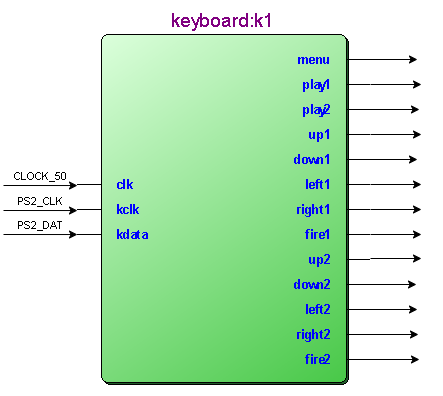


Hình 4.1 Sơ đồ khối tổng quát.

* Chương trình gồm các khối chính.
  + Khối KEYBOARD: tín hiệu từ bàn phím.
  + Khối AUDIO: thực hiện việc phát nhạc.
  + Khối VGA\_main: thực hiện việc hiển thị.
  + Khối SD\_CARD: thực hiện việc đọc dữ liệu từ thẻ SD.
  + Khối CENTER: điều khiển chương trình chạy.
  + Khối gamescreen,gamescreen2: thực hiện việc điều khiển trò chơi dựa trên các tín hiệu nhận vào từ bàn phím.
  + Khối colorization: thực hiện việc tô màu.
  + Khối vga\_640x480: thực hiện việc điều khiển việc quét tín hiệu trên màn hình.
  + Khối lostscreen, firstscreen: hiển thị màn hình thông báo kết quả.

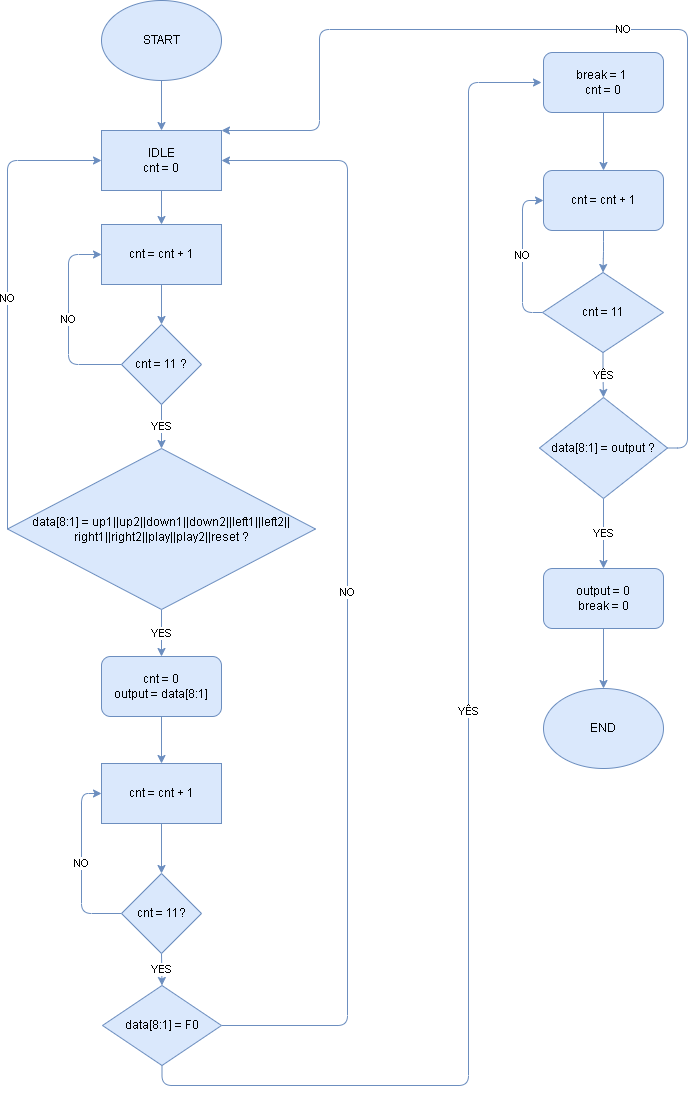
## 4.2. Khối KEYBOARD.

* Khối Keyboard:



Hình 4.2 Các tín hiệu khối Keyboard

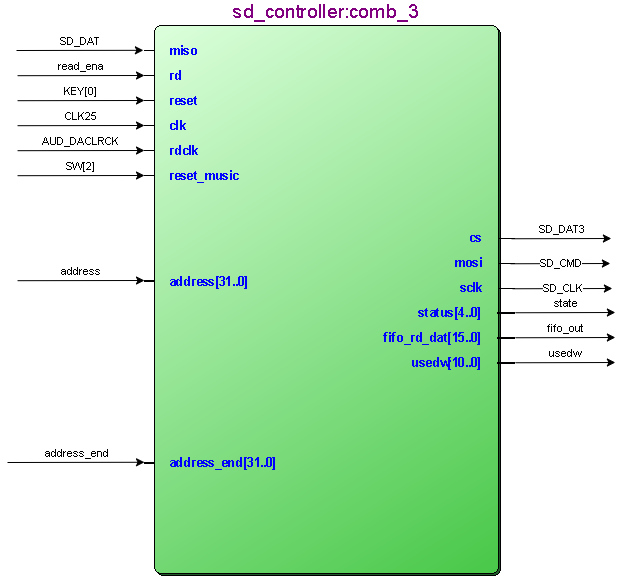
* Khối Keyboard: thực hiện giao tiếp với bàn phím.
  + Giải thích các chân tín hiệu.
    - Clk: xung nhịp hệ thống 50MHz.
    - Kclk: xung clock từ bàn phím .
    - Kdata: data từ bàn phím.
    - Menu,play1,play2,up1,up2,down1,down2,left1,left2,right1,right2: các tín hiệu ngõ ra từ bàn phím.
* Sơ đồ giải thuật:



Hình 4.3 Sơ đồ giải thuật khối Keyboard.

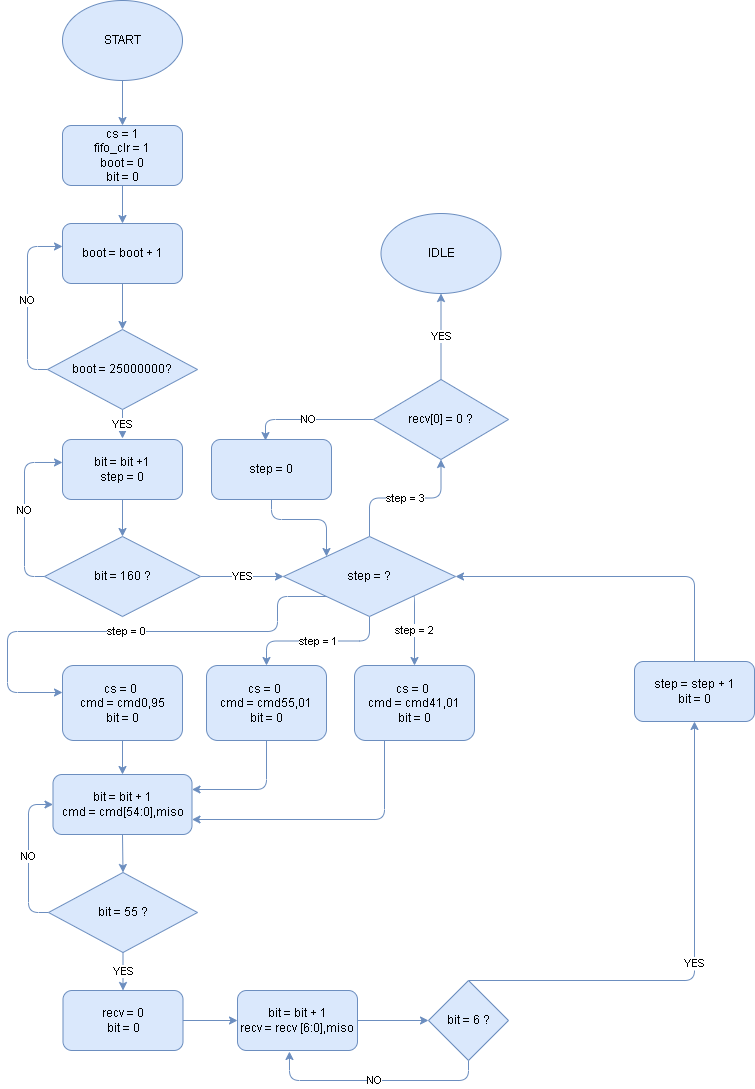
## 4.3. Khối SD card

* + Sơ đồ khối sd card tổng quát như sau:



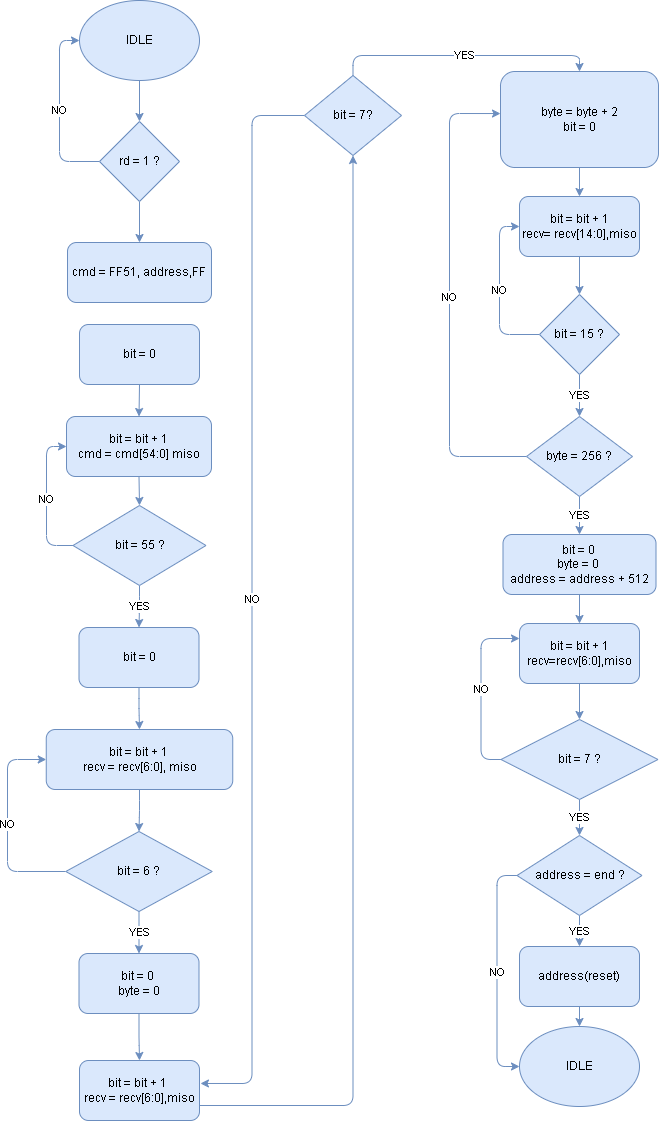
Hình 4.4 Các tín hiệu khối SD card.

* + Giải thích các chân tín hiệu:
    - SD\_DAT: dữ liệu nhận từ thẻ nhớ.
    - SD\_CMD:gửi lệnh cho thẻ nhớ.
    - SD\_DAT3: chân cho phép của thẻ nhớ.
    - SD\_CLK: phát xung clk 25Mhz cho thẻ nhớ.
    - Address: địa chỉ bắt đầu bài hát.
    - Address\_end : địa chỉ kết thúc bài hát.
    - Usedw: cho biết số thanh ghi có trong fifo.
    - Fifo\_out: chân bộ fifo.
    - SW[2]: bắt đầu lại bài hát.
    - KEY[0]: reset hệ thống.
    - Read\_ena: cho phép đọc từ thẻ nhớ (khi fifo không đầy).
    - Clk25: cung cấp xung 25Mhz.
  + Sơ đồ giải thuật:
    - Khởi động SD\_card.



Hình 4.5 Sơ đồ giải thuật khởi động SD card.

* + - Đọc đơn khối từ SD card.



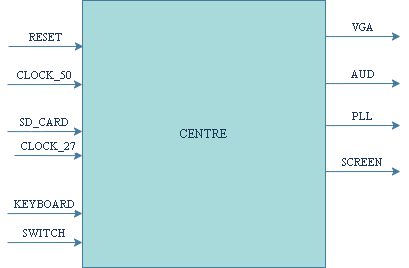
Hình 4.6 Sơ đồ giải thuật khối đọc đơn khối SD card

## 4.4. Khối PLL và giảm tần số

* + Khối PLL: chức năng tạo xung clock: 18,432 Mhz cho giao tiếp với audio module.
  + Giảm tần số: chức năng tạo xung clock: 25Mhz cho giao tiếp với VGA và SD card.
  + Khối PLL được tích hợp sẵn trên DE1.

## 4.5. Khối trung tâm

- Khối trung tâm được hiển thị như sau



Hình 4.7 Tín hiệu chính khối trung tâm.

- Giải thích các yếu tố khối tác động:

+ Reset: Để khởi động lại hệ thống.

+ CLOCK\_50, CLOCK\_27: xung clock của hệ thống.

+ SD\_card: Nhận các tín hiệu từ SD\_card.

+ Keyboard: Nhận các tín hiệu từ bàn phím.

+ SWITCH: Nhận các tín hiệu từ switch.

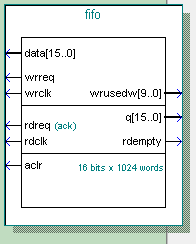
+ SCREEN: chọn ảnh nào để hiện thị lên màn hình

+ AUD: chọn bài hát để phát.

+ VGA: điều khiển màn hình hiển thị chính xác những gì cần.

+ PLL: cung cấp xung clock phù hợp cho các thiết bị hoạt động.

## 4.6. Khối FIFO

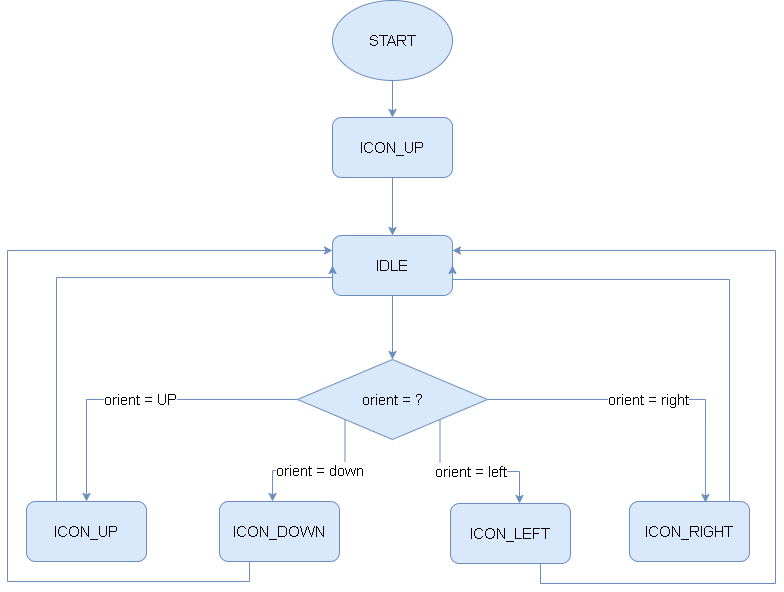


Hình 4.8 Khối FIFO.

* + Giải thích các chân cụ thể:
    - Data: dữ liệu đưa vào fifo.
    - Wrreq: chân cho phép ghi vào fifo luôn cho phép.
    - Wrclk: xung clock cho lúc ghi vào fifo.
    - Wrusedw: cho biết số thanh ghi có trong fifo.
    - Rdreq: chân cho phép đọc ra từ fifo.
    - Rdclk: xung clock cho lúc đọc ra từ fifo.
    - Aclr: ngõ reset .
    - Rdempty: cho biết fifo có rỗng hay không.
    - Q: dữ liệu ngõ ra fifo.
  + Ta thiết lập:
    - FiFO xung ghi chạy ở clock hệ thống.
    - Khi thông báo có byte\_available = 1 mới ghi vào fifo một lần, wrreq chỉ như một xung vì ta chỉ muốn ghi một lần vào fifo một thanh ghi.
    - Đảm bảo fifo không bao giờ đầy bằng cách thiết lập read\_ena của SD CARD mức thấp khi số lượng thanh ghi trong fifo vượt quá một mức nhất định.
    - Đọc từ fifo với tần số bằng với tần số lấy mẫu của âm thanh được chọn để tín hiệu được liền mạch .

## Khối vẽ xe tăng và viên đạn và map.

### Sơ đồ điều hướng xe tăng.



Hình 4.9 Điều hướng xe tăng.

* Mẫu xe tăng:

+ Bên đỏ

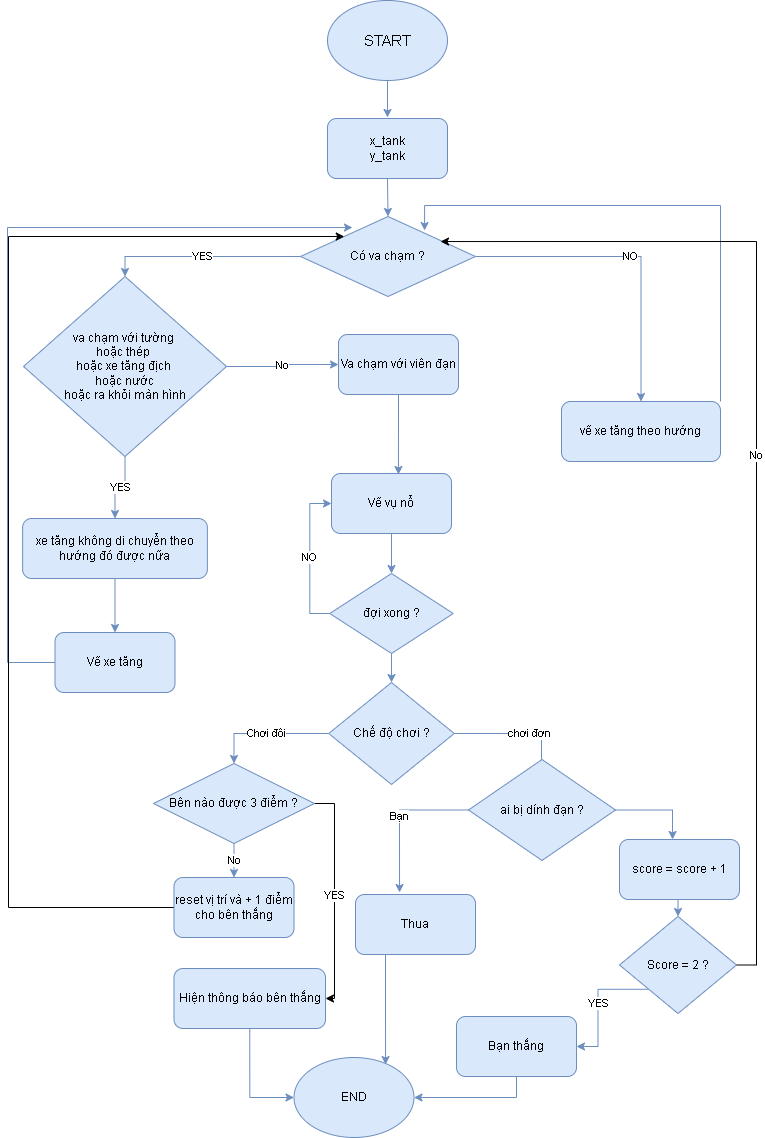
 

+ Bên xanh

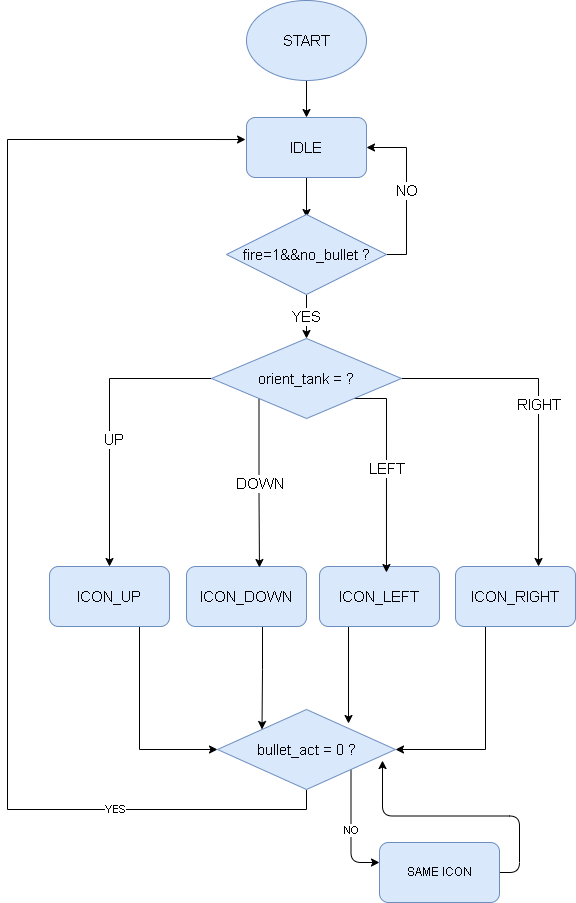
* Cách vẽ theo hướng:
  + Khi tọa độ quét vào tọa độ xe tăng thì địa chỉ trong ROM tăng dần đọc ra bit màu cần hiện. Cứ thế đến khi vẽ hết xe tăng rồi lặp lại.
  + Đối với các hướng lên và trái thì địa chỉ bắt đầu từ 0.
  + Riêng với hướng xuống và phải để tiết kiệm bộ nhớ ta thiết kế xe tăng đối xứng từ đó các hướng phải và xuống chỉ cần địa ngược địa chỉ có trong ROM.
* Nhận biết vật cản để cản trở di chuyển
  + Ta dựa vào file RAM để biết được tọa độ vật cản
  + Căn cứ vào vị trí tương đối của xe tăng để đọc bốn hướng của xe tăng từ RAM xem có vật cản hay không nếu có thì ngăn di chuyển.
  + Nếu có bắn phá thì địa chỉ trong RAM sẽ được cập nhật tình trạng từ đó xe tăng có thể đi qua các vật cản đã bị phá hủy.
* Điều khiển tốc độ xe tăng:
  + Để kiểm soát tốc độ ta cần dùng một bộ đếm.
  + Để khi bộ đếm hoàn thành một chu kỳ thì mới được phép tăng 1 pixel khi di chuyển.
  + Viên đạn cũng tương tự như vậy.

### 4.7.2 Tình trạng xe tăng.



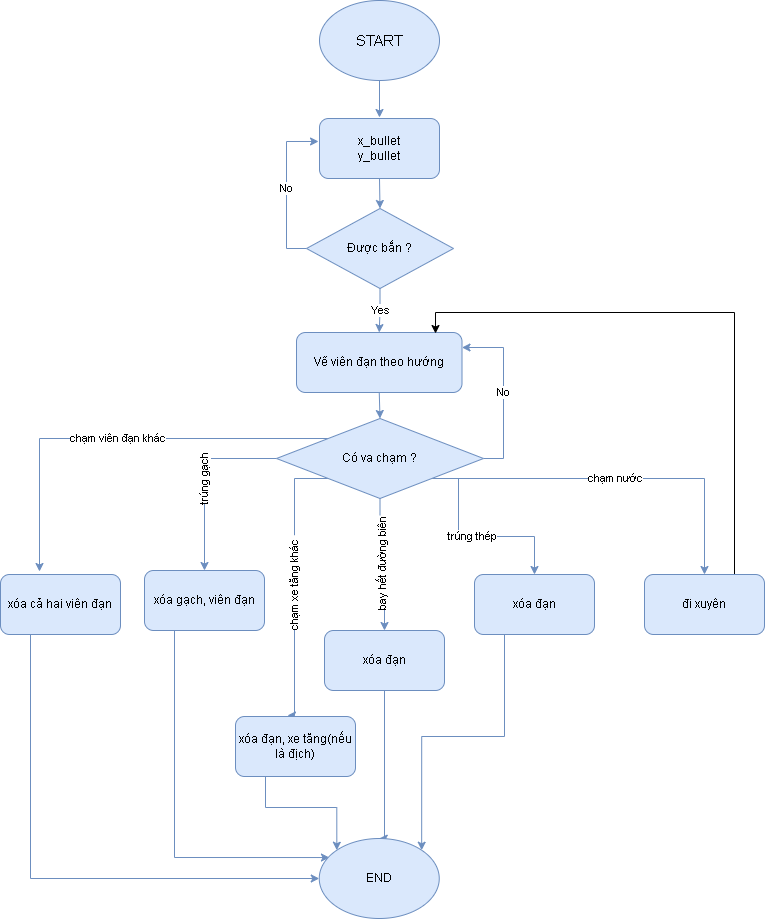
Hình 4.10 Sơ đồ các va chạm có thể có của xe tăng.

### 4.7.3 Sơ đồ điều hướng viên đạn



Hình 4.11 Sơ đồ điều hướng viên đạn.

### 4.7.4 Tình trạng viên đạn



Hình 4.12 Các va chạm có thể có của viên đạn.

### 4.7.5 Hiển thị trên map và hình nền thông báo

- Để tiết kiệm bộ nhớ khi hiện các hình nền thông báo ta chỉ cần lưu hình có độ phân giải 160x120 rồi dùng một thanh ghi bỏ 2 bit thấp của các tín hiệu quét như vậy ta sẽ có hình mới có độ phân giải 640x480 đủ để hiển thị cả màn hình.

- Còn đổi với map như đã trình bày phần lý thuyết thì ta chỉ cần dùng bộ nhớ ROM để lưu trữ các hình hiển thị trên map và bộ nhớ RAM để hiển thị các vị trí muốn hiển thị để tiết kiệm bộ nhớ.

- Riêng việc hiển thị hình ảnh động như nước ta dùng hai bộ nhớ ROM lưu hai hình thái của nước từ đó cứ sau mỗi khoảng thời gian bằng nhau ta hiện lần lượt một trong hai hình ảnh đó từ đó sẽ tạo ra cảm giác nước chuyển động

+ Nước 1 : 

+ Nước 2: 

+ Viên gạch : 

+ Thép: 

Hình 4.13 Một số các vật thể trên map.

* Một số gợi ý thêm như : cỏ, băng ..v..v
* Để hiển thị được các hình ảnh ta dùng Matlab để chuyển từ file ảnh sang file mif. Đoạn code tham khảo ở phần phụ lục.

## 4.8. Khối Audio.

### 4.8.1 Khối I2C.

- Khối I2C giống với phần lý thuyết đã trình bày phía trên.

- Ta chọn các thiết lập sau cho hệ thống.

+ Left headphone out.

+ Right headphone out.

+ Analogue Audio Path Control.

+ Digital Audio Path Control.

+ Power Down Control.

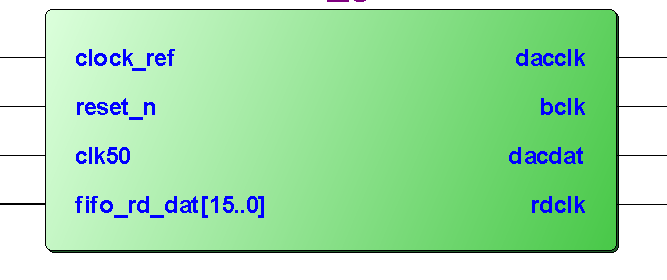
+ Digital Audio Interface Format.

+ Sampling Control.

+ Active Control.

### 4.8.2 Khối phát nhạc.

- Khối điều khiển phát nhạc như sau:



Hình 4.14 Các tín hiệu khối phát nhạc.

* Giải thích các chân của khối :

+ Clock\_ref: có tần số 18.432Mhz từ bộ PLL đầu ra chân AUD\_XCK.

+ Reset\_n: chân reset khối.

+ Clk50: xung clock hệ thống 50Mhz.

+ Fifo\_rd\_dat[15:0]: tín hiệu âm thanh được đọc ra từ fifo.

+ Dacdat: đầu ra cho chân AUD\_DACDAT.

+ Bclk: đầu ra cho chân AUD\_BCLK.

+ Rdclk: xung clock cung cấp cho chân điều khiển đọc của FIFO.

+ Dacclk: đầu ra cho chân AUD\_DACLRCK.

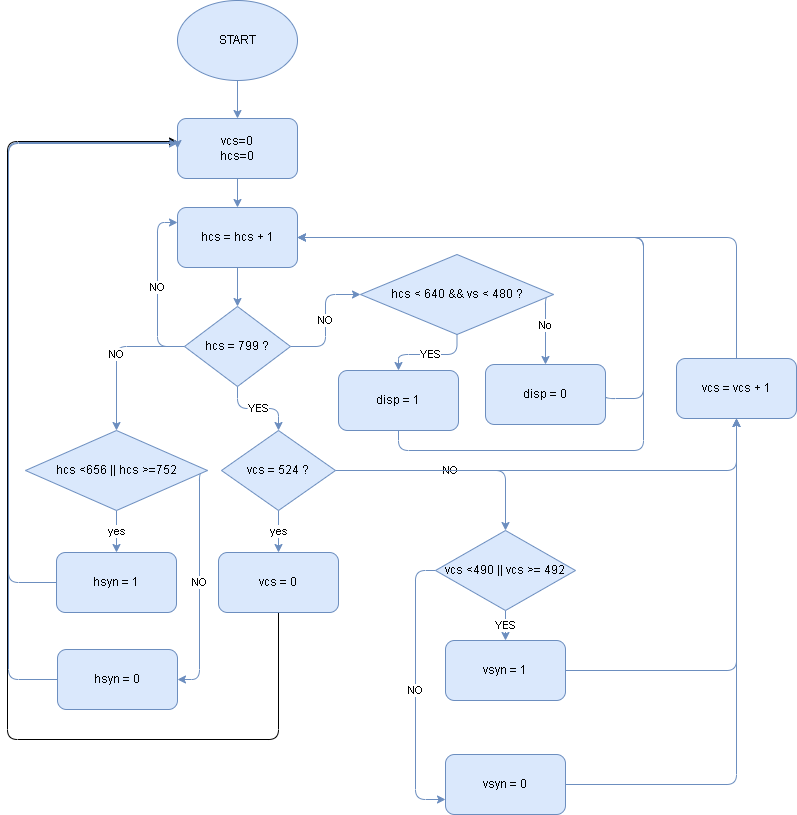
* Với các thiết lập như lý thuyết để bộ điều khiển hoạt động như mong muốn ta chọn âm thanh được phát ra ở tần số lấy mẫu là 8KHz, hai kênh, nhạc 16 bits.

+ AUD\_BCLK: 256Khz.

+ AUD\_XCK: 18.432 MHz.

+ AUD\_LRCK:8Khz.

## 4.9. Khối VGA

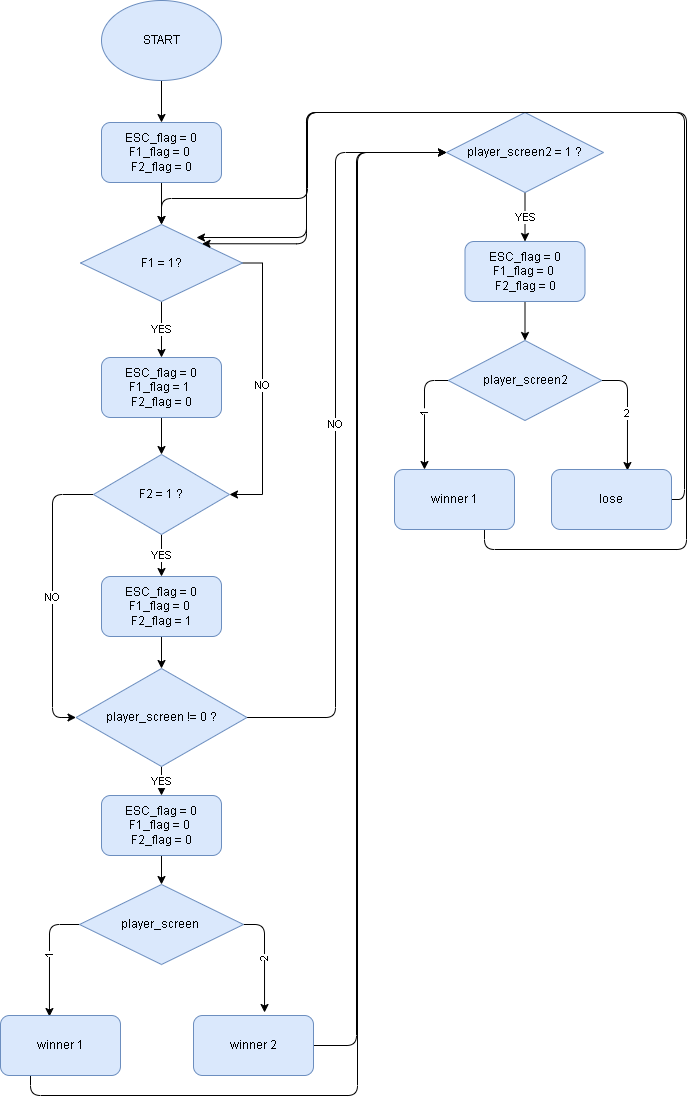
* Khối điều khiển VGA có sơ đồ giải thuật như sau: 

Hình 4.15 Sơ đồ điều khiển các tín hiệu điều khiển VGA.

* Xung clock điều khiển khối Vga có xung clock bằng 25 Mhz.
* VGA\_BLANK luôn bằng 1.
* VGA\_SYNC luôn bằng 0.

## 4.10. Khối hiển thị

* Khối hiển thị có sơ đồ như sau:

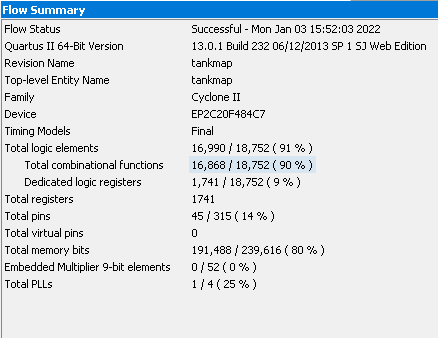


Hình 4.16 Sơ đồ điều khiển hiển thị trên màn hình.

* Tùy theo bàn phím ấn phím nào thì màn hình sẽ hiển thị phần đã quy định sẵn đó.

# KẾT QUẢ THỰC HIỆN

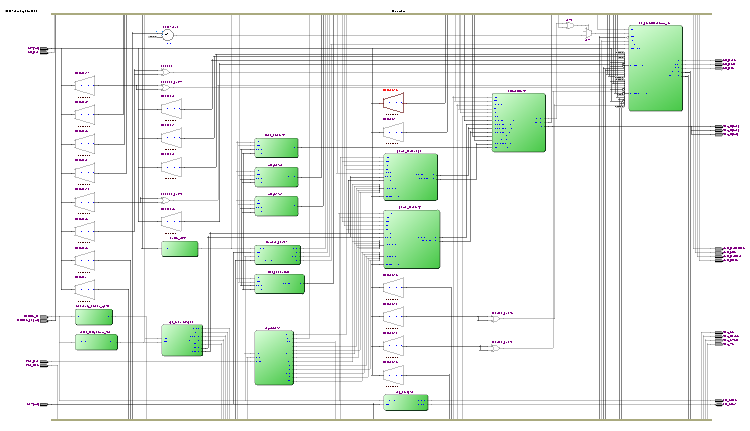
## 5.1 Kết quả biên dịch



Hình 5.1 Kết quả biên dịch.

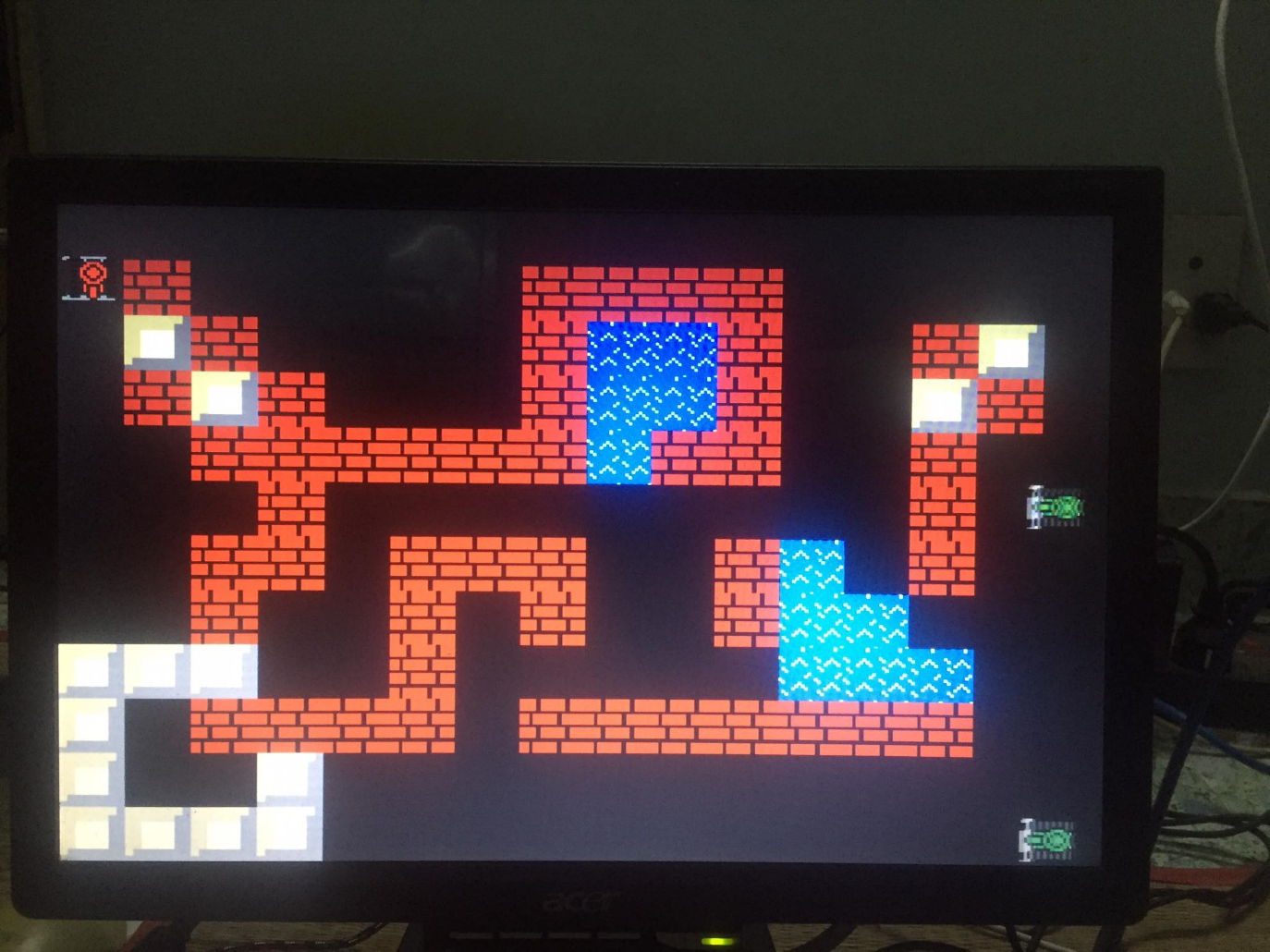
## 5.2 Kết quả khi chạy chương trình

- RTL:



Hình 5.2 RTL schematic.

- Màn hình trò chơi:



Hình 5.3 Màn hình trò chơi.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 6.1 Cách thực hiện chương trình

- Gắn thẻ nhớ đã có sẵn bài hát vào kit.

- Kết nối với bàn phím, màn hình, loa, dây nguồn

- Bật nguồn kit DE1 rồi chạy chương trình.

- Hướng dẫn trò chơi

+ ESC để trở về menu, F1 để chơi đôi, F2 để chơi đơn

+ Player 1: W S A D để di chuyển SPACE để bắn

+ Player 2: I J K L để di chuyển ENTER để bắn ( chơi đơn thì player 2 không thể điều khiển).

+ Khi đã có kết quả nhấn ESC hoặc F1 hoặc F2 để có thể bắt đầu ván mới

+ SW[9], SW[8] để chọn bài hát, SW[7] để reset nhạc.

+ SW[4:3] để chọn map với chế độ chơi đơn và SW[2:1] để chọn map chế độ chơi đôi

## 6.2 Kết luận

* Sau khi thực hiện đề tài em đã hiểu hơn về lập trình bằng ngôn ngữ phần cứng.
* Các yêu cầu ban đầu đề ra em đều hầu như đáp ứng được. khiến trò chơi vận hành theo như mong muốn, hình ảnh hiển thị chính xác, âm thanh nghe ổn.
* Đề tài sử dụng toàn bộ bằng ngôn ngữ verilog.
* Đã giải quyết được khó khăn ban đầu về bộ nhớ.

## Hướng phát triển

* Tuy đã hoàn thành cơ bản nhưng vẫn có khả năng cải thiện thêm như
  + Map đa dạng hơn nhiều cơ chế hơn.
  + Xe tăng địch có khả năng phát hiện và đuổi theo .
  + Có thêm nhiều chế độ hơn.
  + Thêm các chế độ nâng cấp
  + Hệ thống chạy mượt mà hơn
  + Tự động nạp lại map khi kết thúc trò chơi.

# TÀI LIỆU THAM KHẢO

1. Pong P.Chu “FPGA prototyping by Verilog examples,” Xilinx SpartanTM-3 version, Hoboken, New Jersey, John Wiley& Sons, 1959.
2. FPGA4 STUDENT, “Tic Tac Toe Game in Verilog and LogiSim”. <https://www.fpga4student.com/2017/06/tic-tac-toe-game-in-verilog-and-logisim.html>.
3. “DE1 development and education board user manual”,[Online].

<https://cs.colby.edu/courses/F21/cs232-labs/DE1_User_Manual.pdf>.

1. “Portable Internet Audio CODEC with Headphone Driver and Programmable Sample Rates”,[Online] <https://www.mouser.com/datasheet/2/76/WM8731_v4.9-532414.pdf>.
2. “A Strategical Approach for Implementing Digital Games on FPGA”,[Online].  <https://turcomat.org/index.php/turkbilmat/article/download/5037/4212/9359>.
3. “Lecture 12: SPI and SD cards”,[Online]. <http://www.dejazzer.com/ee379/lecture_notes/lec12_sd_card.pdf>.
4. Xgsomebra, “Tank-battle-1.0”,2017[Online], <https://github.com/XGsombra/Tank-Battle-1.0/stargazers>.
5. “How to use MMC/SDC”,[Online]. <http://elm-chan.org/docs/mmc/mmc_e.html>.
6. “24-bit audio CODEC”,[Online] <http://media.ee.ntu.edu.tw/personal/pcwu/dclab/dclab_09.pdf>.

# PHỤ LỤC

## Chuyển từ file ảnh sang định dạng mif

* Ta dùng matlab để chuyển sang file mif

function [outfname, rows, cols] = bmp2mif(infile, outfname, numrows, numcols)

img = imread(infile);

imgresized = imresize(img, [numrows numcols]);

[rows, cols, rgb] = size(imgresized);

imgscaled = imgresized/16 - 1;

imshow(imgscaled\*16);

fid = fopen(outfname,'w');

fprintf(fid,'-- %3ux%3u 12bit image color values\n\n',rows,cols);

fprintf(fid,'WIDTH = 12;\n');

fprintf(fid,'DEPTH = %4u;\n\n',rows\*cols);

fprintf(fid,'ADDRESS\_RADIX = UNS;\n');

fprintf(fid,'DATA\_RADIX = UNS;\n\n');

fprintf(fid,'CONTENT BEGIN\n');

count = 0;

for r = 1:rows

for c = 1:cols

red = uint16(imgscaled(r,c,1));

green = uint16(imgscaled(r,c,2));

blue = uint16(imgscaled(r,c,3));

color = red\*(256) + green\*16 + blue;

fprintf(fid,'%4u : %4u;\n',count, color);

count = count + 1;

end

end

fprintf(fid,'END;');

fclose(fid);

* Hoặc

%read the image

%I = imread('memscreen 40x40.png');

I = imread('screen.png');

imshow(I);

%Extract RED, GREEN and BLUE components from the image

R = I(:,:,1);

G = I(:,:,2);

B = I(:,:,3);

%make the numbers to be of double format for

R = double(R);

G = double(G);

B = double(B);

%Raise each member of the component by appropriate value.

R = R\*7/255; % 8 bits -> 3 bits

G = G\*7/255; % 8 bits -> 3 bits

B = B\*3/255; % 8 bits -> 2 bits

%tranlate to integer

R = uint8(R); % float -> uint8

G = uint8(G);

B = uint8(B);

%minus one cause sometimes conversion to integers rounds up the numbers wrongly

%R = R-1; % 3 bits -> max value is 111 (bin) -> 7 (dec)(hex)

%G = G-1;

%B = B-1; % 11 (bin) -> 3 (dec)(hex)

%shift bits and construct one Byte from 3 + 3 + 2 bits

G = bitshift(G, 2); % 3 << G (shift by 3 bits)

R = bitshift(R, 5); % 6 << B (shift by 6 bits)

COLOR = R+G+B; % R + 3 << G + 6 << B

%save variable COLOR to a file in HEX format for the chip to read

fileID = fopen ('mem\_screen.mif', 'w');

for j = 1:size(COLOR, 1)

for i = 1:size(COLOR, 2)

color\_out = COLOR(j,i);

%for mouse icon begin

%{

if color\_out == 182

color\_out = 0;

elseif color\_out == 255

color\_out = 17;

elseif color\_out == 0

color\_out = 16;

end

if color\_out == 0

fprintf (fileID, '%x', color\_out);

end

%}

%{

addr = (j-1)\*size(COLOR, 2) + i-1;

%}

%for mouse icon end

fprintf (fileID, '%x\n', color\_out); % COLOR (dec) -> print to file (hex)

end

end

% COLOR (dec) -> print to file (hex)

%save variable COLOR to a file in HEX format for the chip to read

%fileID = fopen ('Mickey.list', 'w');

%fprintf (fileID, '%x\n', COLOR); % COLOR (dec) -> print to file (hex)

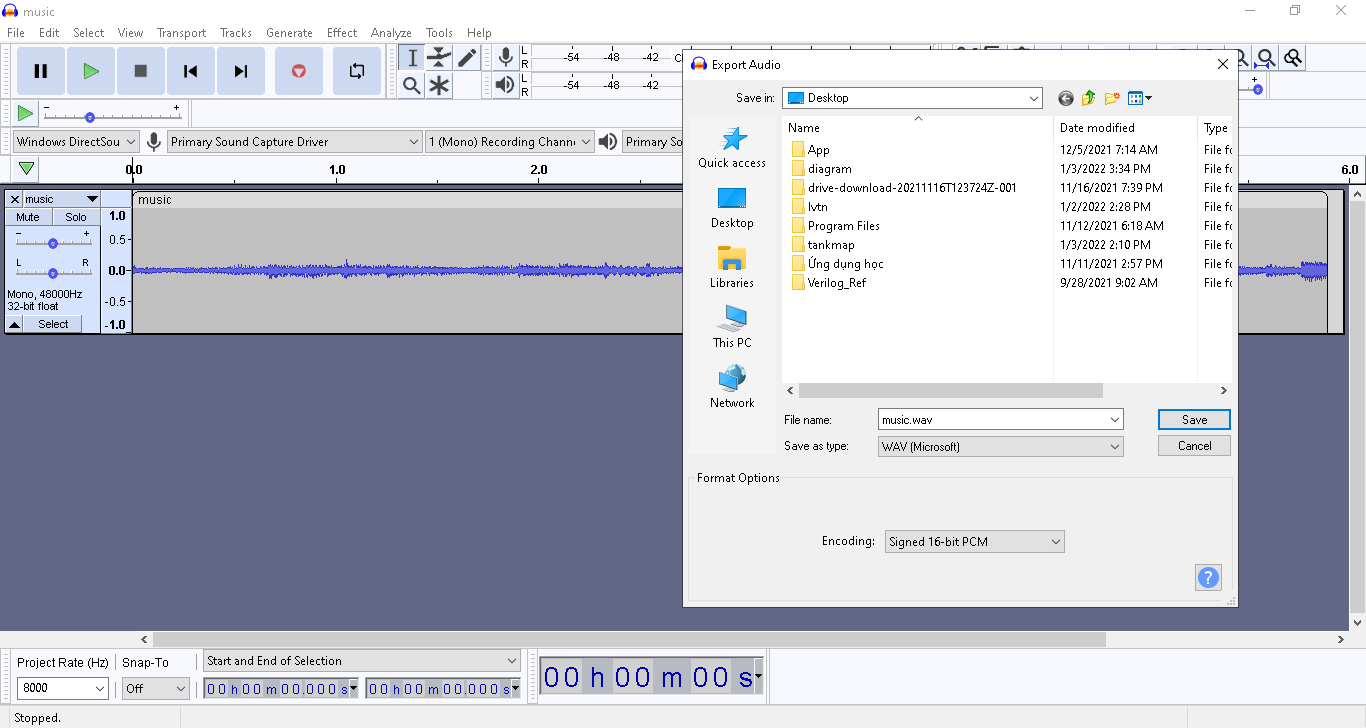
fclose (fileID);

%translate to hex to see how many lines

COLOR\_HEX = dec2hex(COLOR);

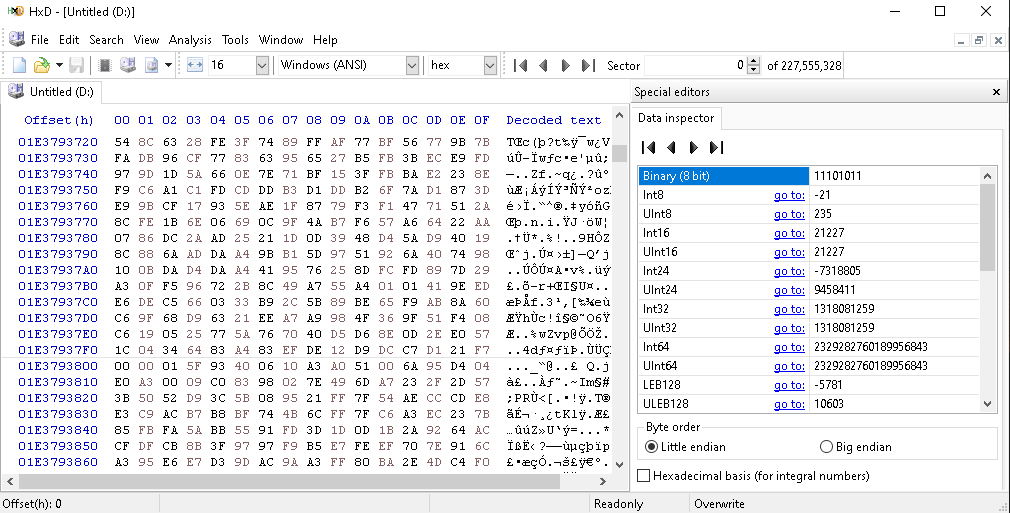
## Chuyển dữ liệu file nhạc vào thẻ nhớ SD card

* Muốn chuyển từ file nhạc mp3 tải trên mạng ta phải dùng phần mềm có tên là Audacity để chuyển từ file nhạc mp3 sang dịnh dạng wav với các thông số như sau:
  + Tải chương trình trên https://www.audacityteam.org/download
  + Nhạc Mono
  + 16 bit PCM
  + Tần số lấy mẫu 8Khz



Hình 8.1 Tạo file wav từ chương trình Audacity.

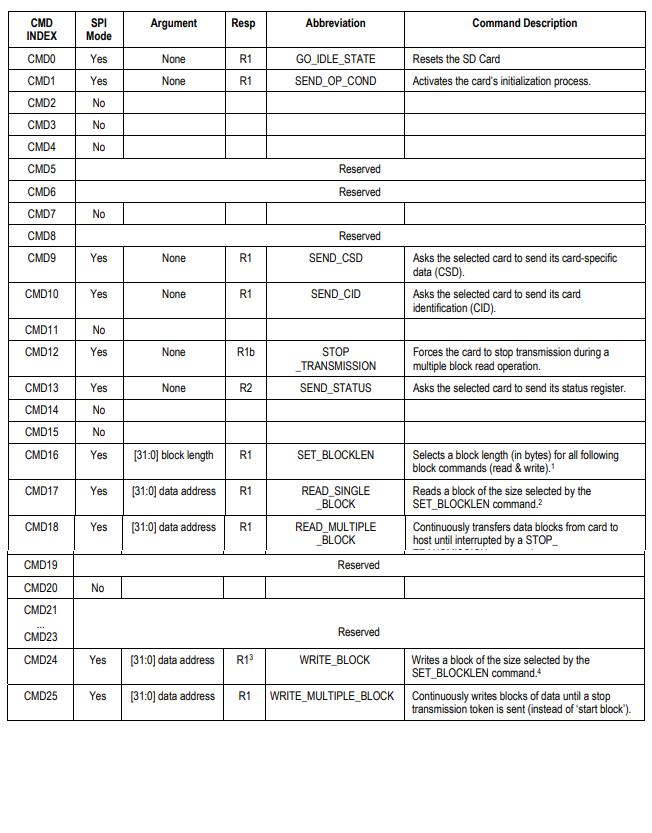
* Tiếp đó là tải file nhạc vào thẻ nhớ ta dùng phần mềm HxD tải ở <https://mh-nexus.de/en/downloads.php?product=HxD20>
* Sau đó ta mở file nhạc wav vừa tạo rồi copy đoạn mã hex trên ghi đè lên SD card
* Lưu ý tránh ghi đè lên vùng boot của SD card



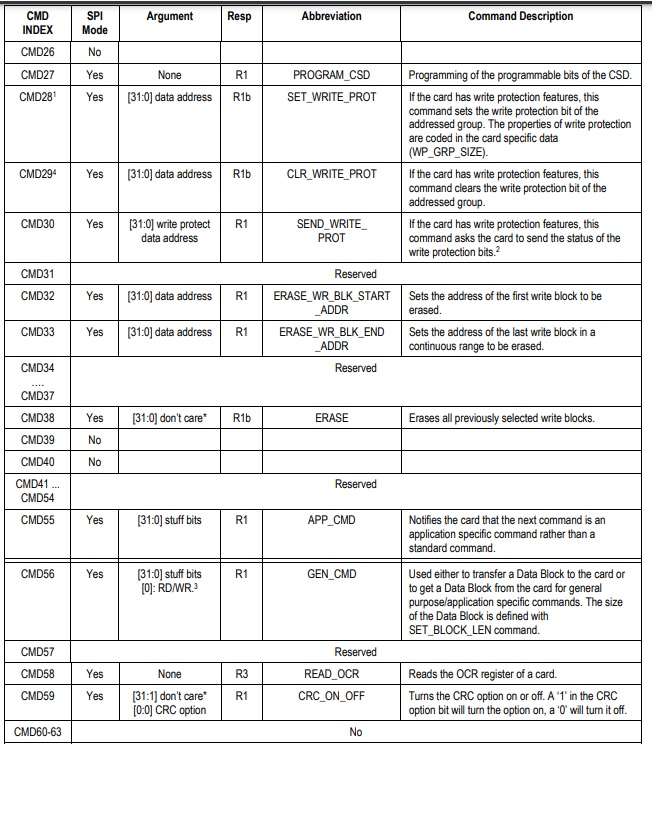
Hình 8.2 Tạo file nhạc ghi vào SD card.

## Một số bảng tra lệnh và số liệu

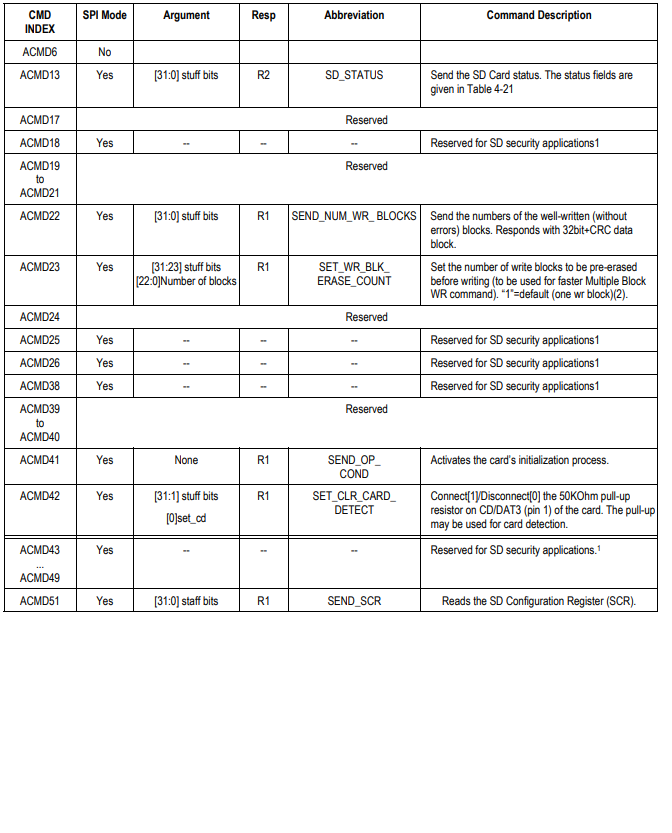
### Bảng lệnh SD card



Bảng 8.1 Một số lệnh SD card.

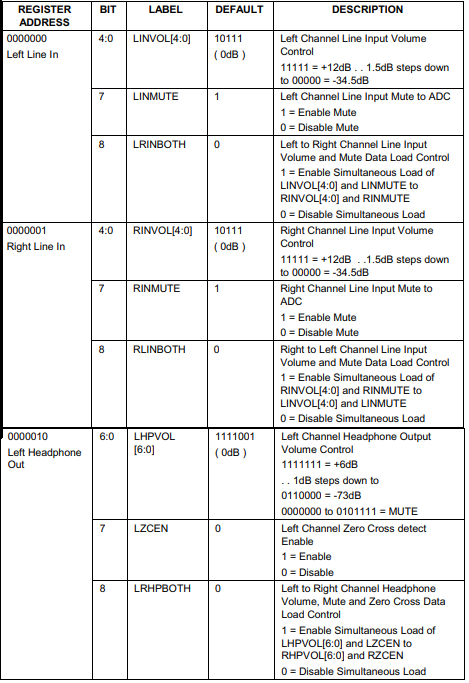


Bảng 8.2 Một số lệnh SD card.

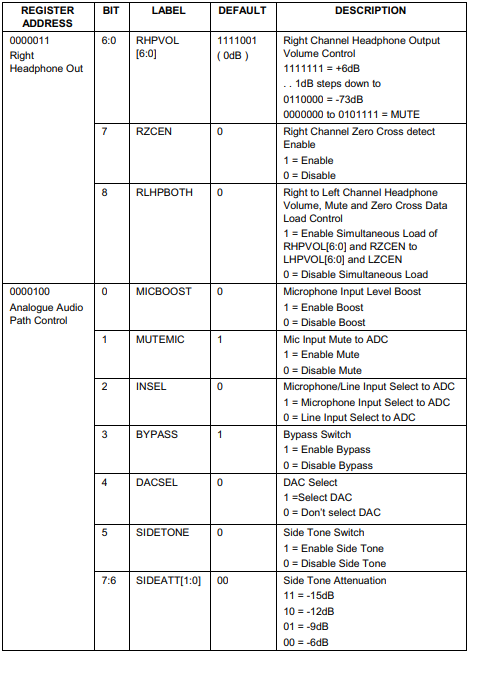


Bảng 8.3 Một số lệnh SD card.

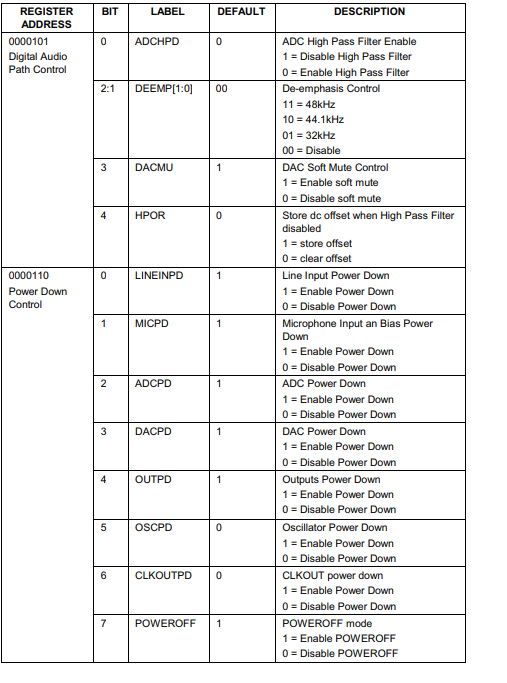
### 8.3.2 Bảng cấu hình WM8731



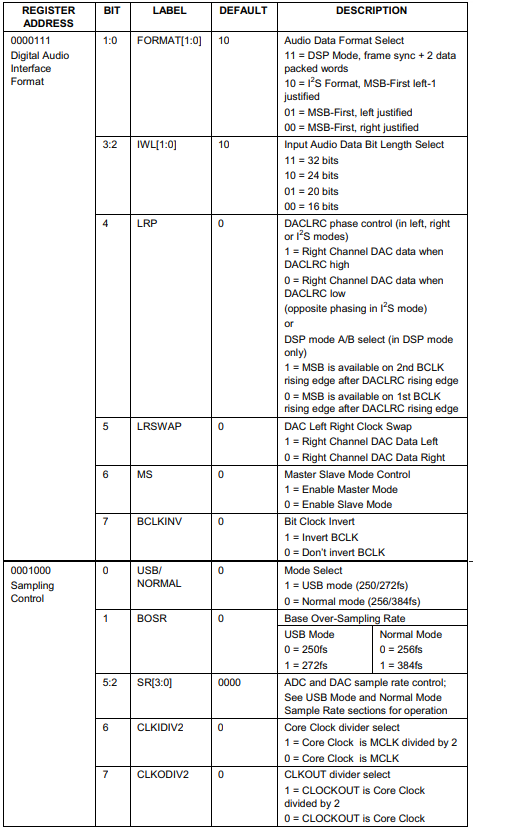
Bảng 8.4 Cấu hình Audio



Bảng 8.5 Cấu hình Audio



Bảng 8.6 Cấu hình Audio



Bảng 8.7 Cấu hình Audio



Bảng 8.8 Cấu hình Audio

-------------------------------------------HẾT-------------------------------------------