

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ



ĐỒ ÁN TỐT NGHIỆP
NGÀNH CNKT ĐIỆN TỬ - VIỄN THÔNG

**NÂNG CAO ĐỘ PHÂN GIẢI HÌNH ẢNH
BẰNG TRÍ TUỆ NHÂN TẠO TRÊN XILINX
ZYNQ ULTRASCALE+ MPSOC – ZCU102**

SVTH : HÀ NGUYỄN MINH PHƯỚC

MSSV : 20161245

Lớp : 20161CLVT2B

Tp. Hồ Chí Minh, tháng 06 năm 2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ



ĐỒ ÁN TỐT NGHIỆP
NGÀNH CNKT ĐIỆN TỬ - VIỄN THÔNG

**NÂNG CAO ĐỘ PHÂN GIẢI HÌNH ẢNH
BẰNG TRÍ TUỆ NHÂN TẠO TRÊN XILINX
ZYNQ ULTRASCALE+ MPSOC – ZCU102**

**SVTH : HÀ NGUYỄN MINH PHƯỚC
MSSV : 20161245
Lớp : 20161CLVT2B**

Tp. Hồ Chí Minh, tháng 06 năm 2024

TRANG THÔNG TIN ĐỒ ÁN

1. Thông tin sinh viên

Họ và tên sinh viên: HÀ NGUYỄN MINH PHƯỚC MSSV: 20161245

Email: 20161245@student.hcmute.edu.vn Điện thoại: 0377066783

2. Thông tin đề tài

- Tên đề tài: NÂNG CAO ĐỘ PHÂN GIẢI HÌNH ẢNH BẰNG TRÍ TUỆ NHÂN TẠO TRÊN XILINX ZYNQ ULTRASCALE+ MPSOC – ZCU102.
- Đơn vị quản lý: Bộ môn Kỹ thuật Máy tính – Viễn thông, Khoa Điện – Điện tử, Trường Đại học Sư phạm Kỹ thuật TP Hồ Chí Minh.
- Thời gian thực hiện đồ án: 19/02/2024 đến 01/06/2024 (15 tuần).
- Thời gian bảo vệ:

3. Lời cam đoan

Tôi cam đoan đồ án tốt nghiệp này là công trình nghiên cứu của tôi dưới sự hướng dẫn khoa học của GVC.Ths. Đậu Trọng Hiển. Những số liệu, nội dung nghiên cứu và kết quả trong đồ án là hoàn toàn trung thực. Các nội dung tham khảo từ nguồn bên ngoài được trích dẫn đầy đủ.

Tp. Hồ Chí Minh, ngày ... tháng 06 năm 2024

Hà Nguyễn Minh Phước

LỜI CẢM ƠN

Em xin chân thành cảm ơn quý thầy cô trong bộ môn CNKT Điện tử - Viễn thông đã dạy cho em kiến thức và giúp đỡ em giải quyết những khó khăn trong quá trình làm đồ án tốt nghiệp.

Đặc biệt xin chân thành cảm ơn thầy GVC.ThS. Đậu Trọng Hiển đã tận tình giúp đỡ và giải quyết những khó khăn trong quá trình chọn đề tài và hỗ trợ em xuyên suốt trong quá trình thực hiện.

Em cũng muốn bày tỏ lòng biết ơn đặc biệt đến Ths. Trương Quang Phúc đã điều kiện rất lớn cho em để có thể tiếp xúc và làm việc với phần cứng ZCU102. Sự chia sẻ kiến thức, động viên và tạo điều kiện thuận lợi từ Thầy đã đóng góp lớn vào sự thành công của đề tài, giúp em vượt qua những thách thức và hoàn thành công việc một cách xuất sắc.

Dù đã cố gắng hết sức, em thừa nhận rằng có những thiếu sót không mong muốn trong báo cáo. Em trân trọng và mong nhận được sự góp ý chân thành từ Thầy/Cô nhằm hoàn thiện báo cáo và tích lũy kinh nghiệm quý báu cho những bước tiến phía trước.

Cuối cùng, em xin kính chúc Thầy/Cô vluôn tràn đầy sức khỏe, hạnh phúc và đạt được nhiều thành công trong giảng dạy và công việc. Xin chân thành cảm ơn sự hỗ trợ và dành tặng lời chúc tốt đẹp nhất đến Thầy/Cô và mọi người.

Sinh viên thực hiện

Hà Nguyễn Minh Phước

TÓM TẮT

Trong thời đại công nghệ số phát triển nhanh chóng, nhu cầu về chất lượng hình ảnh cao trong các ứng dụng như y tế, giám sát an ninh, và truyền thông là rất lớn. Tuy nhiên, các thiết bị cảm biến hình ảnh thường có giới hạn về độ phân giải do chi phí và công nghệ. Do đó, nghiên cứu và phát triển các kỹ thuật siêu phân giải ảnh (Super-Resolution, SR) trở nên vô cùng cần thiết nhằm nâng cao chất lượng hình ảnh mà không cần thay đổi phần cứng cảm biến.

RCAN (Residual Channel Attention Network) là một trong những mô hình tiên tiến trong lĩnh vực siêu phân giải ảnh, đã chứng minh hiệu quả vượt trội so với các phương pháp truyền thống. Tuy nhiên, việc triển khai RCAN trên các thiết bị nhúng như FPGA, cụ thể là Zynq Ultrascale+ MPSoC - ZCU102, lại là một thách thức do yêu cầu về tài nguyên và hiệu năng và chưa được ứng dụng nhiều tại Việt Nam và trên thế giới. Việc chọn ZCU102 là do đây là một nền tảng FPGA mạnh mẽ, hỗ trợ nhiều ứng dụng xử lý ảnh phức tạp, và phù hợp để hiện thực hóa các mô hình học sâu như RCAN.

Đề tài sử dụng mô hình RCAN để thực hiện siêu phân giải ảnh trên nền tảng ZCU102. Quá trình thực hiện bao gồm các bước chính: Thiết kế IP Core cho mạng học sâu, cấu hình cho hệ thống nhúng chạy Linux cho ZCU102. Phân tích mô hình RCAN trên Vitis-AI và tiến hành huấn luyện, lượng tử hóa và biên dịch mô hình chạy trên ZCU102.

Kết quả đạt được cho thấy mô hình RCAN sau khi được triển khai trên ZCU102 có khả năng nâng cao chất lượng hình ảnh đầu vào đáng kể với độ phân giải cao hơn, đồng thời đáp ứng được yêu cầu về hiệu năng và tài nguyên.

ABSTRACT

In the rapidly advancing digital technology era, the demand for high-quality images in applications such as healthcare, security surveillance, and communication is immense. However, image sensor devices often have resolution limitations due to cost and technology constraints. Therefore, research and development of super-resolution (SR) techniques are crucial to enhance image quality without changing the sensor hardware.

RCAN (Residual Channel Attention Network) is one of the advanced models in the field of image super-resolution, demonstrating superior performance compared to traditional methods. However, implementing RCAN on embedded devices like FPGA, specifically the Zynq Ultrascale+ MPSoC - ZCU102, is a challenge due to resource and performance requirements and has not been widely applied in Vietnam or globally. The choice of ZCU102 is because it is a powerful FPGA platform that supports many complex image processing applications and is suitable for realizing deep learning models like RCAN.

The project uses the RCAN model to perform image super-resolution on the ZCU102 platform. The implementation process includes the main steps: Designing an IP Core for the deep learning network, configuring the embedded system running Linux for the ZCU102, analyzing the RCAN model on Vitis-AI, and conducting training, quantization, and compiling the model to run on the ZCU102.

The results show that the RCAN model, after being deployed on the ZCU102, significantly enhances the input image quality with higher resolution, while meeting performance and resource requirements.

MỤC LỤC

Trang tựa	TRANG
TRANG THÔNG TIN ĐỒ ÁN	iii
LỜI CẢM ƠN	iv
TÓM TẮT	v
ABSTRACT	vi
MỤC LỤC.....	vii
DANH MỤC CÁC TỪ VIẾT TẮT	x
DANH SÁCH BẢNG	xi
DANH SÁCH HÌNH.....	xii
CHƯƠNG 1: TỔNG QUAN.....	1
1.1 Tính cấp thiết của đề tài	1
1.2 Mục tiêu nghiên cứu	2
1.3 Tình hình nghiên cứu hiện nay	2
1.3.1. Tình hình nghiên cứu trong nước	2
1.3.2. Tình hình nghiên cứu ngoài nước	3
1.4 Đối tượng và phạm vi nghiên cứu	3
1.5 Bố cục đồ án tốt nghiệp	4
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	6
2.3. Tổng quan về Residual Channel Attention Networks (RCAN)	8
2.3.1. Kiến trúc mô hình (Network Architecture)	9
2.3.2. Trích xuất các đặc trưng cơ bản	9
2.3.3. Trích xuất các đặc trưng sâu bằng cơ chế Residual in Residual (RIR)	10
2.3.4. Channel Attention (CA)	10

2.3.5. Residual Channel Attention Bock (RCAB)	11
2.3.6. Mô-đun tăng kích thước	11
2.4. Các thông số quan trọng trong xử lý ảnh	12
2.4.1. Mean Squared Error.....	12
2.4.2. Peak Signal-to-Noise Ratio	13
2.4.3. Structural Similarity Index	13
2.5. Tổng quan về Zynq Ultrascale+ MPSoC - ZCU102.....	14
2.5.1. Kiến trúc của Zynq Ultrascale+ MPSoC - ZCU102	16
2.5.2. Đơn vị xử lý (PS) của Zynq Ultrascale+ MPSoC - ZCU102.....	17
2.5.3. Logic lập trình (FPGA Programmable Logic - PL)	30
2.5.4. Giao tiếp giữa hệ thống xử lý (PS) và Logic lập trình (PL).....	33
2.5.5. Tổng quan về chuẩn giao tiếp phần cứng AMBA AXI [9].....	35
2.6. Tổng quan về Intellectual Property (IP)	38
2.6.1. Giới thiệu về Intellectual Property (IP)	38
2.6.2. Nền tảng phần cứng của Deep Learning Processor Unit (DPU) [10]	39
CHƯƠNG 3: THIẾT KẾ HỆ THỐNG	41
3.1. Yêu cầu hệ thống.....	41
3.2. Sơ đồ khói và đặc điểm kỹ thuật hệ thống	41
3.3. Thiết kế hệ thống.....	42
3.3.1. Nền tảng phần mềm	42
3.3.2. Nền tảng phần cứng	69
Chương 4: KẾT QUẢ	78
4.1. Kết nối IP đầy đủ.....	78
4.2. Các tập tin xây dựng hệ thống nhúng.....	82
4.3. Triển khai mô hình siêu phân giải ảnh.....	83

4.3.1. Huấn luyện	83
4.3.2. Chạy mạng siêu phân giải ảnh trên phần cứng	84
4.3.3. Chạy mạng siêu phân giải ảnh trên Cloud (Kaggle)	93
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	95
5.1. Kết luận	95
5.2. Hướng phát triển	95
TÀI LIỆU THAM KHẢO.....	96
PHỤ LỤC.....	98

DANH MỤC CÁC TỪ VIẾT TẮT

SoC	System on Chip
FPGA	Field-Programmable Gate Array
AI	Artificial Intelligence
DL	Deep Learning
DPU	Deep Learning Processor Unit
CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
RL	Reinforcement Learning
RCAN	Residual Channel Attention Networks
GAN	Generative Adversarial Networks
SR	Super Resolution
VDSR	Very Deep Super Réolution
DRCN	Deeply-Recursive Convolutional Network
RIR	Residual in Residual
RCAB	Residual Channel Attention Bock
MSE	Mean Square Error
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Index Measure
MDM	MicroBlaze Debug Module
DCM	Digital Clock Manage

DANH SÁCH BẢNG

BẢNG	TRANG
Bảng 2.1. Thông số kỹ thuật của ARM Cortex A53 trong ZCU102	19
Bảng 2.2. Danh sách các giao diện ngoại vi I/O kết nối với MIO)	24
Bảng 2.3. Danh sách các giao diện ngoại vi I/O tốc độ cao kết nối với SIOU ..	28
Bảng 3.1. Thông tin các chân I/O của IP Zynq Ultrascale+ MPSoC	50
Bảng 3.2. Thông tin các chân của IP DPU	51
Bảng 3.3. Cấu hình DPU IP	53
Bảng 3.4. Chức năng các chân của IP Clocking Wizard [12]	55
Bảng 3.5. Cấu hình IP Clock Wizard [10]	56
Bảng 3.6. Chức năng các chân của IP processor System Reset [13]	57
Bảng 3.7. Các thư mục con trong file code của model RCAN	63
Bảng 3.8. Các thư mục con trong file float của model RCAN	65
Bảng 3.9. Các thư mục con trong file float của model RCAN	65
Bảng 3.10. Thông số quá trình huấn luyện mạng RCAN	66
Bảng 3.11. Các kết nối giữa giao diện thẻ SD với XCZU9EG (U1)	71
Bảng 3.12. Các kết nối Ethernet, XCZU9EG MPSoC với thiết bị PHY	73
Bảng 3.13. Chức năng các đèn led của giao diện Ethernet	75
Bảng 3.14. Kết nối giữa DPAUX	76
Bảng 4.1. Tài nguyên của các khối IP khi tổng hợp	79
Bảng 4.2. Thông số các giao diện và đường địa chỉ thiết kế IP học sâu	79
Bảng 4.3. Kích thước và dung lượng ảnh	86
Bảng 4.4. Thông số PSNR và SSIM của ảnh	93

DANH SÁCH HÌNH ẢNH

HÌNH	TRANG
<i>Hình 2.1. Ví dụ về siêu phân giải ảnh</i>	6
<i>Hình 2.2. Cấu trúc của một nơ-ron</i>	7
<i>Hình 2.3. Ví dụ về một lớp chập</i>	8
<i>Hình 2.4. Sơ đồ kiến trúc mạng RCAN</i>	9
<i>Hình 2.5. Kiến trúc Channel Attention của mạng RCAN</i>	10
<i>Hình 2.6. Kiến trúc RCAB của mạng RCAN</i>	11
<i>Hình 2.7. Thông tin một số linh kiện trên ZCU102</i>	15
<i>Hình 2.8. Sơ đồ kiến trúc tổng quát của ZCU102</i>	15
<i>Hình 2.9. Sơ đồ khái kiến trúc của ZCU102</i>	16
<i>Hình 2.10. Kiến trúc của ZCU102</i>	18
<i>Hình 2.11. ARM Cortex-A53</i>	18
<i>Hình 2.12. Sơ đồ khái APU trong ZCU102</i>	19
<i>Hình 2.13. Sơ đồ khái của RPU trong ZCU102</i>	21
<i>Hình 2.14. Sơ đồ khái của GPU trong ZCU102</i>	22
<i>Hình 2.15. Sơ đồ kết nối của MIO, EMIO và SIOU của ZYNQ MPSoC</i>	24
<i>Hình 2.16. Zynq MPSoC Programmable Logic</i>	31
<i>Hình 2.17. Kết nối và các thành phần bên trong của CLB (</i>	32
<i>Hình 2.18. Sơ đồ khái kết nối giữa PS và PL</i>	34

<i>Hình 2.19. Sơ đồ khái niệm các kênh giao tiếp của chuẩn AXI.....</i>	36
<i>Hình 2.20. Sơ đồ khái niệm quá trình đọc dữ liệu</i>	37
<i>Hình 2.21. Sơ đồ khái niệm quá trình ghi dữ liệu.....</i>	37
<i>Hình 2.22. Sơ đồ khái niệm của DPUCZDX8G</i>	39
<i>Hình 3.1. Sơ đồ khái niệm hệ thống siêu phân giải ảnh</i>	42
<i>Hình 3.2. Sơ đồ khái niệm nền tảng phần mềm</i>	42
<i>Hình 3.3. Quy trình thiết kế nền tảng phần mềm</i>	44
<i>Hình 3.4. Lưu đồ thiết kế IP tạo file XSA trên Vivado IDE</i>	44
<i>Hình 3.5. Lưu đồ thiết kế dự án Linux trên công cụ PetaLinux.....</i>	46
<i>Hình 3.6. Quy trình thiết kế hệ thống siêu phân giải ảnh trên công cụ Vitis AI.</i>	47
<i>Hình 3.7. Mô tả kết nối DPU được tích hợp vào Zynq Ultrascale+ MPSOC</i>	49
<i>Hình 3.8. IP Zynq Ultrascale+ MPSOC</i>	50
<i>Hình 3.9. IP Deep Learning Processor Unit (DPU)</i>	51
<i>Hình 3.10. Các tín hiệu clock cần thiết cho IP DPUCZDX8G.....</i>	54
<i>Hình 3.11. Kết nối giữa IP Clocking Wizard và IP DPU</i>	55
<i>Hình 3.12. Cấu hình IP Processor System Reset</i>	57
<i>Hình 3.13. Các thông số cấu hình cho dự án PetaLinux</i>	59
<i>Hình 3.14. Thiết lập các gói package người dùng</i>	59
<i>Hình 3.15. Kích hoạt SSH và vô hiệu hóa Dropbear.....</i>	60
<i>Hình 3.16. Vô hiệu hóa CPU IDLE</i>	61
<i>Hình 3.17. Kho thư viện Vitis AI Model Zoo</i>	62
<i>Hình 3.18. Môi trường làm việc Vitis AI trên Docker</i>	62

<i>Hình 3.19. Các tập tin có trong mô hình RCAN</i>	63
<i>Hình 3.21. Mô tả chức năng của quá trình lượng tử</i>	67
<i>Hình 3.22. Trình biên dịch Vitis AI</i>	68
<i>Hình 3.23. Quy trình thiết kế phần cứng</i>	69
<i>Hình 3.24. Kết nối nguồn cho ZCU102</i>	70
<i>Hình 3.25. Sơ đồ kết nối thẻ SD</i>	71
<i>Hình 3.26. Sơ đồ khói kết nối Ethernet</i>	73
<i>Hình 3.27. Sơ đồ kết nối mạch Reset</i>	74
<i>Hình 3.28. Sơ đồ kết nối của khói hiển thị VESA</i>	76
<i>Hình 4. 1. Sơ đồ các kết nối IP đầy đủ của hệ thống</i>	78
<i>Hình 4. 2. Các tập tin được tạo sau khi build kernel</i>	82
<i>Hình 4.3. ZCU102 vừa bật nguồn</i>	83
<i>Hình 4.4. ZCU102 khởi động hoàn tất</i>	83
<i>Hình 4.5. Biểu đồ giá trị mất mát trên 30 epoch của quá trình huấn luyện mạng siêu phân giải RCAN</i>	84
<i>Hình 4.6. Kết quả siêu phân giải ảnh 253027.png</i> của tập dữ liệu B100.....	84
<i>Hình 4.7. Kết quả siêu phân giải ảnh img070.png</i> của tập dữ liệu Urban100...	85
<i>Hình 4.8. Kết quả siêu phân giải ảnh butterfly.png</i> của tập dữ liệu Set5.....	85
<i>Hình 4.9. Kết quả siêu phân giải ảnh coastguard.png</i> của tập dữ liệu Set14	86
<i>Hình 4.10. Biểu đồ histogram ảnh 253027 dùng bicubic_x3</i>	87
<i>Hình 4.11. Biểu đồ histogram ảnh HR 253027</i>	87
<i>Hình 4.12. Biểu đồ histogram ảnh 253027 dùng RCAN</i>	87

<i>Hình 4.13. Biểu đồ histogram ảnh img070 dùng bicubic_x3</i>	88
<i>Hình 4.14. Biểu đồ histogram ảnh HR img070.....</i>	88
<i>Hình 4.15. Biểu đồ histogram ảnh img070 dùng RCAN.....</i>	88
<i>Hình 4.16. Biểu đồ histogram ảnh butterfly dùng bicubic_x3</i>	90
<i>Hình 4.17. Biểu đồ histogram ảnh HR butterfly</i>	90
<i>Hình 4.18. Biểu đồ histogram ảnh butterfly dùng RCAN</i>	90
<i>Hình 4.19. Biểu đồ histogram ảnh coastguard dùng bicubic_x3</i>	92
<i>Hình 4.20. Biểu đồ histogram ảnh HR coastguard.....</i>	92
<i>Hình 4.21. Biểu đồ histogram ảnh coastguard dùng RCAN.....</i>	92

CHƯƠNG 1: TỔNG QUAN

1.1 Tính cấp thiết của đề tài

Trong thời đại số hóa và phát triển công nghệ, hình ảnh đóng vai trò quan trọng trong giao tiếp và truyền tải thông tin. Chất lượng ảnh không chỉ ảnh hưởng đến trải nghiệm người dùng mà còn có ảnh hưởng đáng kể đến nhiều lĩnh vực khác nhau như y tế, an ninh, công nghiệp và nghệ thuật.

Trong lĩnh vực y tế, chất lượng ảnh có thể quyết định đúng sai trong việc chẩn đoán bệnh, giúp các chuyên gia y tế nhìn rõ các chi tiết quan trọng và đưa ra quyết định chính xác. Trên thực tế, việc nâng cao chất lượng ảnh có thể cứu sống mạng người và cải thiện chất lượng cuộc sống của hàng triệu người trên toàn cầu.

Trong lĩnh vực an ninh, chất lượng ảnh là yếu tố quan trọng để nhận diện khuôn mặt, phát hiện vật cản và giám sát các hoạt động đáng ngờ. Nhờ đó, việc nâng cao chất lượng ảnh có thể giúp tăng cường an ninh và đảm bảo sự an toàn cho cộng đồng.

Trong lĩnh vực truyền thông và quảng cáo, hình ảnh sắc nét và rõ ràng không chỉ thu hút sự chú ý của người xem mà còn truyền đạt thông điệp một cách hiệu quả hơn. Trong giáo dục, việc sử dụng hình ảnh độ phân giải cao giúp sinh viên tiếp cận kiến thức một cách trực quan và sinh động hơn, tăng cường hiểu biết và tương tác học tập. Trong lĩnh vực y tế, hình ảnh chất lượng cao đóng vai trò quan trọng trong chẩn đoán và điều trị bệnh tật.

Ngoài ra, việc có được hình ảnh rõ ràng và chi tiết cũng là yếu tố quan trọng trong công nghệ, đặc biệt là trong lĩnh vực xe tự lái, an ninh và giám sát. Hình ảnh độ phân giải cao giúp hệ thống phát hiện và nhận dạng đối tượng một cách chính xác và nhanh chóng, từ đó nâng cao độ tin cậy và hiệu suất của các ứng dụng này.

Tóm lại, trong một xã hội ngày càng phụ thuộc vào công nghệ và truyền thông hình ảnh, tính cấp thiết của việc độ phân giải hình ảnh không chỉ là vấn đề của cá

nhân mà còn là yếu tố quyết định sự thành công và phát triển của các ngành công nghiệp và cả xã hội.

1.2 Mục tiêu nghiên cứu

Mục tiêu nghiên cứu của đề tài là áp dụng các phương pháp và thuật toán học sâu (deep learning) để tăng cường chất lượng ảnh và cải thiện khả năng xử lý ảnh trên nền tảng phần cứng SoC. Cụ thể, các mục tiêu nghiên cứu có thể bao gồm:

Giảm nhiễu ảnh: Sử dụng mô hình deep learning như Convolutional Neural Networks (CNN) để giảm nhiễu trong ảnh. Phương pháp này có thể giúp loại bỏ nhiễu và tái tạo lại các chi tiết quan trọng trong ảnh.

Tăng cường độ tương phản: Áp dụng deep learning để tăng cường độ tương phản trong ảnh, từ việc làm rõ các đặc trưng và chi tiết đến việc tạo ra những hình ảnh sắc nét và sống động hơn.

Tái tạo chi tiết: Sử dụng mô hình deep learning như Residual Channel Attention Networks (RCAN), Generative Adversarial Networks (GAN), ... để tái tạo lại các chi tiết bị mất trong quá trình xử lý ảnh, giúp cải thiện độ phân giải và chất lượng tổng thể của ảnh.

Nâng cao độ phân giải ảnh: Sử dụng deep learning để tăng độ phân giải của ảnh thông qua các thuật toán như Super-Resolution của các thư viện Pytorch hay Tensorflow. Mục tiêu là tạo ra các ảnh có độ chi tiết cao hơn và rõ nét hơn, đặc biệt là khi làm việc với ảnh có độ phân giải thấp.

Tối ưu hóa hiệu suất tính toán: Tận dụng sức mạnh tính toán song song của FPGA trên SoC để tối ưu hóa hiệu suất xử lý ảnh. Mục tiêu là đạt được tốc độ xử lý cao và đáp ứng được yêu cầu thời gian thực trong việc nâng cao chất lượng ảnh.

1.3 Tình hình nghiên cứu hiện nay

1.3.1. Tình hình nghiên cứu trong nước

Ngành thiết kế và ứng dụng SoC là một ngành còn non trẻ và có nhiều triển vọng ở Việt Nam. Do đó, việc ứng dụng mạng học sâu trên SoC, cụ thể trên các Kit Zynq của Xilinx vẫn chưa được phát triển. Tính tới thời điểm hiện tại, chỉ có một số đề tài ứng dụng các IP Core được cung cấp sẵn bởi nhà phát triển để thiết kế hệ thống như là phát triển hệ điều hành nhúng chạy bằng Linux cho SoC, bộ lọc Sobel cho hình ảnh, hệ thống xử lý ảnh và video thời gian thực, ...

Bài luận văn tốt nghiệp của tác giả Nguyễn Hoàng Nhật Uyên về đề tài Triển khai bộ lọc Sobel trên Xilinx Zynq-7000 vào năm 2022 [1] là một đề tài nổi bật cho việc ứng dụng thiết kế trên SOC. Đề tài đạt được thành tựu đó là triển khai và ứng dụng được bộ lọc Sobel cho các hình ảnh trên Zynq-7000 và so sánh thông số ngõ ra của ảnh hình ảnh đã xử lý như PSRN, SSIM,... trên phần cứng Zynq-7000 và phần mềm OpenCV. Từ đó suy ra môi trường nào làm việc tối ưu hơn. Tuy nhiên, bài thiết kế vẫn có điểm hạn chế như không thể hiển thị kết quả đã được xử lý bởi bộ lọc thông qua cổng VGA trên Zynq-7000.

1.3.2. Tình hình nghiên cứu ngoài nước

Ứng dụng mạng học sâu trên SoC đã được ứng dụng rất nhiều như nhận dạng hành vi con người, nhận dạng biển báo giao thông, nâng cao độ phân giải,

Điển hình là hệ thống siêu phân giải ảnh dùng mạng GAN của Keras triển khai trên Zynq Ultrascale+ MPSOC ZCU102 của tác giả Gokula Krishnan Ravi [2]. Tác giả đã sử dụng mạng GAN của Keras tiến hành huấn luyện, lượng tử hóa và biên dịch mô hình trên phần mềm Vitis AI và triển khai trên Zynq Ultrascale+ MPSOC - ZCU102. Thiết kế của tác giả có nhiều điểm nổi bật có thể phân giải ảnh ở nhiều patch size khác nhau, từ đó có thể suy ra hiệu suất của DPU trên ZCU102. Tuy nhiên, thiết kế có một số điểm hạn chế như là mô hình siêu phân giải GAN không được tích hợp trên Vitis AI nên việc triển khai và thực thi ứng dụng rất khó khăn cho người mới bắt đầu.

1.4 Đối tượng và phạm vi nghiên cứu

1.4.1. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là áp dụng mạng học sâu (Deep Learning) và tận dụng khả năng tính toán của bộ xử lý học sâu (Deep Learning Processor Unit - DPU) trong SoC để nâng cao độ phân giải hình ảnh, từ việc giảm nhiễu, tăng cường độ tương phản, tái tạo chi tiết đến việc nâng cao độ phân giải ảnh. Đồng thời, cũng nhằm tối ưu hóa hiệu suất tính toán để đáp ứng yêu cầu thời gian thực và đạt được kết quả chất lượng cao trong việc xử lý ảnh.

1.4.2 Phạm vi nghiên cứu

Tìm hiểu về chức năng và kết nối các IP cần thiết giúp giao tiếp giữa bộ xử lý học sâu với ZYNQ Ultrascale+ MPSoC. Xây dựng hệ điều hành chạy trên phần cứng. Sử dụng mô hình mạng học sâu cho xử lý hình ảnh RCAN (Residual Channel Attention Networks) của thư viện Pytorch được tích hợp trên Vitis-AI Model Zoo. Tiến hành huấn luyện, lượng tử hóa, biên dịch model và triển khai trên Xilinx ZYNQ Ultrascale+ MPSoC-ZCU102.

1.5 Bố cục đồ án tốt nghiệp

Đồ án tốt nghiệp bao gồm tất cả 5 chương, trong đó tiêu đề và nội dung các chương được trình bày như sau:

CHƯƠNG 1: GIỚI THIỆU

Trình bày tính cấp thiết và lý do lựa chọn đề tài, ý nghĩa khoa học và thực tiễn đạt được khi hoàn thành, mục tiêu nghiên cứu, đối tượng, phạm vi và giới hạn của đề tài, phương pháp nghiên cứu, bố cục của ĐATN và phân định nội dung của vấn đề được nghiên cứu.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

Chương này sẽ cung cấp tổng quan về các thành phần, chức năng, kiến trúc của Zynq Ultrascale+ MPSoC ZCU102 Kit của Xilinx. Trình bày một cách chi tiết và cụ thể về mô hình RCAN của thư viện Pytorch.

CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG

Trình bày sơ đồ thiết kế của hệ thống và giải thích các phương pháp xử lý dữ liệu. Cuối cùng, thực hiện xây dựng mô hình hệ thống trên các phần mềm thích hợp, đảm bảo đáp ứng yêu cầu và mục tiêu của đề tài.

CHƯƠNG 4: TRIỂN KHAI VÀ KẾT QUẢ THỰC HIỆN

Trình bày chi tiết kết quả của từng khôi đã được thực hiện. Triển khai model đã được biên dịch lên ZYNQ Ultrascale+ MPSoC ZCU102.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

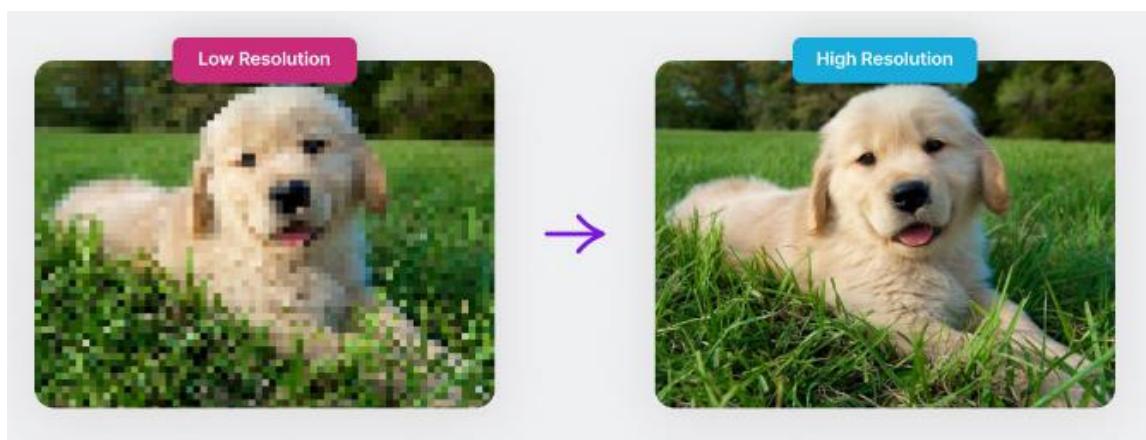
Dựa vào kết quả thu thập được từ chương 4, đưa ra kết luận tổng quan về những nội dung đã đạt được và những điểm còn chưa hoàn thiện của đề tài. Từ đó, đề xuất hướng phát triển tiếp theo nhằm giúp cải thiện hệ thống một cách tốt hơn và ứng dụng vào thực tiễn hiệu quả.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về siêu phân giải hình ảnh

Siêu phân giải hình ảnh (Image Super-Resolution) là một lĩnh vực trong thị giác máy tính và xử lý ảnh nhằm làm cho một hình ảnh đầu vào rõ ràng hơn. Cụ thể, nó tập trung vào việc cải thiện hình ảnh có độ phân giải thấp (LR) bằng cách ước tính một phiên bản có độ phân giải cao (HR) để điền vào các chi tiết bị thiếu. Đây là một phần của các vấn đề phục hồi ảnh, như loại bỏ nhiễu và làm rõ ảnh mờ. Siêu phân giải hình ảnh được áp dụng rộng rãi trong các lĩnh vực như hình ảnh y học, nhiếp ảnh trên điện thoại di động, và giám sát.

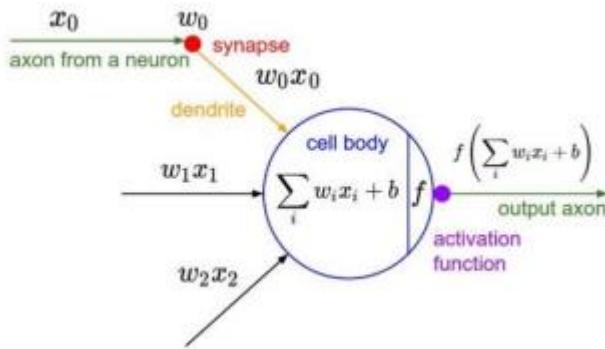
Vấn đề siêu phân giải hình ảnh không có lời giải duy nhất, tức là từ một hình ảnh độ phân giải thấp có thể tạo ra nhiều hình ảnh độ phân giải cao khác nhau. Có nhiều phương pháp cổ điển để giải quyết vấn đề này, bao gồm các phương pháp dựa trên mảng, dựa trên cạnh, biểu diễn thưa thớt và các phương pháp thống kê. Tuy nhiên, với sự phát triển của các mô hình học sâu, các phương pháp học sâu cho siêu phân giải hình ảnh đã được nghiên cứu và phát triển mạnh mẽ. Các mô hình học sâu này thường cho ra kết quả chất lượng cao hơn so với các phương pháp cổ điển.



Hình 2.1. Ví dụ về siêu phân giải ảnh

2.2 Tổng quan về mạng học sâu (Deep Learning Neuron Network)

Học sâu giải quyết các vấn đề liên quan đến việc học đặc trưng của các hiện tượng khác nhau. Học sâu là một nhánh của học máy, chủ yếu được sử dụng để trích xuất các đặc trưng hữu ích chưa biết để biểu diễn một tập dữ liệu cho trước. Học sâu đạt được mức độ học này thông qua việc sử dụng Mạng nơ-ron. Mạng nơ-ron, hay còn gọi là Perceptron đa lớp, được lấy cảm hứng từ cấu trúc của não người, bao gồm các lớp kết nối với nhau, mỗi phần tử được gọi là một nơ-ron. Một nơ-ron tính toán tổng trọng số hoặc tổ hợp tuyến tính của các đầu vào và "kích hoạt" nếu tổng vượt quá ngưỡng. Hàm ngưỡng này có thể được mô hình hóa bằng các hàm kích hoạt như sigmoid, softmax, relu, tanh, v.v. Về cơ bản, một mạng nơ-ron bao gồm nhiều hàm đơn giản, cùng nhau đóng góp vào việc ánh xạ một đầu vào cụ thể tới đầu ra. Điều này giúp mạng nơ-ron có khả năng ánh xạ mọi loại dữ liệu bằng cách trích xuất các đặc trưng hữu ích chưa biết. Vì khả năng ánh xạ qua nhiều loại tập dữ liệu, Mạng nơ-ron sâu còn được gọi là Bộ xấp xỉ phổ quát.

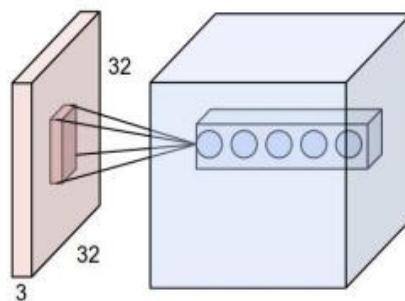


Hình 2.2. Cấu trúc của một nơ-ron

Có một số thuật toán phổ biến trong học sâu:

- **Mạng nơ-ron tích chập** (Convolutional Neural Networks - CNN): Chủ yếu được sử dụng trong xử lý hình ảnh và thị giác máy tính. CNN dùng các lớp tích chập để học các đặc trưng hình ảnh từ dữ liệu, hiệu quả trong các nhiệm vụ như nhận dạng đối tượng và phân loại hình ảnh.

- **Mạng nơ-ron hồi quy** (Recurrent Neural Networks - RNN): Phù hợp với dữ liệu chuỗi và dữ liệu có mối quan hệ thời gian. RNN có khả năng lưu trữ thông tin từ các trạng thái trước đó, thường được sử dụng trong các ứng dụng dự đoán và mô hình hóa chuỗi thời gian.
- **Mạng nơ-ron đối kháng sinh** (Generative Adversarial Networks - GAN): Được dùng để tạo ra dữ liệu mới dựa trên phân phối đã học từ dữ liệu huấn luyện. GAN bao gồm hai mạng nơ-ron cạnh tranh với nhau để tạo ra dữ liệu mới.
- **Mạng nơ-ron tự mã hóa** (Autoencoders): Được sử dụng để trích xuất đặc trưng và giảm chiều dữ liệu, có thể áp dụng trong nhiều nhiệm vụ như nén dữ liệu và tạo ảnh mới.
- **Học tăng cường** (Reinforcement Learning - RL): Sử dụng để học từ tương tác với môi trường, với mục tiêu tối đa hóa phần thưởng nhận được từ môi trường. RL được ứng dụng trong nhiều lĩnh vực như trò chơi và robot học.



Hình 2.3. Ví dụ về một lớp chập

2.3. Tổng quan về Residual Channel Attention Networks (RCAN)

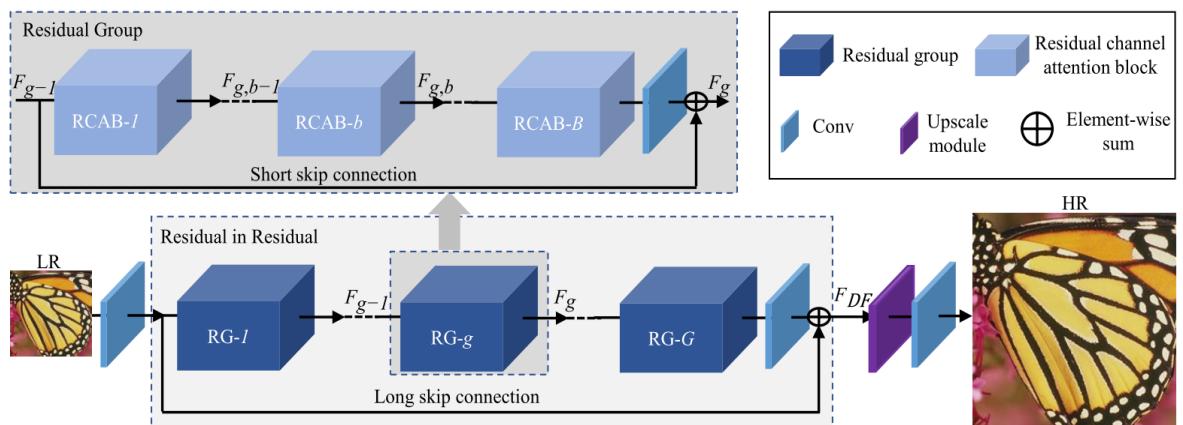
Nhiều phương pháp siêu phân giải hình ảnh (SR) đã được nghiên cứu trong cộng đồng thị giác máy tính. Cơ chế chú ý (Attention) rất phổ biến trong các tác vụ thị giác cấp cao.

CNN sâu cho SR được thực hiện bằng mạng SRCNN cho hình ảnh siêu phân giải [3] và đạt hiệu suất vượt trội. Bằng cách giới thiệu phương pháp học thặng dư (residual learning) để giảm độ phức tạp trong việc huấn luyện, siêu phân

giải hình ảnh sử dụng mạng VDSR [4] và DRCN [5] với 20 lớp và đạt được cải thiện đáng kể về độ chính xác. Những phương pháp này phải nội suy đầu vào Low Resolution (LR) đến kích thước mong muốn, điều này không thể tránh khỏi việc mất một số chi tiết và tăng đáng kể tính toán. Vì vậy, mạng RCAN ra đời nhằm để giải quyết những vấn đề trên.

2.3.1. Kiến trúc mô hình (Network Architecture)

Như được thể hiện trong hình 2.4 ở dưới, RCAN chủ yếu bao gồm bốn phần: trích xuất đặc trưng cơ bản (shallow feature extraction), trích xuất đặc trưng sâu bằng cách dùng cấu trúc Residual in Residual (RIR), mô-đun tăng kích thước, và phần tái tạo.



Hình 2.4. Sơ đồ kiến trúc mạng RCAN

2.3.2. Trích xuất các đặc trưng cơ bản

Đây là bước đầu tiên của quá trình siêu phân giải ảnh gồm một lớp tích chập có kích thước kernel là 3x3 hoặc 3x5. [6] Đây là quá trình trích xuất các đặc trưng cơ bản từ dữ liệu đầu vào một cách đơn giản, đặc trưng này thường không chứa nhiều thông tin phức tạp mà chỉ là những đặc điểm cơ bản của dữ liệu. Công thức được mô tả ở dưới đây:

$$F_0 = H_{SF}(I_{LR}) \quad (2.1)$$

Trong đó: F_0 là ngõ ra của phép chập.

H_{SF} biểu thị cho phép toán chập.

I_{LR} là ảnh ngõ vào có độ phân giải thấp.

2.3.3. Trích xuất các đặc trưng sâu bằng cơ chế Residual in Residual (RIR)

Ngõ ra F_0 của bài toán trích xuất đặc trưng cơ bản được sử dụng cho bước trích xuất đặc trưng sâu RIR. [6] Công thức được mô tả ở dưới đây:

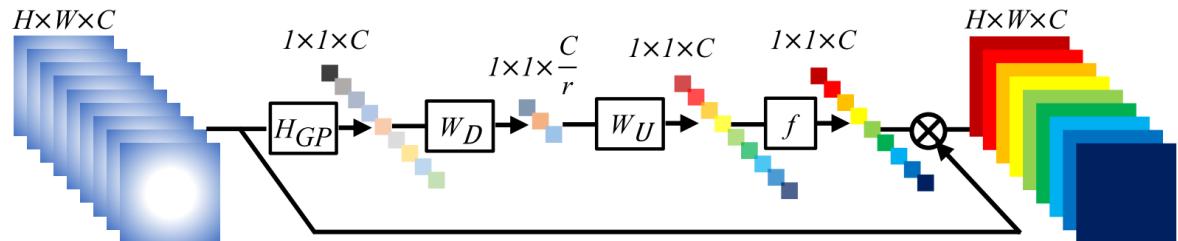
$$F_{DF} = H_{RIR}(F_0) \quad (2.2)$$

Trong đó, H_{RIR} biểu thị phần dư (residual) rất sâu chứa các nhóm dư G (RG). Mỗi RG chưa B khôi chú ý kênh dư RCAB (Residual Channel Attention Blocks). [6] Cấu trúc RIR cho phép huấn luyện CNN rất sâu (trên 400 lớp) cho hình ảnh có hiệu suất cao. Một RG (Residual Group) trong nhóm g-th được xây dựng như sau:

$$F_g = H_g(F_{g-1}) = H_g(H_{g-1}(\dots H_1(F_0) \dots)) \quad (2.3)$$

Trong đó, H_g biểu thị hàm của nhóm dư thứ g. F_{g-1} và F_g lần lượt là ngõ vào và ngõ ra của nhóm dư thứ g.

2.3.4. Channel Attention (CA)



Hình 2.5. Kiến trúc Channel Attention của mạng RCAN

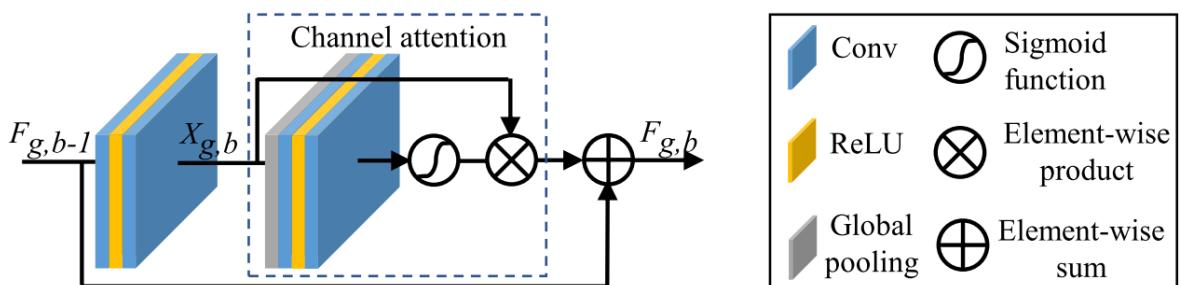
Để làm cho mạng tập trung vào các đặc trưng nổi bật, cần phải khai thác sự tương phản giữa các kênh đặc trưng. Dùng lớp Global Average Pooling được thể hiện trong hình 2.5, giả sử $X = [x_1, \dots, x_c, \dots, x_C]$ là ngõ đầu vào, có C bản đồ đặc trưng (feature map) với kích thước $H \times W$. Thông kê theo kênh $z \in R^C$ có thể được

tính được bằng cách thu gọn X qua các chiều không gian $H \times W$. Sau đó, phần tử thứ c của z được xác định bởi công thức: [6]

$$z_c = H_{GP}(x_c) = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W x_c(i, j) \quad (2.4)$$

Trong đó $x_c(i, j)$ là giá trị tại vị trí (i, j) của đặc trưng thứ c của x . $HGP(\cdot)$ biểu thị hàm tổng hợp toàn cục. Thông kê kênh như vậy có thể được coi là một tập hợp các mô tả cục bộ.

2.3.5. Residual Channel Attention Block (RCAB)



Hình 2.6. Kiến trúc RCAB của mạng RCAN

Các nhóm dư thừa (residual groups) cho phép các phần chính của mạng tập trung vào các thành phần thông tin hơn của các đặc trưng của LR. Tích hợp CA vào RB ta được RCAB (hình 2.6). Đổi với RB thứ b trong RG thứ g , ta có: [6]

$$F_{g,b} = F_{g,b} - 1 + R_{g,b}(X_{g,b}) \cdot X_{g,b} \quad (2.5)$$

Trong đó, $R_{g,b}$ biểu thị chức năng của CA. $F_{g,b}$ và $F_{g,b-1}$ là ngõ vào và ngõ ra của RCAB. $F_{g,b-1}$ được học từ $X_{g,b-1}$ là kết quả của 2 lớp tích chập chồng lên nhau. [6]

$$X_{g,b-1} = W_{g,b}^2 \delta(W_{g,b}^{-1} F_{g,b-1}) \quad (2.6)$$

Trong đó $W_{g,b}^{-1}$ và $W_{g,b}^2$ là trọng số của 2 lớp tích chập chồng lên nhau ở trong RCAB.

2.3.6. Mô-đun tăng kích thước

Dựa vào sơ đồ kiến trúc mạng ở hình 2.4. mô-đun này nằm ở ngõ ra của RIR, sau đó được tăng kích thước bằng công thức: [6]

$$F_{UP} = H_{UP}(F_{DF}) \quad (2.7)$$

Sau khi tăng kích thước, hình ảnh sẽ được cấu trúc lại bằng một lớp tích chập: [6]

$$I_{SR} = H_{REC}(F_{UP}) = H_{RCAN}(I_{LR}) \quad (2.8)$$

Trong đó HREC và HRCAN lần lượt là lớp tái cấu trúc và chức năng của mạng RCAN.

Cuối cùng là tính măt mát của mạng: [6]

$$L(O) = \frac{1}{N} \sum_{i=1}^N ||H_{RCAN}(I_{LR}^i) - I_{HR}^i||_1 \quad (2.9)$$

Trong cấu trúc của RIR, thông số của RG được đặt là G=10.

Trong mỗi RG, thông số của RCAB được đặt là 20.

2.4. Các thông số quan trọng trong xử lý ảnh

2.4.1. Mean Squared Error

Mean Squared Error (MSE) được gọi là lỗi bình phương trung bình, một chỉ số thường được sử dụng trong xử lý tín hiệu và xử lý hình ảnh để đo lường sự khác biệt giữa hai tín hiệu, hai hình ảnh hoặc hai tập dữ liệu.

MSE được tính bằng cách lấy trung bình của bình phương của sự khác biệt giữa các giá trị của hai tín hiệu:

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (2.10)$$

Trong đó:

- n là số lượng mẫu.
- y_i là giá trị thực tế của mẫu thứ i .
- \hat{y}_i là giá trị dự báo của mẫu thứ i .

MSE cho biết mức độ trung bình của sự khác biệt giữa hai tín hiệu. Giá trị MSE càng nhỏ thì hai tín hiệu càng giống nhau. Trong xử lý hình ảnh, MSE thường

được sử dụng để đo lường mức độ mất mát thông tin khi nén hình ảnh hoặc khi thực hiện các phép biến đổi hình ảnh. Đối với một hình ảnh nén, MSE thấp có thể cho thấy chất lượng hình ảnh cao hơn, trong khi MSE cao hơn có thể chỉ ra sự mất mát thông tin lớn hơn và chất lượng hình ảnh kém hơn.

2.4.2. Peak Signal-to-Noise Ratio

PSNR là viết tắt của "Peak Signal-to-Noise Ratio" (Tỷ lệ Tín Hiệu-Độ Nghiễm Cực Đại), một chỉ số được sử dụng phổ biến trong xử lý hình ảnh và video để đo lường chất lượng hình ảnh sau khi nén hoặc xử lý. PSNR thường được sử dụng để so sánh sự tương đồng giữa hình ảnh gốc và hình ảnh đã xử lý, hoặc giữa hai hình ảnh khác nhau.

Công thức tính PSNR dựa trên sự khác biệt giữa hình ảnh gốc và hình ảnh đã xử lý, cũng như trên giá trị tối đa có thể của tín hiệu (thường là 255 cho hình ảnh 8-bit):

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE} \right) \quad (2.11)$$

Trong đó: MAX là giá trị tối đa của tín hiệu (Ví dụ: 255 cho hình ảnh 8 bit).

MSE: là độ lệch bình phương trung bình giữa hai hình ảnh.

PSNR được tính bằng đơn vị decibel (dB). Một giá trị PSNR cao hơn cho thấy sự tương đồng lớn hơn giữa hai hình ảnh, tức là hình ảnh đã xử lý gần giống với hình ảnh gốc. Trong khi đó, một giá trị PSNR thấp hơn thường chỉ ra sự mất mát thông tin lớn hơn và sự tương đồng kém giữa hai hình ảnh.

2.4.3. Structural Similarity Index

SSIM là viết tắt của "Structural Similarity Index Measure" (Chỉ số đo lường Tương đồng Cấu Trúc), là một phương pháp được sử dụng rộng rãi để đánh giá sự tương đồng giữa hai hình ảnh. Khác với PSNR và MSE, SSIM không chỉ xem xét sự khác biệt pixel-by-pixel mà còn tính đến các yếu tố cấu trúc và cảm nhận người nhìn.

SSIM được tính bằng cách so sánh ba yếu tố chính của hình ảnh: sự tương phản, sự độ tương phản và cấu trúc. Công thức tổng quát cho SSIM là:

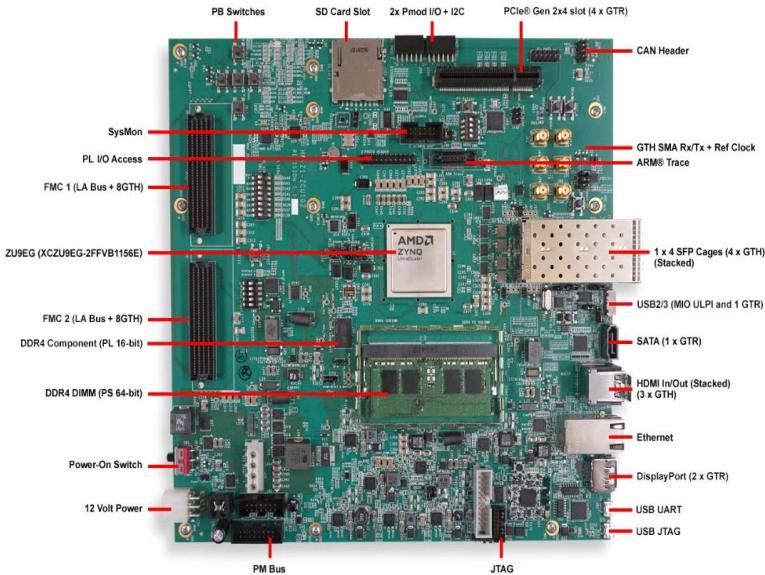
$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (2.12)$$

Giá trị của SSIM thường nằm trong khoảng từ -1 đến 1, với giá trị 1 thể hiện sự tương đồng hoàn hảo giữa hai hình ảnh. Một giá trị SSIM gần 1 chỉ ra rằng hai hình ảnh rất giống nhau. Trong khi đó, một giá trị SSIM gần 0 hoặc âm chỉ ra rằng hai hình ảnh rất khác biệt.

SSIM thường được sử dụng để đánh giá chất lượng hình ảnh sau khi xử lý, như trong quá trình nén, tăng cường hoặc phục hồi hình ảnh. Nó cung cấp một cái nhìn tổng quan về sự tương đồng giữa hai hình ảnh từ góc độ cảm nhận người nhìn.

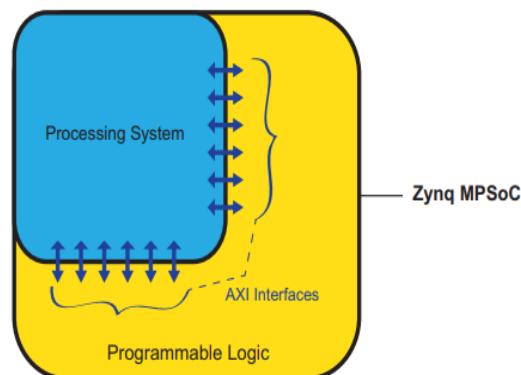
2.5. Tổng quan về Zynq Ultrascale+ MPSoC - ZCU102

Zynq UltraScale+ MPSoC_ZCU102 là một sản phẩm của Xilinx, được ra mắt vào năm 2015. ZCU102 là một SoC (System on Chip), tích hợp một hệ thống xử lý đa lõi dựa trên ARM Cortex-A53 và Cortex-R5 cùng với một FPGA Xilinx 16 nm (PL). CPU ARM Cortex-A53 và Cortex-R5 chịu trách nhiệm chính trong thành phần Processing System (PS) và được điều khiển bởi bộ điều khiển GIC (Generic Interrupt Controller), có sẵn bộ nhớ on-chip, giao diện bộ nhớ ngoại và các giao tiếp ngoại vi đa dạng.



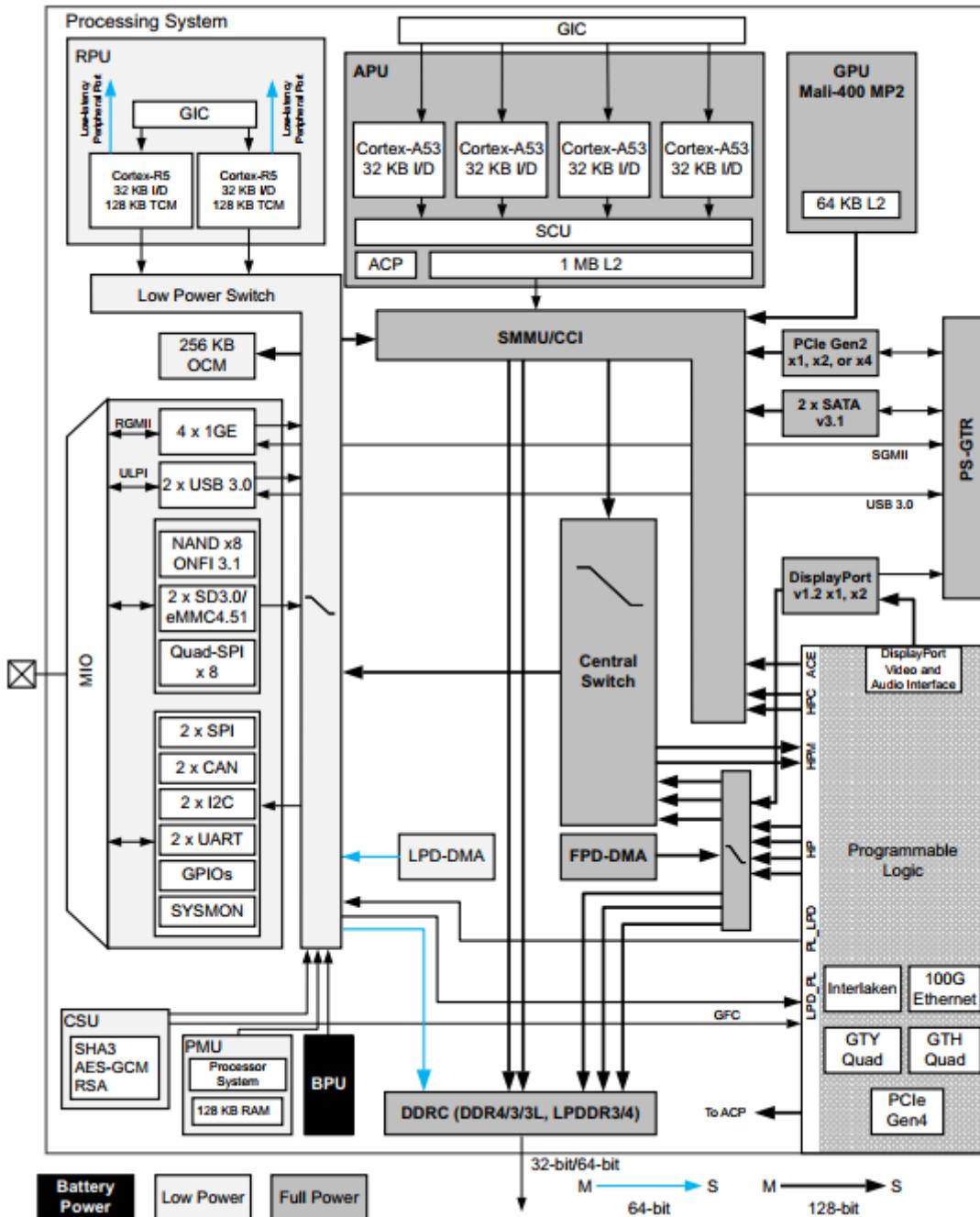
Hình 2.7. Thông tin một số linh kiện trên ZCU102

Nền tảng ZCU102 cung cấp sự hỗ trợ tốt cho việc phát triển các thiết kế SoC, trong đó các khối IP có thể được kết nối thông qua giao diện bus AXI. Điều này tạo điều kiện thuận lợi cho tích hợp linh hoạt và hiệu quả của phần cứng và phần mềm trên cùng một chip, giúp tối ưu hóa hiệu suất và linh hoạt trong quá trình phát triển sản phẩm. Giao diện bus AXI là giao diện chính dùng để giao tiếp giữa các linh kiện trên ZCU102.



Hình 2.8. Sơ đồ kiến trúc tổng quát của ZCU102

2.5.1. Kiến trúc của Zynq Ultrascale+ MPSoC - ZCU102



Hình 2.9. Sơ đồ khái niệm kiến trúc của ZCU102

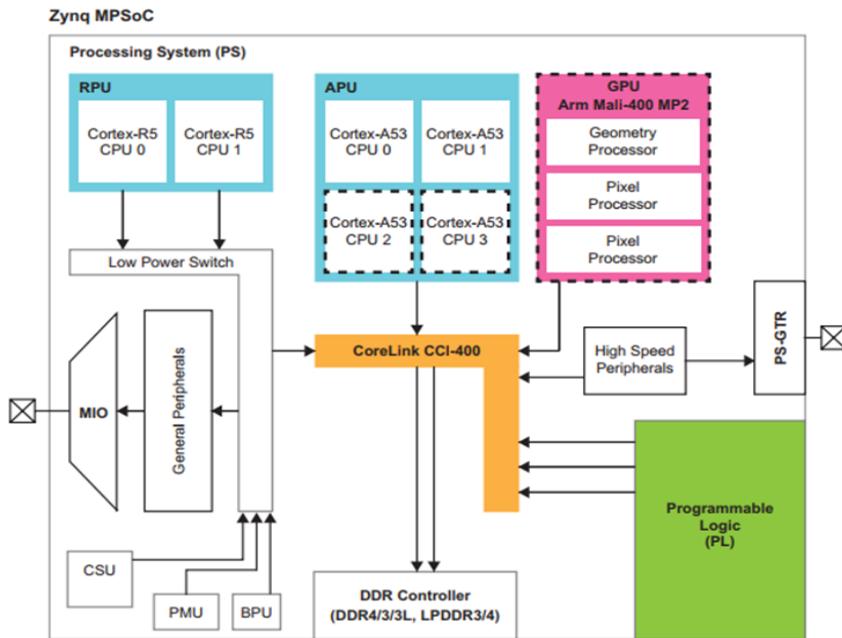
ZCU102 Evaluation Kit có những đặc điểm và tính năng đáng chú ý như sau:

- Hệ thống xử lý (Processing System - PS) [7] [8]:

- Đơn vị xử lý (APU): Bao gồm các lõi ARM Cortex-A53 và Cortex-R5, đảm nhận vai trò chính trong việc thực hiện các tác vụ xử lý.
- Giao diện bộ nhớ (Memory Interfaces - MI): Quản lý việc giao tiếp với bộ nhớ trên chip và ngoài chip.
- Thiết bị ngoại vi (I/O Peripherals - IOP):** Bao gồm các thiết bị ngoại vi để giao tiếp và tương tác với môi trường bên ngoài.
- Khối kết nối: Quản lý các kết nối và giao diện giữa các thành phần trong hệ thống.
- Logic lập trình (FPGA Programmable Logic - PL) [7] [8]:
 - Khối logic có thể tái cấu hình (Configurable Logic Blocks - CLB):** Được sử dụng để tạo và thiết kế các mạch logic linh hoạt.
 - Khối RAM:Cung cấp bộ nhớ lưu trữ linh hoạt cho các ứng dụng và mạch logic.
 - Khối xử lý tín hiệu số (DSP48E1):Thực hiện các phép toán xử lý tín hiệu số.
 - Khối chuyển đổi từ analog sang số (Analog-to-Digital Converter - XADC): Thực hiện chức năng chuyển đổi tín hiệu từ dạng analog sang dạng số.
 - Bộ nhớ DDR4 với hiệu suất cao, hỗ trợ 16 kênh DMA, giúp tối ưu hóa truyền dữ liệu giữa các thành phần hệ thống.
 - Giao diện ngoại vi băng thông cao bao gồm 1G Ethernet, USB 3.0, DisplayPort, PCIe Gen2, và các giao diện khác, tạo điều kiện thuận lợi cho kết nối và tương tác với các thiết bị khác.

2.5.2. Đơn vị xử lý (PS) của Zynq Ultrascale+ MPSoC - ZCU102

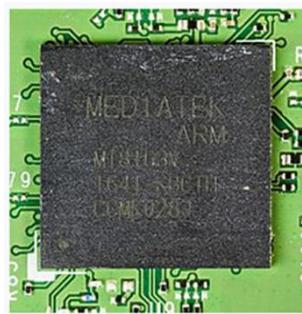
Hình 2.10 dưới đây cung cấp một cái nhìn tổng quát và đơn giản hóa về kiến trúc Zynq MPSoC [11]. Các thành phần như APU, RPU, GPU, PL và bộ điều khiển bộ nhớ ngoài có thể được nhận diện nhanh chóng. Các đơn vị xử lý này giao tiếp với nhau thông qua một kết nối được gọi là "Kết nối đồng bộ bộ nhớ Cache" (Cache Coherent Interconnect - CCI).



Hình 2.10. Kiến trúc của ZCU102 [7]

CCI là một bộ sung vào kiến trúc PS, cho phép các bộ xử lý chia sẻ tác vụ và dữ liệu một cách linh hoạt. Về cơ bản, CCI được sử dụng để đạt được quá trình xử lý bất đối xứng, tạo sự kết nối, hỗ trợ tính mạch lạc giữa các tác vụ xử lý, và đảm bảo rằng mỗi lõi xử lý đang hoạt động trên dữ liệu mới nhất trong toàn bộ PS.

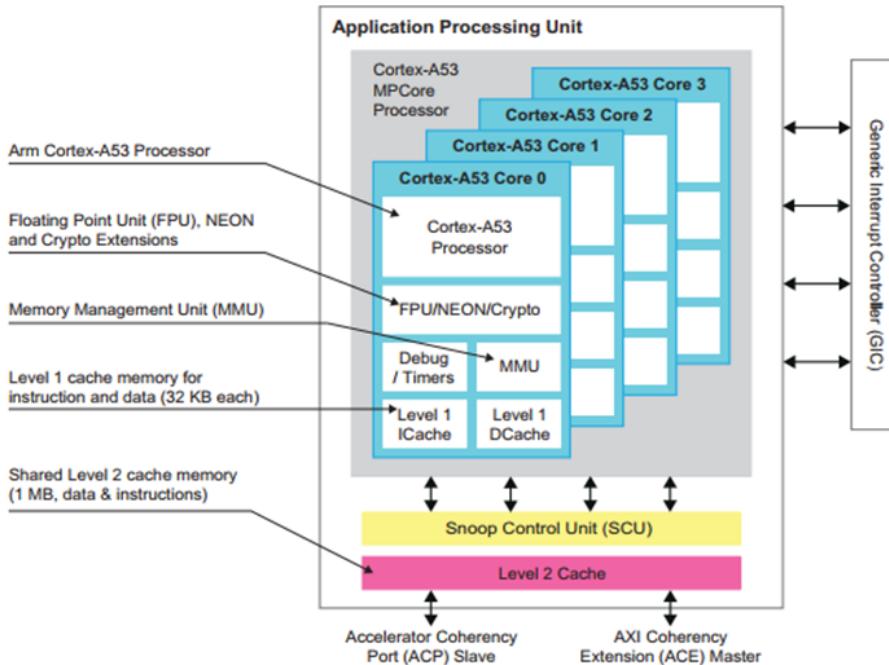
2.5.2.1. Application Processing Unit (APU)



Hình 2.11. ARM Cortex-A53

ARM là viết tắt của "Acorn RISC Machine" hoặc "Advanced RISC Machine," tùy theo giai đoạn lịch sử của công ty. ARM không trực tiếp sản xuất vi xử lý hay chip, mà thay vào đó, tập trung vào việc thiết kế các lõi vi xử lý (CPU

cores) và cấp phép công nghệ này cho các nhà sản xuất chip khác. Các công ty như Qualcomm, Samsung và MediaTek là những ví dụ điển hình về các nhà sản xuất chip sử dụng lõi vi xử lý ARM trong sản phẩm của họ.



Hình 2.12. Sơ đồ khái niệm APU trong ZCU102 [7]

Mỗi lõi Cortex-A53 bao gồm các thành phần tính toán sau: Đơn vị Dầu chấm động (Floating Point Unit - FPU), Động cơ xử lý đa phương tiện (NEON Media Processing Engine - MPE), Tiện ích mở rộng mã hóa (Cryptography Extension - Crypto), Đơn vị Quản lý Bộ nhớ (Memory Management Unit - MMU), và bộ nhớ cache cấp 1 riêng biệt cho mỗi lõi (bao gồm hai đơn vị riêng biệt cho lệnh và dữ liệu). Phần còn lại của APU bao gồm Đơn vị Kiểm soát Snoop (Snoop Control Unit - SCU) và bộ nhớ cache cấp 2 [11].

Các thông số chi tiết của APU được trình bày trong bảng 2.1 dưới đây:

Bảng 2.1. Thông số kỹ thuật của ARM Cortex A53 trong ZCU102

Thông số	Thông số kỹ thuật
----------	-------------------

Chương 2. Cơ sở lý thuyết

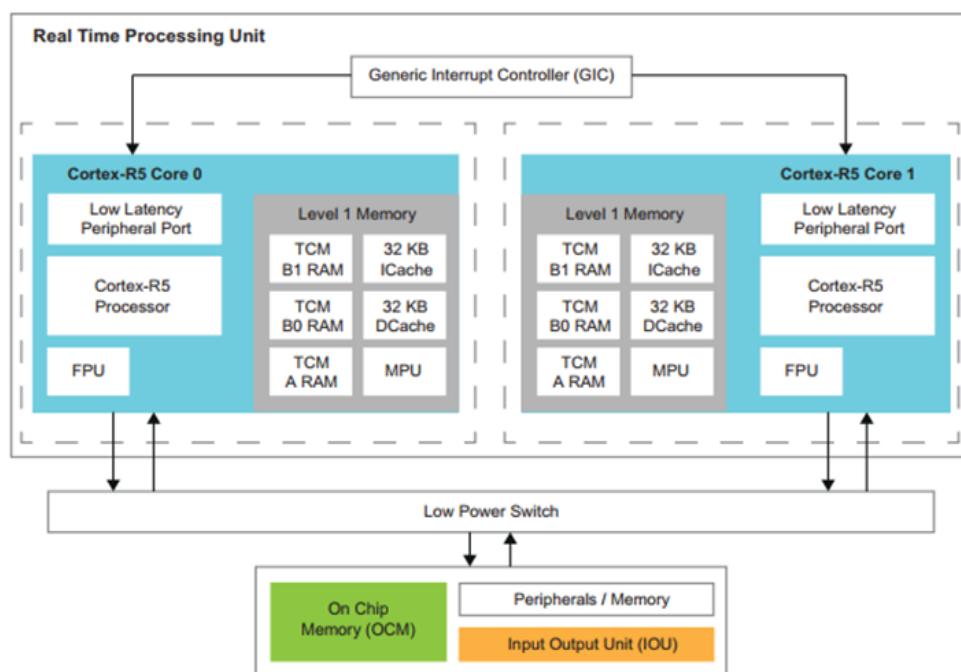
Vị xử lý	ARM Cortex-A53 Quad-Core
Tốc độ xử lý (tối đa)	1.5 GHz
Kiến trúc	ARMv8-A 64-bit
Bộ nhớ đệm L1	- L1i (Instruction Cache): 32 KB
	- L1d (Data Cache): 32 KB
Bộ nhớ đệm L2	512 KB
Bộ nhớ trong (RAM)	4 GB DDR4 (đối với ví dụ)
Bộ nhớ ngoại vi (External)	Khe cắm kết nối SD, USB, PCIe, HDMI, ...
Bộ xử lý đồ họa (GPU)	Arm Mali-400 MP2
Video Codec Unit (VCU)	H.264, H.265
Kết nối tốc độ cao (PS)	PCIe Gen2, USB3.0, SATA 3.1, DisplayPort Gigabit Ethernet
Kết nối không dây	Wi-Fi và Bluetooth (tùy chọn)

Hệ điều hành hỗ trợ

Linux, Android, và các hệ điều hành khác

2.5.2.2. Real-Time Processing Unit (RPU)

RPU chứa một bộ vi xử lý kép Arm Cortex-R5, được thiết kế đặc biệt cho các ứng dụng thời gian thực. Kiến trúc của RPU mang lại hoạt động có độ trễ thấp và hiệu suất ổn định trong toàn bộ đơn vị. Hình 2.16 dưới đây minh họa một sơ đồ khái niệm đơn giản về RPU. [7]



Hình 2.13. Sơ đồ khái niệm của RPU trong ZCU102 [7]

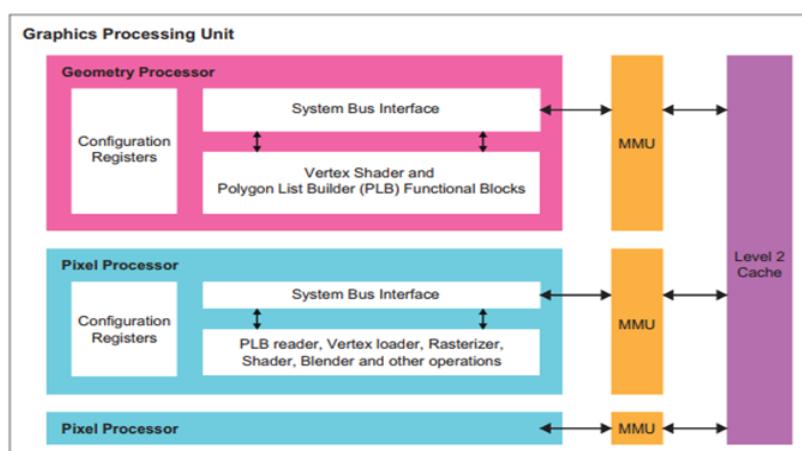
Lõi Cortex-R5 bao gồm một Đơn vị Dấu chấm động (FPU) cho phép thực hiện các tính toán dấu chấm động chính xác ở cả đơn và đôi. Bộ nhớ cấp 1 của mỗi lõi Cortex-R5 có ba Bộ nhớ Liên kết Chặt chẽ (TCMs), hai bộ nhớ cache 32 KB dành cho dữ liệu và lệnh, cùng với Đơn vị Bảo vệ Bộ nhớ (MPU).

Các lõi Cortex-R5 trong RPU có khả năng hoạt động với tốc độ lên đến 600 MHz. Hình 2.16 minh họa các kết nối giữa các lõi xử lý Arm Cortex-R5 và công tắc

tiết kiệm năng lượng, cùng với kết nối tới các thiết bị ngoại vi khác như Đơn vị Đầu vào/Đầu ra (IOU) và Bộ nhớ Trong Chip (OCM).

2.5.2.3. Graphics Processing Unit (GPU)

Đơn vị xử lý đồ họa Arm Mali-400 MP2 (GPU) bao gồm một bộ xử lý hình học (Geometry Processor - GP) và hai bộ xử lý pixel (Pixel Processors - PP). Đơn vị này sử dụng ba Đơn vị Quản lý Bộ nhớ (Memory Management Units - MMU) tích hợp (một cho mỗi bộ xử lý) và một bộ nhớ cache cấp 2 để lưu trữ dữ liệu tạm thời. [7]



Hình 2.14. Sơ đồ khái niệm của GPU trong ZCU102 [7]

GPU có thể đạt được hiệu suất tăng tốc đồ họa hai chiều (2D) và ba chiều (3D) lên đến 667 MHz. Các Giao diện Lập trình Ứng dụng (Application Programming Interfaces - API) OpenGL ES 1.1/2.0 và OpenVG 1.0/1.1 cho phép xử lý đồ họa phức tạp được chuyển giao cho GPU. Hình 2.17 dưới đây cung cấp một sơ đồ khái niệm đơn giản về GPU, minh họa bộ xử lý hình học và pixel cùng với các đường truyền thông của chúng [8].

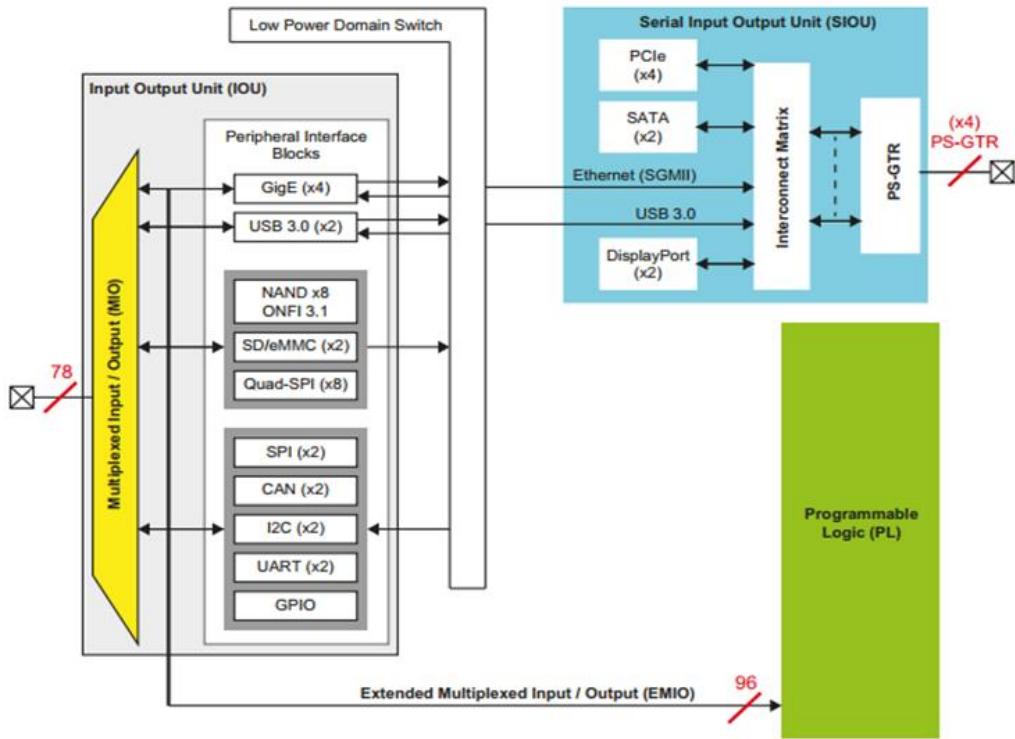
Bộ nhớ cache cấp 2 có dung lượng 64 KB có thể được địa chỉ và bao gồm một giao diện slave của Bus Ngoại vi Tiên tiến (Advanced Peripheral Bus - APB). APB cho phép các thiết bị khác, hoạt động như master, điều khiển bộ nhớ cache cấp 2.

2.5.2.4. Giao tiếp giữa PS với các thành phần ngoại vi trên ZCU102

Hệ thống xử lý (PS) của Zynq MPSoC (ZCU102) có nhiều giao diện để kết nối với các thiết bị ngoại vi. Kết nối ngoại vi sử dụng đầu vào/ra đa chức năng (Multiplexed Input/Output - MIO), cung cấp một giao diện linh hoạt và có thể cấu hình để giao tiếp giữa các chân và các giao diện ngoại vi. Có 78 chân I/O của hệ thống xử lý được kết nối đến các thiết bị ngoại vi bên ngoài thông qua MIO. [8]

Kết nối nhanh với các thiết bị ngoại vi yêu cầu giao tiếp multi-gigabit sử dụng Đơn vị Đầu vào/Đầu ra Nối tiếp (Serial Input/Output Unit - SIOU). SIOU là một khối giao diện nối tiếp nhanh hỗ trợ các giao thức giao diện như PCIe, USB 3.0, DisplayPort, SATA và Ethernet. Có thể kết nối tối đa bốn giao diện nối tiếp nhanh I/O với SIOU tại một thời điểm. Các kết nối cụ thể này cũng có thể được truy cập từ PL thông qua MIO Mở rộng (Extended MIO - EMIO). EMIO tạo ra một đường truyền thông trực tiếp từ các giao diện ngoại vi trong PS đến tài nguyên PL.

Hình 2.13 minh họa kết nối của MIO, EMIO và SIOU trong PS của Zynq MPSoC (ZCU102). Các khối giao diện ngoại vi nằm cạnh MIO, trong khi các khối ngoại vi tốc độ cao được chứa bên trong SIOU. [7]



Hình 2.15. Sơ đồ kết nối của MIO, EMIO và SIOU của ZYNQ MPSoC [7]

❖ Multiplexed Input/Output (MIO)

MIO có khả năng định tuyến nhiều giao diện ngoại vi khác nhau (được chia trong các khối giao diện ngoại vi trong Hình 2.15) đến một tập hợp gồm 78 chân. Bảng 2 dưới đây cung cấp một cái nhìn tổng quan về các giao diện ngoại vi mà có thể được truy cập thông qua MIO. MIO chia sẻ các khối giao diện ngoại vi USB3.0 và Ethernet với SIOU.

Bảng 2.2. Danh sách các giao diện ngoại vi I/O kết nối với MIO [8]

(I/O Interface)	Mô tả
UART (x2)	Universal Asynchronous Receiver Transmitter

Chương 2. Cơ sở lý thuyết

	<p>Là một giao界面 modem dữ liệu tốc độ thấp cho truyền thông nối tiếp không đồng bộ. Thường được sử dụng để kết nối với máy tính hoặc các thiết bị khác thông qua cổng nối tiếp.</p>
SPI (x2)	<p>Serial Peripheral Interface</p> <p>Là một tiêu chuẩn truyền thông nối tiếp dựa trên giao diện 4 chân. Nó có thể được sử dụng ở chế độ master hoặc slave, cho phép truyền và nhận dữ liệu giữa các thiết bị trong hệ thống nhúng.</p>
CAN (x2)	<p>Controller Area Network</p> <p>Là một bộ điều khiển giao diện Bus theo tiêu chuẩn ISO 11898-1, bao gồm cả CAN 2.0A và 2.0B. CAN thường được sử dụng trong các ứng dụng ô tô, công nghiệp và nhúng để truyền thông giữa các thiết bị trong một mạng điều khiển phân tán.</p>
I2C (x2)	<p>Inter-Integrated Circuit Bus</p> <p>Tuân theo đặc tả bus I2C, phiên bản 2. Hỗ trợ cả hai chế độ master và slave.</p>
GPIO	<p>General Purpose Input/Output</p> <p>Là các chân đầu vào/ra được sử dụng để giao tiếp với các thiết bị ngoại vi ngoài của một hệ thống. Trong ZCU102, có tổng cộng 3 bank GPIO, mỗi bank có 26 chân (bits). Các GPIO thường được sử dụng để điều khiển các thiết bị ngoại vi như</p>

Chương 2. Cơ sở lý thuyết

	LED, cảm biến, bộ nhớ, hoặc để đọc trạng thái của các nút nhấn, công tắc và cảm biến.
GigE (x4)	<p>Ethernet (Reduced Gigabit Media Independent Interface - RGMII)</p> <p>Là một chuẩn giao diện Ethernet được sử dụng để kết nối truyền dẫn dữ liệu giữa một MAC (Media Access Controller) và một PHY (Physical Layer) trên một hệ thống tích hợp. Nó hỗ trợ các chế độ truyền dẫn dữ liệu với tốc độ 10Mbps, 100Mbps và 1Gbps, cho phép truyền dữ liệu với các tốc độ khác nhau tùy thuộc vào yêu cầu của ứng dụng.</p>
NAND x8 and ONFI 3.1	<p>NAND Flash Memory Interface</p> <p>Là một giao diện được thiết kế để kết nối với bộ nhớ flash NAND trong các hệ thống nhúng. Nó cung cấp khả năng truy cập và quản lý dữ liệu trên bộ nhớ flash NAND. Trong ZCU102, giao diện NAND Flash Memory hỗ trợ cả giao diện Open NAND Flash Interface (ONFI) phiên bản 3.1, một tiêu chuẩn giao diện được sử dụng để tương tác với bộ nhớ flash NAND một cách hiệu quả và tiện lợi.</p>
USB 3.0 (x2)	<p>Universal Serial Bus</p> <p>Universal Serial Bus (USB) là một chuẩn giao diện kết nối được sử dụng rộng rãi trong các thiết bị điện tử để truyền dữ liệu và cung cấp nguồn điện. Trong trường hợp này, giao diện USB tuân theo chuẩn USB 3.0, cung cấp tốc độ truyền dữ liệu nhanh</p>

	<p>hơn và hiệu suất cao hơn so với các phiên bản trước đó của USB.</p> <p>Ngoài ra, giao diện USB có thể hoạt động như một máy chủ (host), thiết bị (device) hoặc linh hoạt ("on-the-go" hoặc OTG), có nghĩa là nó có khả năng chuyển đổi giữa các chế độ máy chủ và thiết bị tùy thuộc vào yêu cầu của ứng dụng cụ thể. Điều này cho phép giao diện USB thích ứng linh hoạt với các tình huống sử dụng khác nhau, như kết nối các thiết bị lưu trữ với máy tính hoặc điều khiển các thiết bị ngoại vi khác.</p>
SD/eMMC (x2)	<p>Secure Digital and embedded MultiMediaCard Memory Interface</p> <p>Là các giao diện được sử dụng để kết nối và quản lý dữ liệu trên các loại bộ nhớ nhúng như thẻ nhớ SD và eMMC. Trong ZCU102, giao diện này hỗ trợ chuẩn SD 3.0, cung cấp tốc độ truyền dữ liệu nhanh và hiệu suất cao hơn so với các phiên bản trước đó của SD. Ngoài ra, nó cũng hỗ trợ chuẩn eMMC 4.51, cho phép truy cập và quản lý dữ liệu trên bộ nhớ eMMC một cách hiệu quả và tiện lợi. Điều này cho phép tích hợp dễ dàng của các thiết bị lưu trữ này vào các hệ thống nhúng và điều khiển.</p>
Quad-SPI x8	Quad Serial Peripheral Interface

Là một giao diện truyền thông nối tiếp dựa trên chuẩn SPI (Serial Peripheral Interface) nhưng sử dụng giao diện quad I/O, có nghĩa là nó sử dụng một bus dữ liệu bốn bit thay vì chỉ một bit như trong giao diện SPI tiêu chuẩn. Quad SPI có khả năng truyền dữ liệu với tốc độ cao hơn so với giao diện SPI tiêu chuẩn, vì nó có thể truyền và nhận nhiều bit dữ liệu cùng một lúc. Giúp làm tăng thông lượng dữ liệu và cải thiện hiệu suất truyền thông của giao diện. Quad SPI thường được sử dụng trong các ứng dụng yêu cầu truyền dữ liệu nhanh và hiệu suất cao như trong lưu trữ flash và các thiết bị nhúng khác.

❖ Serial Input Output Unit (SIOU)

SIOU bao gồm bốn bộ điều khiển ngoại vi tốc độ cao, mỗi bộ sử dụng một cặp kênh truyền và nhận đa-gigabit, được gọi là truyền nhận PS-GTR. PS-GTR có khả năng hỗ trợ tốc độ dữ liệu lên đến 6.0 Gb/s và có thể giao tiếp với bất kỳ khối ngoại vi tốc độ cao nào sử dụng ma trận liên kết được thể hiện trong Hình 2.12 ở trang 16. Bảng 4 dưới đây cung cấp danh sách các thiết bị tốc độ cao được hỗ trợ bởi truyền nhận PS-GTR.

Bảng 2.3. Danh sách các giao diện ngoại vi I/O tốc độ cao kết nối với SIOU [7]

(I/O Interface)	Mô tả
DisplayPort	DisplayPort Interface Là một giao diện hiển thị số chủ yếu được sử dụng để truyền video, nhưng cũng có khả năng truyền dữ liệu âm thanh, USB

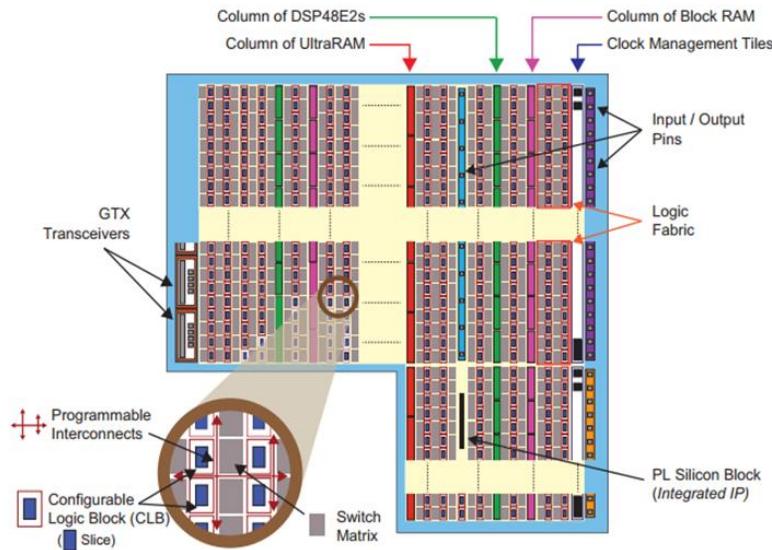
Chương 2. Cơ sở lý thuyết

	và các dạng khác của dữ liệu. Tối đa hai kênh có thể giao tiếp với các truyền nhận PS-GTR, cho phép truyền dẫn dữ liệu với tốc độ cao và hiệu suất ổn định trong các ứng dụng hiển thị số.
USB 3.0 (x2)	<p>Universal Serial Bus</p> <p>Tuân thủ các yêu cầu và quy định kỹ thuật của USB 3.0, cho phép các thiết bị kết nối một cách nhanh chóng và đáng tin cậy. Một đặc điểm quan trọng của USB 3.0 là khả năng vận hành ở chế độ "on-the-go" (OTG). Điều này có nghĩa là thiết bị có thể chuyển đổi linh hoạt giữa các vai trò máy chủ và thiết bị, tùy thuộc vào yêu cầu của ứng dụng. Sự linh hoạt này giúp tăng cường khả năng kết nối và tương tác giữa các thiết bị khác nhau.</p>
SATA (x2)	<p>Serial Advanced Technology Attachment</p> <p>SATA sử dụng dụng các cặp dây truyền chênh lệch cho truyền thông nối tiếp chủ yếu với các thiết bị lưu trữ ở tốc độ 1.5, 3.0 và 6.0 Gb/giây tùy thuộc vào thế hệ SATA (1, 2 hoặc 3 tương ứng). SATA thường được sử dụng để kết nối ổ cứng trong các máy tính và thiết bị lưu trữ khác, cung cấp khả năng truy cập và lưu trữ dữ liệu với tốc độ và hiệu suất cao.</p>
PCIe	<p>Peripheral Component Interconnect Express</p> <p>Là một chuẩn giao tiếp mở rộng máy tính nhanh chóng và truyền thông nối tiếp. Được sử dụng để kết nối các card mở</p>

	<p>rộng như card đồ họa, card mạng, card âm thanh và các thiết bị ngoại vi khác với ZCU102 mạch chủ.</p> <p>PCIe cung cấp tốc độ truyền dẫn dữ liệu cao và hiệu suất mạnh mẽ, cho phép truyền tải dữ liệu giữa các thành phần của hệ thống với tốc độ rất nhanh.</p>
GigE (x4)	<p>Ethernet (SGMII)</p> <p>Giao diện MAC Ethernet hỗ trợ các chế độ tốc độ truyền dẫn là 10 Mbps, 100 Mbps và 1 Gbps. SIOU (Serial Input Output Unit) chỉ hỗ trợ Giao diện Ngoại vi Độc lập trung bình Gigabit (SGMII) đến khối giao diện ngoại vi GigE.</p>

2.5.3. Logic lập trình (FPGA Programmable Logic - PL)

PL là một thành phần quan trọng của Zynq MPSoC, vì nó cung cấp tăng tốc phần cứng cho các phép toán số tính toán cường độ cao. PL sử dụng kiến trúc FPGA (Field-Programmable Gate Array) Kintex UltraScale+ 16nm, chứa đựng tất cả các tính năng và ưu điểm đã được thảo luận trong suốt phần này.

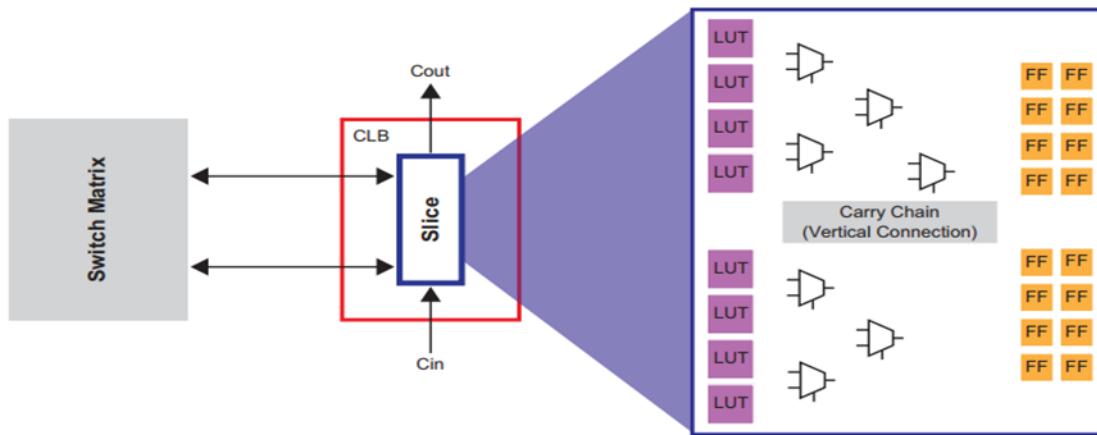


Hình 2.16. Zynq MPSoC Programmable Logic [7]

2.5.3.1. Cấu trúc Logic của Programmable Logic

PL chủ yếu bao gồm cấu trúc FPGA Kintex UltraScale+ 16nm, các Slices và Configurable Logic Blocks (CLBs). Sắp xếp của CLBs trong cấu trúc FPGA là các mảng hai chiều. Mỗi CLB được đặt gần một ma trận chuyển mạch để có thể định tuyến dữ liệu đến một tài nguyên PL khác. Ngoài ra, CLBs có thể kết nối với các tài nguyên tương tự khác bằng cách sử dụng liên kết khác có thể lập trình được.

Một CLB bao gồm một slice, chứa đựng các tài nguyên cần thiết để triển khai mạch logic tuần tự và logic tổ hợp. Mỗi phần slice trong các thiết bị Zynq MPSoC được cấu thành từ 8 x 6-input Lookup Tables (LUTs), 16 Flip-Flops (FFs) và logic định tuyến bổ sung. Các slice đối diện theo chiều dọc có thể được kết nối với nhau để triển khai các mạch toán lớn thông qua việc sử dụng logic carry (được ghi chú là Cin và Cout trong Hình 2.17). Bộ mạch này bao gồm các bộ chuyển mạch và kết nối chuỗi logic được sử dụng để định tuyến tín hiệu trung gian giữa các slice liền kề.



Hình 2.17. Kết nối và các thành phần bên trong của CLB [7] [8]

Cấu trúc logic FPGA có thể được sử dụng để xây dựng các mạch toán học khác nhau như cộng, nhân và chia. Các mạch này sau đó có thể được kết hợp để thực hiện các hàm toán học tùy ý.

Lookup Tables (LUTs) có thể được sử dụng như bộ nhớ cục bộ nhỏ, hoặc được kết nối với nhau để tạo thành một mảng bộ nhớ lớn hơn. Cấu trúc này được biết đến là RAM phân tán (Distributed RAM), và do hiệu quả về tài nguyên và tính linh hoạt của nó trong việc đặt, nó làm cho nó hiệu quả khi một mạch logic yêu cầu bộ nhớ.

2.5.3.2. Lưu trữ và xử lý tín hiệu

Ngoài cấu trúc logic, có ba nguồn tài nguyên chính khác để lưu trữ và xử lý dữ liệu trong thành phần PL của ZCU102. Đó là Block RAMs, UltraRAMs và các đơn vị xử lý tín hiệu DSP48E2. [7]

❖ Block RAMs

Block RAMs trong ZCU102 có thể được cấu hình để hoạt động như Bộ Nhớ Truy Cập Ngẫu Nhiên (RAM), Bộ Nhớ chỉ đọc (ROM) và Bộ đệm First In First Out (FIFO), đồng thời hỗ trợ Mã Hóa Sửa Lỗi (ECC). Mỗi Block RAM trong PL có giao diện hai cổng (mỗi cổng có khả năng truy cập không gian bộ nhớ và dữ liệu

được lưu trữ trong khi hoạt động dưới cùng một hoặc các miền đồng hồ khác nhau). Mỗi Block RAM có thể lưu trữ lên đến 36 KB thông tin và có thể được cấu hình như một RAM 36 KB hoặc hai RAM 18 KB riêng lẻ.

❖ Ultra RAM

Một UltraRAM đơn có khả năng lưu trữ lên đến 288Kb. UltraRAM chỉ có sẵn trong một số thiết bị Zynq MPSoC được lựa chọn và đại diện cho một lựa chọn thay thế cho bộ nhớ ngoại vi (như SRAM) trong các thiết kế phần cứng. UltraRAM có giao diện đồng bộ, hai cổng và có thể được kết hợp để tạo ra các mảng bộ nhớ lớn lên đến 100 Mb. Điều này là khả dụng mà không cần sử dụng thêm cấu trúc logic, vì UltraRAM có phần cứng định tuyến riêng biệt, cho phép chúng hoạt động ở tần số đồng hồ cao.

❖ DSP48E2

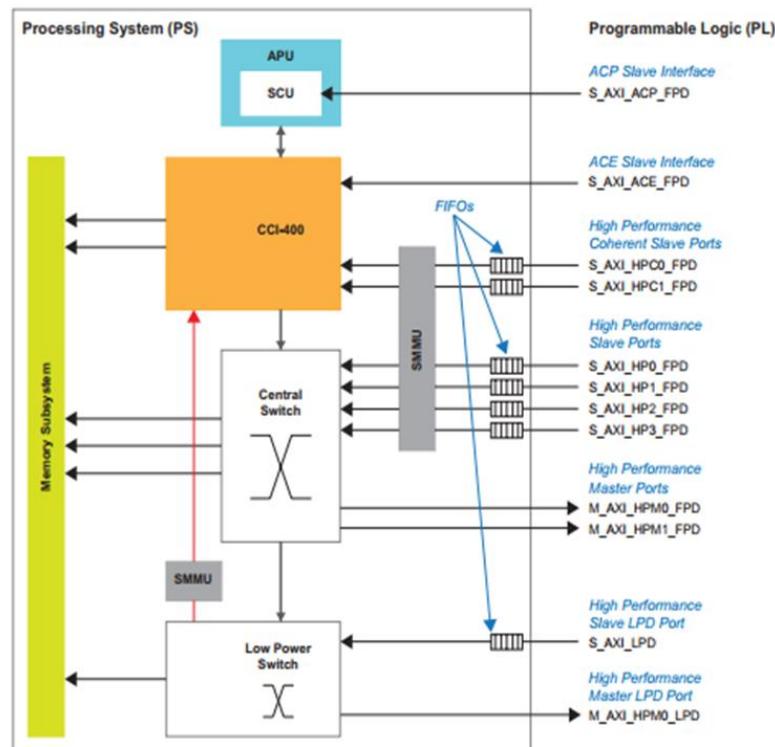
Các đơn vị DSP48E2 trong PL của tất cả các thiết bị Zynq MPSoC là các phép nhân phần cứng cung cấp xử lý tín hiệu hiệu quả và có tốc độ cao cho các ứng dụng xử lý tín hiệu. Các đơn vị này được sắp xếp thành các cột khắp cấu trúc logic FPGA, như được thể hiện trong Hình 2.3.6a ở trang 21, để giảm độ trễ định tuyến đến các mạch logic lân cận. Ngoài ra, mỗi đơn vị được căn chỉnh theo chiều ngang với một Block RAM, cung cấp kết nối hiệu quả giữa hai loại tài nguyên này. Có thể kết hợp nhiều đơn vị DSP48E2 nếu một mạch toán yêu cầu chiều dài từ vựng lớn hơn so với một đơn vị DSP48E2 đơn có thể cung cấp.

2.5.4. Giao tiếp giữa hệ thống xử lý (PS) và Logic lập trình (PL)

Chuẩn giao diện chính được sử dụng trong PL để truyền dữ liệu giữa các khối IP là AXI. Có thể kết nối nhiều cổng AXI trong PL từ các nguồn khác nhau bằng cách sử dụng một AXI interconnect. AXI interconnect hoạt động như một công tắc trong cấu trúc logic, để PL có thể định tuyến lưu lượng từ nhiều nguồn đến các đích đến của chúng. PS cũng chứa nhiều liên kết, một số trong đó thuộc chuẩn

AXI, trong khi các liên kết khác thuộc một dạng chuẩn AMBA khác, phụ thuộc vào giao diện.

Khi giao tiếp giữa PL và PS, AXI hỗ trợ chuyển dữ liệu với công suất truyền tải cao và độ trễ thấp. Chuẩn AXI thực hiện các cổng dành riêng giữa PS và PL. Hình 2.3.7 minh họa hầu hết các kết nối này, chỉ hiển thị các tài nguyên trong PS kết nối trực tiếp với PL. PS bao gồm CCI-400, APU (và SCU), đơn vị quản lý bộ nhớ hệ thống (SMMU), công tắc trung tâm và công tắc tiết kiệm điện. Hệ thống bộ nhớ cũng đã được bao gồm để minh họa các tài nguyên bộ nhớ khác nhau trong PS mà mỗi công tắc và liên kết có thể truy cập. SMMU cung cấp chuyển đổi địa chỉ giữa PS và PL, để PL có thể sử dụng địa chỉ ảo.



Hình 2.18. Sơ đồ khái kết nối giữa PS và PL

Có bốn giao diện mà PL có thể sử dụng để duy trì tính nhất quán với PS, trong đó có ba giao diện được kết nối trực tiếp:

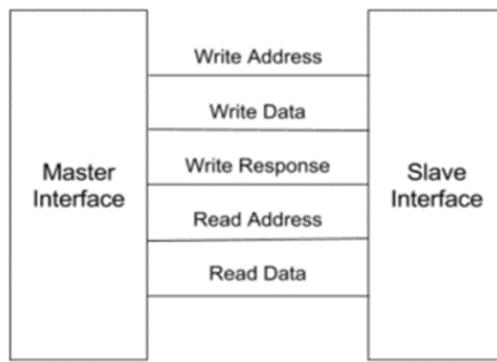
- AXI Coherency Extension (ACE): ACE có thể truy cập bộ nhớ hệ thống và cũng bộ nhớ cục bộ của APU, thông qua CCI; giúp chia sẻ thông tin cập nhật.
- High-Performance Coherent Ports: Các cổng hiệu suất cao này được kết nối trực tiếp với CCI để cho phép truyền thông dữ liệu giữa PL và PS. Cả hai cổng đều bao gồm bộ đệm FIFO để cho phép truyền chuyển và truyền thông tốc độ cao giữa PS và các thành phần trong PL.
- Accelerator Coherency Port (ACP): ACP sử dụng một phần nhỏ của giao thức ACE-Lite và cung cấp một kết nối không đồng bộ duy nhất giữa PL và SCU, trong APU.

2.5.5. Tổng quan về chuẩn giao tiếp phần cứng AMBA AXI [9]

Giao thức AXI, một phần của hệ thống giao tiếp AMBA do ARM phát triển, là một giao thức BUS phổ biến trong vi điều khiển ARM. Đặc điểm nổi bật của AXI là khả năng linh hoạt và mở rộng, đặc biệt là trong việc xử lý dữ liệu đợt (burst), giúp cải thiện hiệu suất của các hệ thống có yêu cầu về tần số cao và băng thông lớn. Thiết kế của AXI tập trung vào việc đạt được hiệu suất cao và độ trễ thấp, đồng thời đảm bảo tương thích với các giao thức khác như AHB và APB trong họ AMBA. AXI xác định chuẩn giao tiếp trong ba loại chính: MASTER và Interconnect bus, SLAVE và Interconnect bus, MASTER kết nối trực tiếp với SLAVE.

2.5.5.1. Cấu trúc

Giao thức AXI thiết lập một cơ chế và quy trình xử lý cho hai hành động chính: đọc và ghi. Cả hai hoạt động này được tổng hợp dưới khái niệm chung là một "truy cập" (access), trong đó mỗi truy cập được coi là một "giao dịch" (transaction).



Hình 2.19. Sơ đồ khái niệm các kênh giao tiếp của chuẩn AXI

AXI hoạt động dựa trên năm loại kênh độc lập, mỗi kênh có vai trò và nhiệm vụ riêng, và không phụ thuộc vào các kênh khác để hoàn thành công việc. Tuy nhiên, chúng vẫn liên quan đến nhau. Năm kênh này bao gồm:

- Kênh Địa Chỉ Ghi (Write Address Channel): Truyền thông tin địa chỉ và điều khiển liên quan đến quá trình ghi từ MASTER đến SLAVE.
- Kênh Dữ Liệu Ghi (Write Data Channel): Truyền dữ liệu ghi từ MASTER đến SLAVE.
- Kênh Đáp Ứng Ghi (Write Response Channel): Truyền thông tin phản hồi về quá trình ghi từ SLAVE đến MASTER.
- Kênh Địa Chỉ Đọc (Read Address Channel): Truyền thông tin địa chỉ và điều khiển liên quan đến quá trình đọc từ MASTER đến SLAVE.

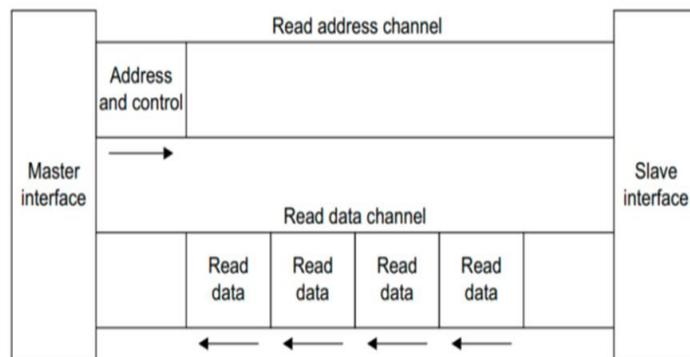
2.5.5.2. Quá trình hoạt động chuẩn giao tiếp AXI

Một hoạt động ghi hoặc đọc trong giao thức này được xem xét là một transaction. Mỗi transaction bao gồm bốn thông tin cơ bản:

- Thông tin địa chỉ: Xác định đích mà transaction sẽ được xử lý.
- Thông tin điều khiển: Xác định thuộc tính của transaction, như loại burst, độ dài burst, kích thước burst và các thuộc tính khác.
- Thông tin dữ liệu: Chứa dữ liệu của transaction.

- Thông tin phản hồi: Xác định trạng thái của transaction, bao gồm việc kiểm tra xem có lỗi hay không.

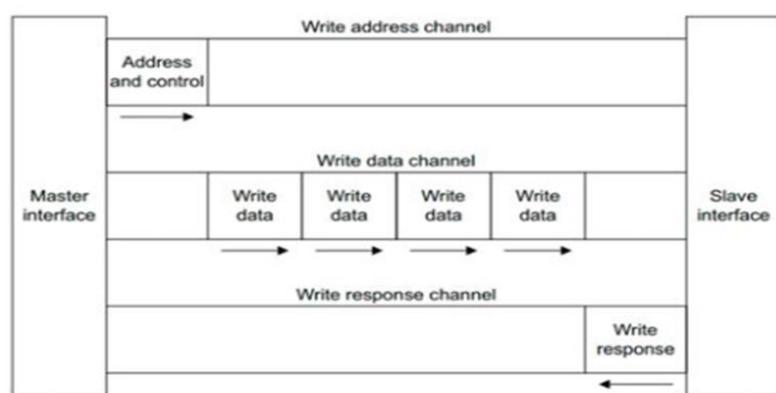
Một transaction đọc được quản lý bởi hai kênh: kênh Địa Chỉ Đọc và kênh Dữ Liệu Đọc, được mô tả chi tiết trong hình 2.18 dưới đây:



Hình 2.20. Sơ đồ khái quát trình đọc dữ liệu

Trong quá trình đọc của AXI, có hai bước xử lý:

- Phía MASTER gửi thông tin địa chỉ và điều khiển để bắt đầu quá trình đọc trên kênh Địa Chỉ Đọc.
- Phía SLAVE phản hồi dữ liệu kèm thông tin phản hồi trên kênh Dữ Liệu Đọc. Số lượng dữ liệu và thông tin phản hồi được xác định bởi thông tin điều khiển được truyền từ phía MASTER qua kênh Địa Chỉ Đọc.



Hình 2.21. Sơ đồ khái quát trình ghi dữ liệu

Quá trình ghi trong AXI gồm ba bước xử lý:

- Phía MASTER gửi thông tin địa chỉ và điều khiển để khởi động quá trình ghi trên kênh Địa Chỉ Ghi.
- Phía MASTER truyền dữ liệu ghi qua kênh Dữ Liệu Ghi. Số lượng dữ liệu ghi được xác định bởi thông tin điều khiển từ phía MASTER qua kênh Địa Chỉ Ghi.
- Phía SLAVE gửi thông tin phản hồi về việc ghi thành công qua kênh Phản Hồi Ghi sau khi quá trình burst ghi đã hoàn thành.

2.6. Tổng quan về Intellectual Property (IP)

2.6.1. Giới thiệu về Intellectual Property (IP)

Các IP diện cho các mô-đun độc lập hoặc khối logic được tích hợp vào FPGA hoặc mạch tích hợp được tùy chỉnh cho ứng dụng ASIC (Mạch tích hợp Đặc biệt cho Ứng dụng). Quá trình phát triển chúng thường sử dụng ngôn ngữ mô tả phần cứng như VHDL, Verilog, System Verilog hoặc ngôn ngữ tổng hợp cấp cao (High-level Synthesis - HLS) như C. Các thành phần IP có khả năng thực hiện nhiều chức năng phức tạp và có thể được tái sử dụng trong quá trình thiết kế mạch tích hợp. Ba loại chính là IP cứng (hard IP), IP mềm (soft IP) và IP bán cứng (firm IP) [10].

Các thành phần IP mềm được phát triển bằng ngôn ngữ mô tả phần cứng như VHDL, Verilog, System Verilog. Ưu điểm của các thành phần IP mềm là khả năng sửa chữa lỗi và linh hoạt trong việc thay đổi nhõ trong ứng dụng.

Các thành phần IP bán cứng được thiết kế ở cấp độ cổng (gate-level netlist), là mã RTL chuyển đổi sang cổng logic sau quá trình tổng hợp. Ưu điểm của các thành phần IP bán cứng là khả năng sửa chữa ở mức độ nhất định và đã hoàn tất thử nghiệm lỗi ở IP mềm.

Các thành phần IP cứng là kết quả của firm IP sau giai đoạn kết nối cổng logic theo trật tự cụ thể. Sau khi được sản xuất, các thành phần cứng trở thành một phần của

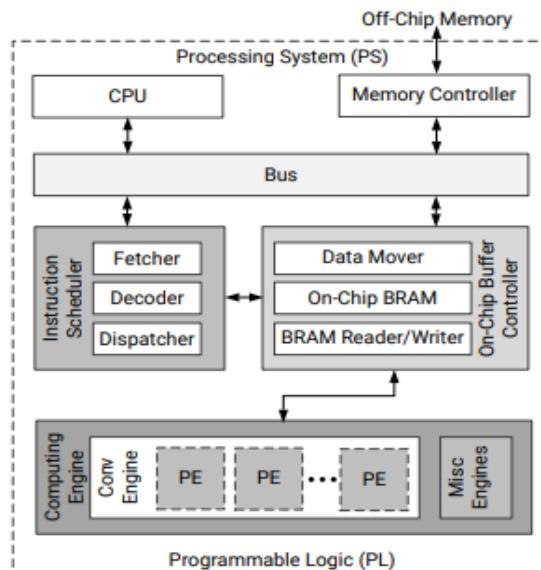
SoC và thực hiện chức năng cụ thể. Vị trí cố định trong FPGA khiến chúng không thể chuyển sang FPGA khác.

2.6.2. Nền tảng phần cứng của Deep Learning Processor Unit (DPU) [10]

Xilinx® DPUCZDX8G là một IP được thiết kế để tối ưu hóa cho các mạng nơ-ron tích chập. Một số ví dụ về các mạng nơ-ron tích chập đã được triển khai bao gồm VGG, ResNet, GoogLeNet, YOLO, SSD, MobileNet và FPN, cùng với nhiều mạng khác.

IP DPUCZDX8G được triển khai trong logic lập trình (PL) của thiết bị Zynq® UltraScale+™ MPSoC – ZCU102 với các kết nối trực tiếp đến hệ thống xử lý (PS). DPUCZDX8G thực thi mã microcode đã được biên dịch từ đồ thị mạng nơ-ron và yêu cầu các vị trí bộ nhớ có thể truy cập để chứa hình ảnh đầu vào cũng như dữ liệu tạm thời và dữ liệu đầu ra..

Sơ đồ khái niệm kiến trúc phần cứng của DPUCZDX8G được hiển thị trong hình 2.22 dưới đây:



Hình 2.22. Sơ đồ khái niệm kiến trúc phần cứng của DPUCZDX8G

Sau khi khởi động, DPUCZDX8G lấy lệnh từ bộ nhớ ngoài (off-chip memory) để điều khiển hoạt động của khối. Các lệnh này được tạo ra bởi trình biên dịch Vitis™ AI.

Bộ nhớ trên chip (On-chip memory) được dùng để đệm các hàm kích hoạt ở ngõ vào, các bản đồ đặc trưng và dữ liệu ở ngõ ra nhằm đạt được hiệu quả cao nhất. Dữ liệu được tái sử dụng càng nhiều thì càng tốt. Vì giúp giảm yêu cầu băng thông của bộ nhớ ngoài. Các phần tử xử lý (PE) được tận dụng tối đa cho các khối xây dựng chi tiết như bộ nhân, bộ cộng và bộ tích lũy trong các thiết bị của Xilinx.

CHƯƠNG 3: THIẾT KẾ HỆ THỐNG

3.1. Yêu cầu hệ thống

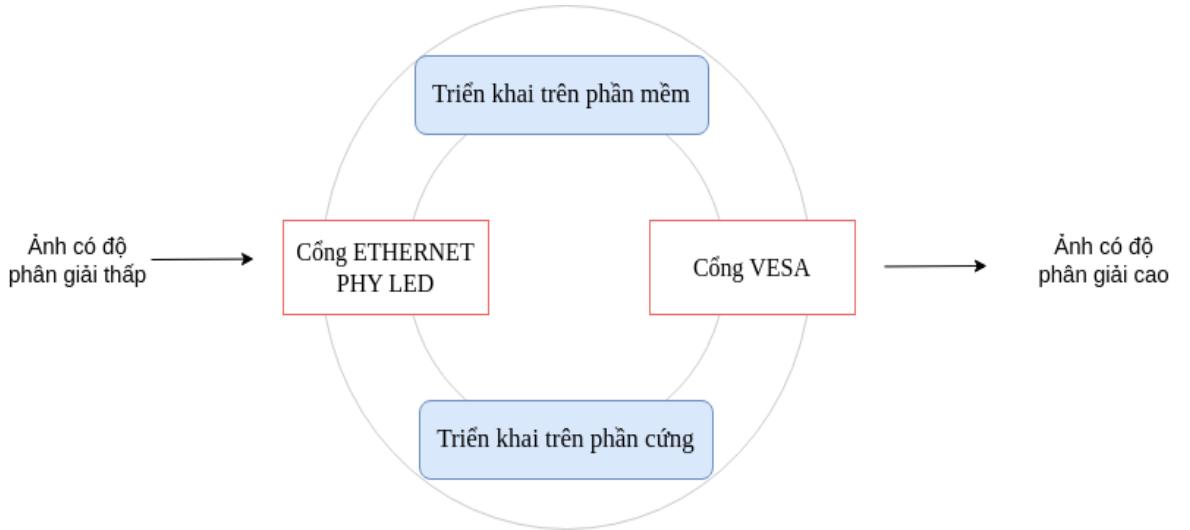
Đề tài yêu cầu thực hiện mô hình mạng siêu phân giải RCAN theo cấu trúc thư viện Pytorch được triển khai trên Kaggle và Zynq Ultrascale+ MPSoC - ZCU102.

- Thiết kế các IP Cores giao tiếp giữa Zynq Ultrascale+ MPSoC với Deep Learning Processor Unit (DPU).
- Xây dựng hệ thống khởi động tích hợp thiết kế IP Cores chạy hệ điều hành Linux cho Xilinx Ultrascale+ MPSoC - ZCU102.
- Tiến hành tiền xử lý thuật toán siêu phân giải ảnh RCAN và triển khai trên Xilinx Ultrascale+ MPSoC - ZCU102.

3.2. Sơ đồ khối và đặc điểm kỹ thuật hệ thống

Hệ thống nâng cao độ phân giải trên Zynq Ultrascale+ MPSoC - ZCU102 tận dụng chức năng của bộ xử lý trong sâu DPU [10] trên ZCU102, chạy mô hình mạng học sâu trên ZCU102 nhằm mục đích cải thiện độ chính xác, và tăng tốc độ xử lý nhanh hơn so với triển khai trên các nền tảng phần mềm khác.

Sơ đồ khối của hệ thống siêu phân giải ảnh được mô tả qua hình 3.1 dưới đây. Trong đó, dữ liệu ngõ vào được đọc qua cổng Ethernet từ máy tính và đưa vào khối xử lý siêu phân giải hình ảnh – đây là bộ xử lý học sâu (DPU) được xây dựng trên cả phần mềm và phần cứng, các module giao tiếp với nhau bằng chuẩn AXI4-Stream và kết quả của nó sẽ được lưu vào bộ nhớ DDR trên phần cứng, có thể hiển thị bằng cổng VESA.



Hình 3.1. Sơ đồ khối hệ thống siêu phân giải ảnh

3.3. Thiết kế hệ thống

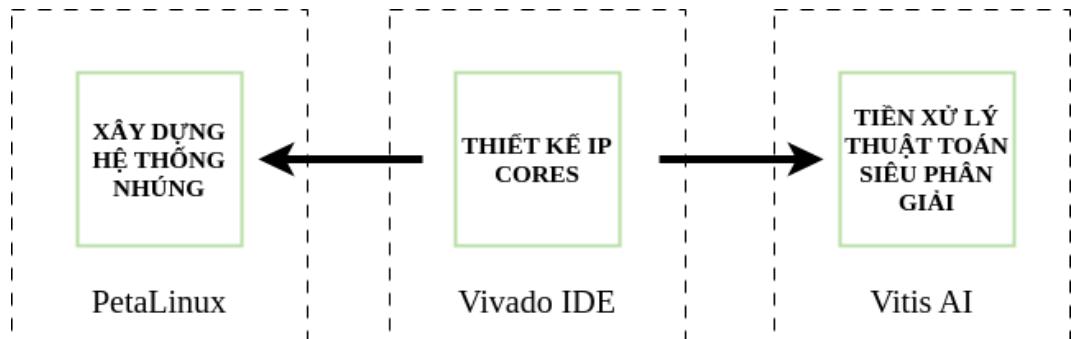
Để xây dựng hệ thống cần phải triển khai trên hai môi trường nền tảng:

Nền tảng phần mềm: Nền tảng này là nơi thực hiện thiết kế IP Cores, xây dựng hệ thống nhúng, và xác định môi trường để triển khai thuật toán siêu phân giải ảnh. Cụ thể sử dụng thư viện Pytorch được tích hợp trên Vitis AI Model Zoo.

Nền tảng phần cứng: Nền tảng này tiến hành chạy môi trường làm việc Linux và triển khai thuật toán siêu phân giải ảnh đã biên dịch được nhúng từ nền tảng phần mềm.

3.3.1. Nền tảng phần mềm

Sơ đồ khối nền tảng phần mềm được mô tả ở hình 3.2 dưới đây:



Hình 3.2. Sơ đồ khối nền tảng phần mềm

Trên nền tảng phần mềm, gồm có ba khối chính:

Thiết kế IP Cores: Khối này thực hiện trên phần mềm Vivado IDE được Xilinx phát triển. Kết nối các IP Cores đã được thiết kế bởi Xilinx như Zynq Ultrascale+ MPSoC, Deep Learning Processor Unit, Processor System Reset,... với nhau và tiến hành cài đặt thông số cần thiết, xác minh, đóng gói, triển khai, tạo bitstreams,... để tạo nền tảng phần cứng cho quá trình xây dựng hệ thống nhúng.

Xây dựng hệ thống nhúng: Khối này thực hiện trên công cụ PetaLinux được Xilinx phát triển. Công cụ PetaLinux có chức năng tự động hóa quá trình xây dựng (build) hệ thống chạy hệ điều hành Linux nhúng dựa trên các cấu hình và mã nguồn được tạo từ khái niệm thiết kế IP Cores. Quá trình này bao gồm việc tạo ra hệ điều hành Linux kernel, các công cụ hệ thống như busybox, thư viện và ứng dụng cần thiết khác, cũng như việc tạo ra hệ thống tập tin rootfs (file system) để chạy trên Zynq Ultrascale+ MPSoC - ZCU102.

Tiền xử lý thuật toán siêu phân giải: Khối này thực hiện trên Kaggle và phần mềm Vitis AI IDE được Xilinx phát triển. Vitis AI được tích hợp các thuật toán mạng học sâu của các thư viện như Tensorflow, Pytorch, Caffe,... trong thư mục Vitis AI Model Zoo. Ngoài ra Vitis AI còn có các công cụ tiền xử lý các thuật toán như tối ưu, lượng tử hóa, biên dịch,... Công cụ Kaggle sẽ huấn luyện mô hình mạng và phần mềm Vitis-AI sẽ xử lý mô hình mạng đã được huấn luyện để triển khai trên ZCU102.

3.3.1.1. Quy trình thiết kế nền tảng phần mềm

Quy trình thiết kế gồm có ba khái niệm chính được mô tả ở hình 3.3 dưới đây:

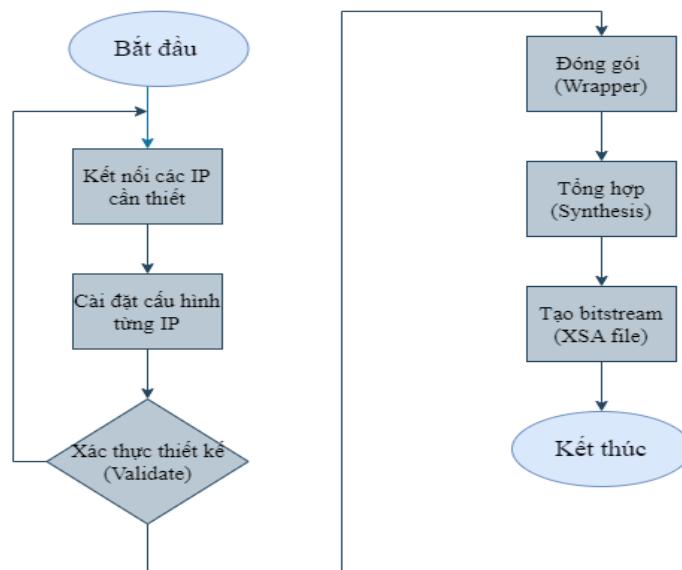
**Hình 3.3. Quy trình thiết kế nền tảng phần mềm**

Nội dung thiết kế nền tảng phần mềm được mô tả từng bước như sau:

❖ Phần mềm Vivado IDE:

Đây là bước đầu tiên. Mục tiêu là tạo ra nền tảng phần cứng được định dạng .xsa.

Lưu đồ thiết kế lõi IP trên Vivado được mô tả trong hình dưới đây:

**Hình 3.4. Lưu đồ thiết kế IP tạo file XSA trên Vivado IDE**

Bước 1: Kết nối các I/O của các IP Cores như Zynq Ultrascale+ MPSOC, Deep Learning Processor Unit,Clock Wi với nhau.

Bước 2: Cài đặt các thông số kết nối như số lõi DPU, chuẩn kết nối, tần số,...

Bước 3: Xác thực thiết kế (Validate Design) là một bước quan trọng. Quá trình này sẽ kiểm tra tính ràng buộc thiết kế như ràng buộc thời gian (timing

constraints), ràng buộc bố trí (placement constraints), ràng buộc kết nối (routing constraints), kiểm tra các khối chức năng được kết nối đúng cách, không có lỗi kết nối (routing errors) và không có lỗi thiết kế logic nào xảy ra. Đảm bảo rằng chúng được thiết lập chính xác và đáp ứng được yêu cầu của FPGA.

Bước 4: Đóng gói thiết kế (Wrapper) sẽ đóng gói các IP thành các module bậc cao để có thể dễ dàng quản lý kết nối giữa các IP Cores và các phần khác của dự án.

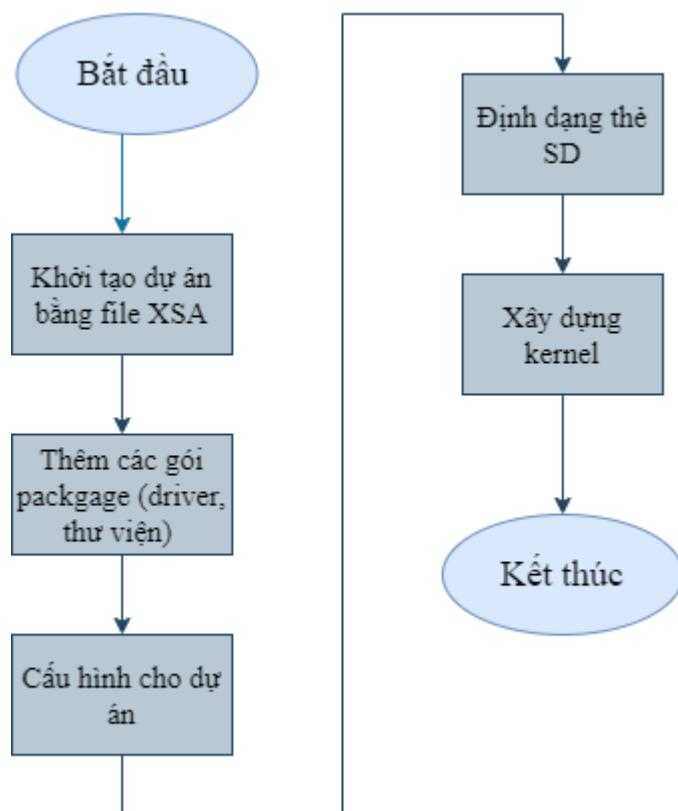
Bước 5: Tổng hợp thiết kế (Synthesis) giúp tối ưu hóa cấu trúc và chức năng của IP để đảm bảo tính tương thích, hiệu suất tốt nhất và sử dụng tài nguyên FPGA một cách hiệu quả nhất.

Bước 6: Bitstream là một tập hợp các dữ liệu được lập trình để cấu hình FPGA và các tài nguyên liên quan, chẳng hạn như bộ nhớ, cấu trúc logic và các kết nối,...

Bước 7: Tệp XSA mô tả toàn bộ hệ thống phần cứng, bao gồm cả cấu hình FPGA, IP cores, bộ nhớ, các kết nối và tài nguyên khác. File XSA có thể dễ dàng tích hợp vào các dự án khác và sử dụng lại trong các môi trường làm việc khác nhau như PetaLinux, Vitis AI,... được Xilinx phát triển.

❖ Công cụ PetaLinux

Lưu đồ thiết kế dự án PetaLinux để xây dựng các kernel chạy hệ điều hành Linux cho ZCU102 được mô tả trong hình dưới đây:



Hình 3.5. Lưu đồ thiết kế dự án Linux trên công cụ PetaLinux

Bước 1: Tạo dự án PetaLinux với tập tin mô tả phần cứng được chọn là file XSA được tạo ra từ bước 7 ở phần mềm Vivado.

Bước 2: Thêm các gói package hỗ trợ người dùng như xrt, dfn, vitisai,... Các gói package này chứa các driver hỗ trợ phần cứng, thư viện, công cụ hỗ trợ phần mềm nhúng,... Danh sách các gói package đầy đủ được liệt kê ở phần phụ lục 2.

Bước 3: Kích hoạt OpenSSH và vô hiệu hóa Dropbear nhằm mục đích cải thiện tốc độ truyền dữ liệu nhanh hơn trong nền tảng nhúng.

Bước 4: Vô hiệu hóa CPU IDLE trong cấu hình của Kernel giúp tránh trường hợp hệ thống bị treo do các giao tiếp AXI chưa hoàn tất.

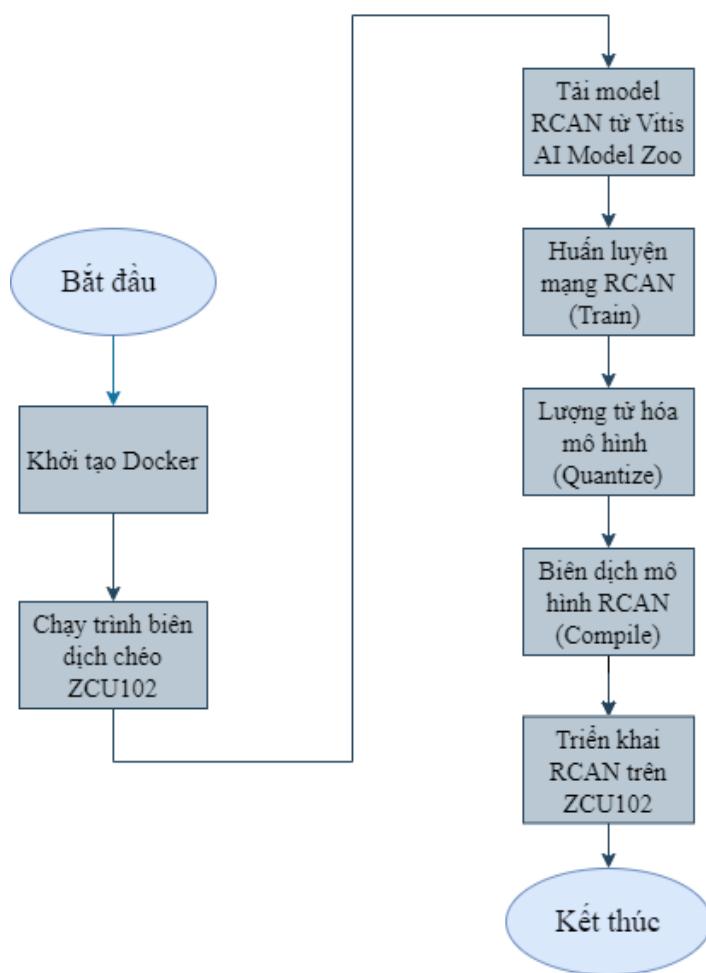
Bước 5: Cập nhật Device Tree giúp mô tả các thành phần phần cứng hệ thống dựa trên cấu hình phần cứng từ file .xsa.

Bước 6: Cài đặt định dạng thẻ SD. Vitis AI cần 2 phân vùng trên thẻ SD đó là FAT32 chứa các tập tin khởi động và EXT4 chứa các cấu hình của Linux chạy trên hệ thống nhúng.

Bước 7: Xây dựng hệ thống. PetaLinux sẽ tiến hành xây dựng các tập tin cần thiết để chạy trên hệ thống nhúng.

❖ Công cụ Vitis AI

Vitis AI SDK được sử dụng để triển khai các mô hình học sâu trên Zynq MPSoC. Để triển khai mô hình, cần phải tuân theo một số bước nhất định.



Hình 3.6. Quy trình thiết kế hệ thống siêu phân giải ảnh trên công cụ Vitis AI

Hình 3.6 mô tả chi tiết các bước làm việc cho dự án siêu phân giải ảnh trên công cụ Vitis AI.

Bước 1: Cài đặt môi trường làm việc cho Vitis AI trên docker, sau đó tiến hành chạy trình biên dịch chéo trên Zynq Ultrascale+ MPSoC-ZCU102.

Bước 2: Khởi tạo mô hình mới hoặc mô hình đã tích hợp sẵn trong thư viện Vitis AI Model Zoo. Trong bài báo cáo này, sử dụng mô hình RCAN được tích hợp sẵn trong thư viện Vitis AI Model Zoo được thiết kế theo thư viện Pytorch. Mục đích nhằm giúp tiết kiệm thời gian và nâng cao độ chính xác của model.

Bước 3: Tiến hành huấn luyện mô hình (Training). Mô hình mạng RCAN được huấn luyện trên công cụ Kaggle. Sử dụng tập dữ liệu có sẵn DIV2K gồm hơn 800 ảnh có độ phân giải cao khác nhau dùng để huấn luyện và 800 ảnh dùng để kiểm tra.

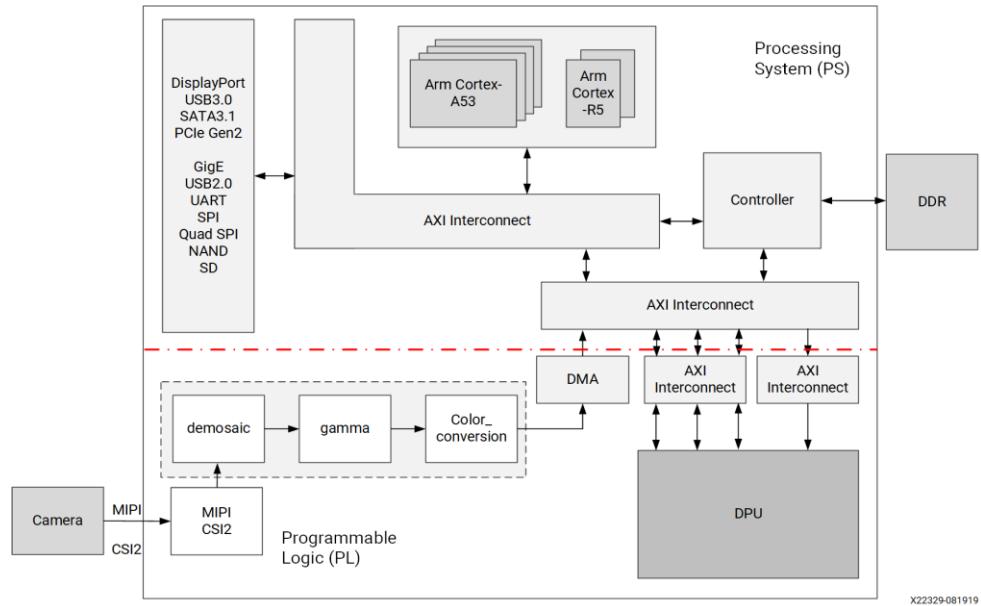
Bước 4: Tiến hành lượng tử hóa mô hình (Quantisation) bằng công cụ Vitis AI Quantisation. Đây là một bước bắt buộc, nếu thiếu bước này thì mô hình không thể triển khai được. Bước này chuyển đổi mô hình RCAN có định dạng mô hình từ FP32 thành INT8.

Bước 5: Tiến hành biên dịch mô hình (Compile) để tạo tập tin có đuôi .elf. Quá trình chuyển đổi mô hình mạng thành mã máy (ngôn ngữ bậc thấp) có thể chạy được trên phần cứng.

Bước 6: Chuyển model RCAN đã được biên dịch vào ZCU102 qua cổng Ethernet. Tiến hành kiểm tra model bằng cách siêu phân giải ảnh của tập tin benhmark.

3.3.1.2. Thiết kế IP Cores trên Vivado IDE

Hệ thống mô tả phần cứng gồm có các IP chính sau: Zynq Ultrascale+ MPSoC, Deep Learning Processor Unit (DPU), Processor System Reset, Clocking Wizard, AXI Interconnect.



Hình 3.7. Mô tả kết nối DPU được tích hợp vào Zynq UltraScale+ MPSOC

Hình 3.7 ở trên minh họa một hệ thống trong đó dữ liệu cần được xử lý được ghi lại bằng thiết bị Zynq® UltraScale+™ MPSoC có camera MIPI được kết nối hoặc được lưu trữ ở bộ nhớ DDR. Deep Learning Processor Unit (DPU) được tích hợp vào hệ thống thông qua kết nối AXI để thực hiện các tác vụ suy luận học sâu.

Khối IP Zynq UltraScale+ MPSOC

Khối IP Zynq UltraScale+ MPSOC chứa các thông tin cấu trúc về thành phần PS và PL của ZCU102. IP Cores này được biểu diễn ở hình 3.8 dưới đây.

**Hình 3.8. IP Zynq UltraScale+ MPSOC**

Thông tin các chân I/O của IP được mô tả chi tiết ở bảng 3.1 dưới đây: [11]

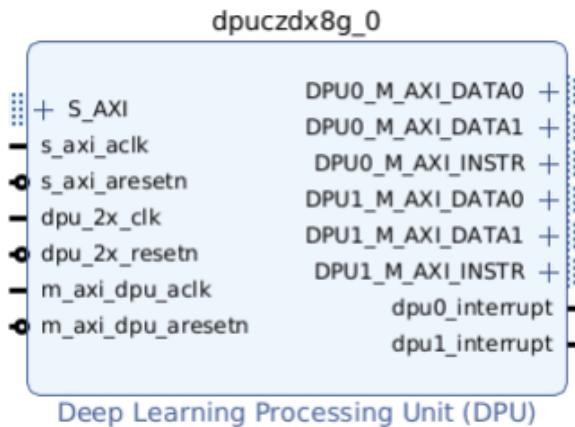
Bảng 3.1. Thông tin các chân I/O của IP Zynq UltraScale+ MPSOC

Tên giao diện	Viết tắt	I/O	Vị trí	Mô tả
S_AXI_HP{0:3}_FPD	HP{0:3}	I	PL	Đường truyền dữ liệu có băng thông lớn từ PS đến PL.
S_AXI_LPD	PL_LPD	I	PL	Đường truyền dữ liệu có băng thông nhỏ từ PS đến PL
M_AXI_HPM(0,1)_FPD	HPM(0,1)	O	PS	Cho phép PL thực hiện đọc/ghi đến bộ nhớ và tài nguyên khác trong PS với hiệu suất cao.
saxihp(0:3)_fpd_aclk		I	PS	Tín hiệu clock có chức năng đồng bộ hóa truy cập (đọc) dữ liệu.
pl_ps_irq(0,1)		I	PL	Tín hiệu ngắt từ PL gửi tới PS.
pl_resetn0			PL	Tín hiệu reset hoạt động của PL.

Khối IP Deep Learning Processor unit (DPU) [10]

Khối IP Deep Learning Processor unit (DPU) có thể được định cấu hình với một số tùy chọn được xác định trước, bao gồm số lõi DPU, kiến trúc tích chập, xếp tầng

DSP, sử dụng DSP và sử dụng UltraRAM.. Hình 3.7 dưới đây hiển thị trang cấu hình của DPU.



Hình 3.9. IP Deep Learning Processor Unit (DPU)

Thông tin về các chân I/O của IP DPU được mô tả trong bảng 3.2 dưới đây:

Bảng 3.2. Thông tin các chân của IP DPU

Tên tín hiệu	Loại tín hiệu	Kích thước	I/O	Mô tả
S_AXI	Giao diện Slave được ánh xạ với bộ nhớ	32	I/O	Giao diện AXI được ánh xạ với bộ nhớ 32 bit cho các thanh ghi.
s_axi_aclk	Clock	1	I	Tín hiệu clock AXI cho S_AXI.
s_axi_aresetn	Reset	1	I	Tín hiệu reset tích cực thấp cho S_AXI.
dpu_2x_clk	Clock	1	I	Tín hiệu clock gấp cho khối DSP của DPUCZDX8G. Tần số gấp đôi m_axi_dpu_aclk.
dpu_2x_resetn	Reset	1	I	Tín hiệu reset tích cực

				thấp cho khối DSP.
m_axi_dpu_aclk	Clock	1	I	Tín hiệu clock cấp cho các khối logic trong DPUCZDX8G.
m_axi_dpu_aresetn	Reset	1	I	Tín hiệu reset tích cực mức thấp cho khối logic.
DPU(0,1)_M_AXI_DATA0	Giao diện Master được ánh xạ với bộ nhớ.	128	I/O	128 bit cho phiên bản Zynq Ultrascale+ MPSoC.
DPU(0,1)_M_AXI_DATA1	Giao diện Master được ánh xạ với bộ nhớ.	128	I/O	128 bit cho phiên bản Zynq Ultrascale+ MPSoC.
DPU(0,1)_M_AXI_INSTR	Giao diện Master được ánh xạ với bộ nhớ.	32	I/O	Giao diện AXI được ánh xạ với bộ nhớ 32 bit dùng để truyền tải các dòng lệnh cho DPUCZDX8G.
dpu(0,1)_interrupt	Ngắt	1	O	Ngõ ra tín hiệu ngắt tích cực mức cao cho DPUXZDX8G.
SFM_M_AXI (tùy chọn)	Giao diện AXI Master được ánh xạ với bộ nhớ.	128	I/O	Giao diện AXI được ánh xạ cho dữ liệu softmax.

- Số lõi DPU: Có thể chọn tối đa 4 lõi trong một IP. Càng nhiều lõi giúp tăng hiệu suất. Ở trường hợp siêu phân giải ảnh, chỉ cần dùng 1 lõi.

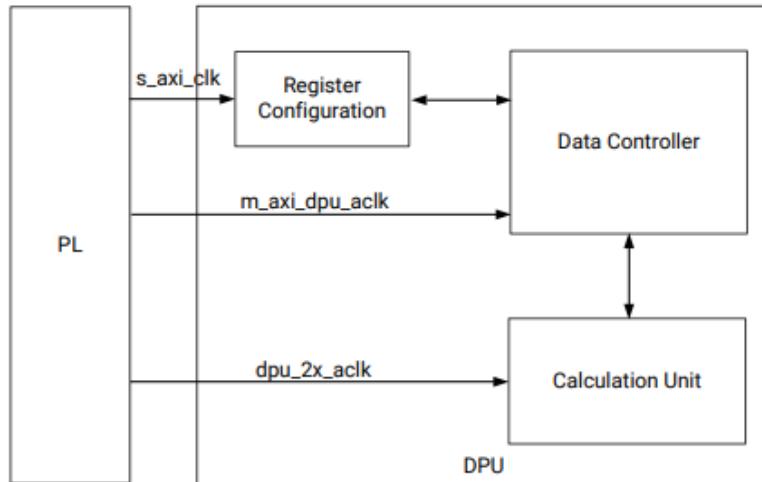
- Kiến trúc DPU: IP DPU có thể được cấu hình với nhiều kiến trúc tích chập khác nhau có liên quan đến tính song song của đơn vị tích chập. Các kiến trúc cho IP DPU bao gồm B512, B800, B1024, B1152, B1600, B2304, B3136 và B4096. Trong trường hợp này sử dụng kiến trúc B4096.
- Mức sử dụng RAM: Xác định tổng dung lượng bộ nhớ trên chip được sử dụng trong các kiến trúc DPU. Quá trình siêu phân giải ảnh chỉ chạy một mô hình CNN nên chọn Low.
- Tùy chọn ALU Parallel và ALU Parallel lần lượt là 4. Alu Relu Type là Relu + Relu5.

Bảng 3.3. Cấu hình DPU IP

Số lượng lõi	1
Kiến trúc DPU	B4096
Mức sử dụng bộ nhớ RAM	Low
Khả năng tăng cường các kênh	Enabled
Loại hàm kích hoạt ReLU	ReLU + ReLU6
Số lượng ALU song song	4
Số lõi Softmax	1
Phiên bản	1.4.1
Chuẩn truyền AXI	AXI4

✍ Khối tạo tín hiệu clock IP Clock Wizard

Trong DPU, có 3 khối cần tín hiệu clock đó là khối cấu hình thanh ghi, khối điều khiển dữ liệu và khối tín toán. Ba tín hiệu clock này có thể được cấu hình độc lập để phù hợp với các yêu cầu về hệ thống và hiệu suất.



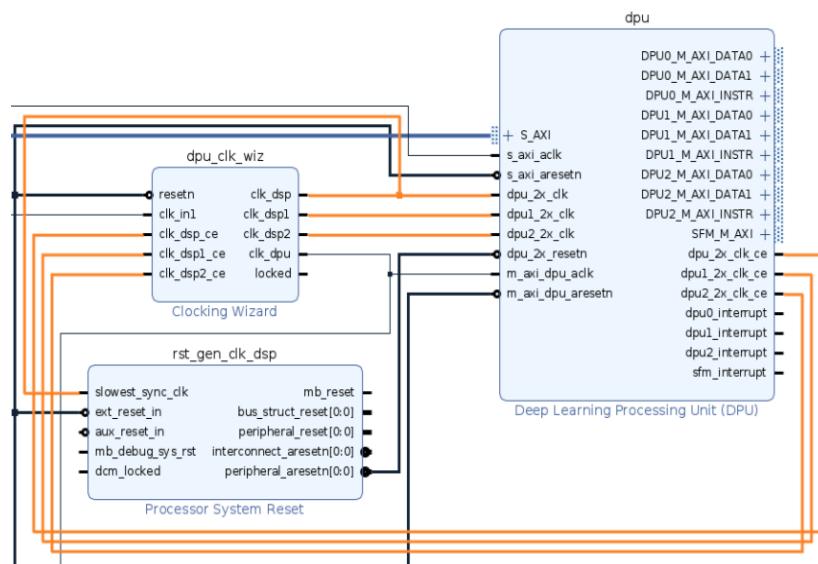
Hình 3.10. Các tín hiệu clock cần thiết cho IP DPUCZDX8G

Tín hiệu *s_axi_aclk* được sử dụng cho mô-đun cấu hình thanh ghi. Mô-đun này nhận cấu hình DPUCZDX8G thông qua giao diện S_AXI. Tín hiệu clock S_AXI có thể chung với tín hiệu clock M_AXI hoặc độc lập. Các thanh ghi cho cấu hình DPUCZDX8G được cập nhật với tần suất rất thấp và hầu hết các thanh ghi này chỉ được đặt tại thời điểm bắt đầu của một tác vụ nào đó. Bởi vì M_AXI trực tiếp ảnh hưởng đến hiệu suất DPU, Xilinx khuyến nghị người dùng cấu hình S_AXI là một tín hiệu clock độc lập với tần số 100 MHz. [10]

Chức năng chính của mô-đun điều khiển dữ liệu là thiết lập thời gian cho các luồng dữ liệu trong IP DPUCZDX8G. Mô-đun điều khiển dữ liệu được đồng bộ hóa từ tín hiệu *m_axi_dpu_aclk*. Việc chuyển dữ liệu giữa DPUCZDX8G và bộ nhớ ngoài được đồng bộ hóa từ tín hiệu clock của mô-đun điều khiển dữ liệu, do đó, tín hiệu *m_axi_dpu_aclk* được coi như là tín hiệu clock cho mô-đun điều khiển dữ liệu và giao diện master AXI_M của DPUCZDX8G. *m_axi_dpu_aclk* nên được kết nối với tín hiệu clock của master AXI_MM.

Các bộ xử lý DSP trong mô-đun đơn vị tính toán nằm trong miền tín hiệu clock của *dpu_2x_clk*, tín hiệu này chạy ở tần số gấp đôi tần số của mô-đun điều khiển dữ liệu.

Để đáp ứng được những yêu cầu trên, IP Clock Wizard được sử dụng trong thiết kế này. Thông tin về kết nối giữa IP Clock Wizard với IP DPU được mô tả ở hình 3.11 dưới đây:



Hình 3.11. Kết nối giữa IP Clocking Wizard và IP DPU

Thông số các ngõ I/O của IP được biểu diễn trong bảng 3.4 ở dưới:

Bảng 3.4. Chức năng các chân của IP Clocking Wizard [12]

Tên tín hiệu	I/O	Mô tả
reset/resetn	I	Chân này có thể cấu hình trong phần cài đặt. Tín hiệu sẽ tích cực mức cao khi được đặt là reset, tích cực mức thấp khi đặt là resetn.
clk_in1	I	IP Clocking Wizard sẽ tạo tín hiệu clock ngõ ra dựa vào clock ngõ vào tại <i>clk_in1</i> .
clk_out(dsp, dsp1,	O	Ngõ ra của tín hiệu clock được tạo bởi IP Clocking

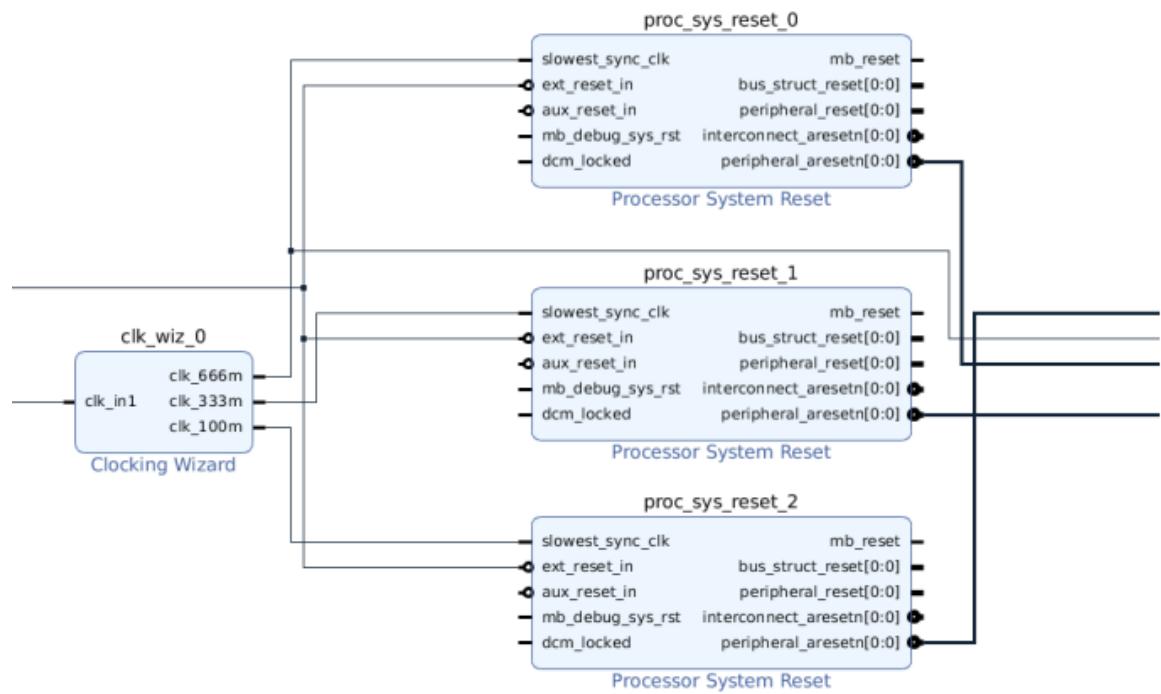
dsp2, dpu)		Wizard.
clk_out_ce(dsp, dsp1, dsp2,dpu)	I	Trạng thái của các bộ đệm ở ngõ ra. Các bộ đệm này có thể là BUFGCE, BUFHCE, BUFR, hoặc BUFGCE_DIV được sử dụng để điều khiển các tín hiệu clock ở ngõ ra.
locked	O	Tín hiệu này được kích hoạt khi tín hiệu clock ngõ ra đã hoạt động đúng như được thiết lập.

Tần số `s_axi_aclk` được đặt thành 100 MHz và `m_axi_dpu_aclk` được đặt thành 325 MHz. Tần số của `dpu_2x_clk` được đặt thành 650 MHz. Cấu hình cho IP Clock Wizard được hiển thị trong bảng 3.2 dưới đây:

Bảng 3.5. Cấu hình IP Clock Wizard [10]

Tên tín hiệu	Thiết lập	Mô tả
clk_DSP	650MHz	Tần số của tín hiệu <code>dpu_2x_clk</code> .
clk_DPU	325MHz	Tần số của tín hiệu <code>m_axi_dpu_aclk</code> .
Duty Cycle	50%	Độ rộng xung của mức cao và mức thấp bằng nhau.
Driver	Buffer with CE	Driver là bộ đệm với cấu hình CE (Clock Enable).

Khởi tạo tín hiệu reset IP Processor System Reset



Hình 3.12. Cấu hình IP Processor System Reset

Thiết kế kết nối các tín hiệu clock và reset mô tả trong hình 3.12 ở trên.

Đối với IP DPUCZDX8G, có ba tín hiệu clock ở ngõ vào nên mỗi tín hiệu clock sẽ có một tín hiệu reset tương ứng. [10] Mỗi tín hiệu reset phải được đồng bộ hóa với tín hiệu clock tương ứng. Nếu các tín hiệu clock và reset không được đồng bộ thì IP DPU không thể hoạt động chính xác và hiệu quả. Vì vậy, khởi IP Processor System Reset được sử dụng để tạo ra tín hiệu reset được đồng bộ hóa.

Bảng 3.6. Các chức năng của IP processor System Reset [13]

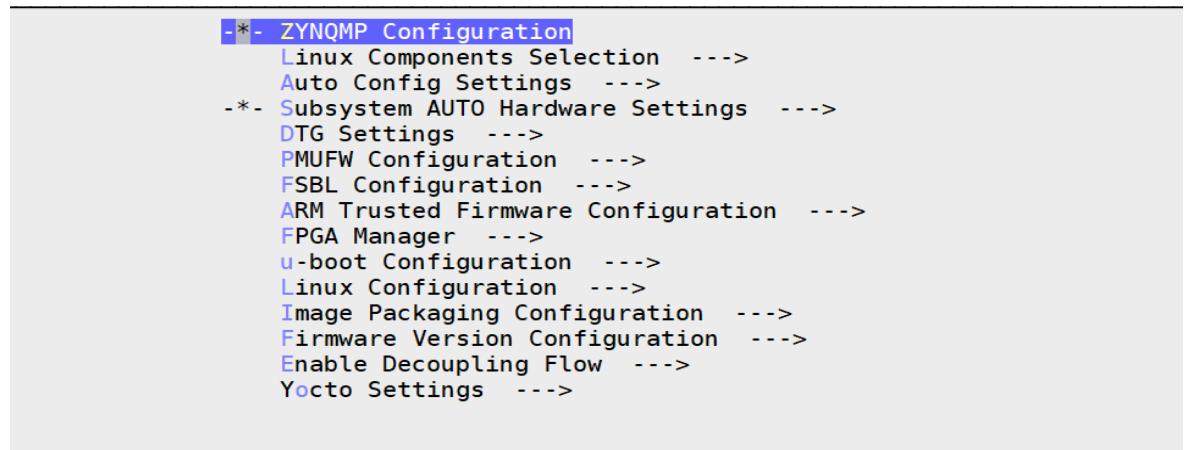
Tên tín hiệu	I/O	Mô tả
slowest_sync_clk	I	Tín hiệu clcok được đồng bộ chậm nhất.
ext_reset_in	I	Ngõ vào tín hiệu reset bên ngoài. Tích cực mức cao hoặc mức thấp dựa vào

		cạnh tích cực của tín hiệu Ext Reset Active.
mb_debug_sys_RST	I	Ngõ vào tín hiệu reset MDM. Luôn tích cực mức cao, kích thước được đặt theo kích thước của tín hiệu External Reset Active Window.
aux_reset_in	I	Ngõ vào tín hiệu reset. Tích cực mức cao hoặc thấp dựa vào cạnh tích cực của tín hiệu Auxiliary Reset Active.
dcm_locked	I	DCM sẽ khóa tín hiệu clock khi tín hiệu này đã đạt đến tần số và pha mong muốn.
mb_RESET	O	Reset lõi tín hiệu MB (Micro Blaze). Tích cực mức cao.
bus_struct_RESET [0:0]	O	Tín hiệu reset cho kiểu cấu trúc Bus. Tích cực mức cao.
peripheral_RESET [0:0]	O	Tín hiệu reset cho các thiết bị ngoại vi được kết nối trên cùng một bus và được đồng bộ với tín hiệu slowest_sync_clk. Tích cực mức cao.
interconnect_RESETn [0:0]	O	Tín hiệu reset cho các liên kết giữa các IP. Tích cực mức thấp.
peripheral_RESETn [0:0]	O	Tín hiệu reset cho các thiết bị ngoại vi có các liên kết được đồng bộ với tín hiệu slowest_sync_clk. Tích cực mức thấp.

3.3.1.3. Xây dựng hệ thống nhúng

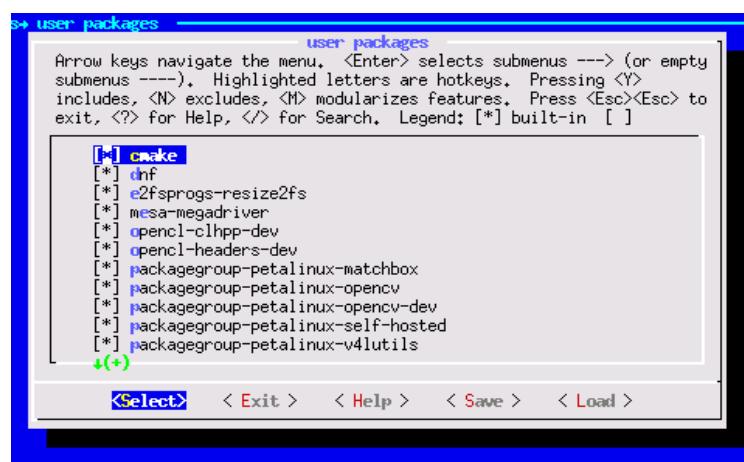
Mục tiêu là khởi tạo được các tập tin khởi động cho ZCU102 có IP mô tả phần cứng là file **.xsa** được tạo ra ở phần mềm Vivado. Công cụ PetaLinux do Xilinx phát triển được sử dụng trong phần này.

- Đầu tiên là cập nhật thông tin phần cứng ZCU102 ở phần DTG Setting trong mục cấu hình của dự án.



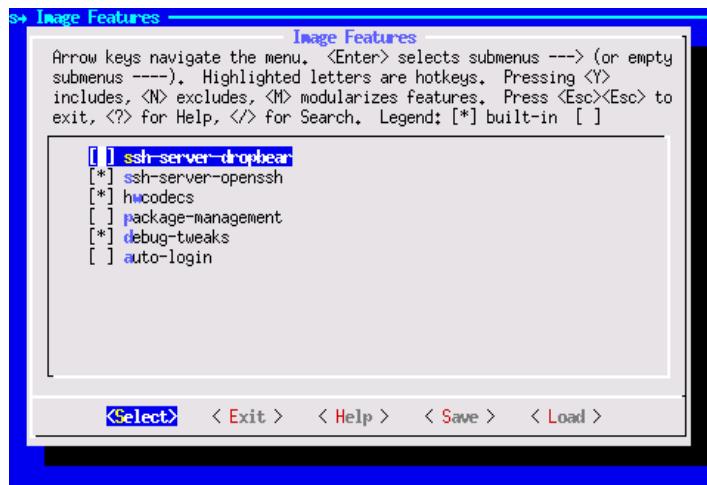
Hình 3.13. Các thông số cấu hình cho dự án PetaLinux

Kích hoạt các gói package cho dự án như *packagegroup-petalinux-vitisai*, *cmake*, *xrt_dev*...



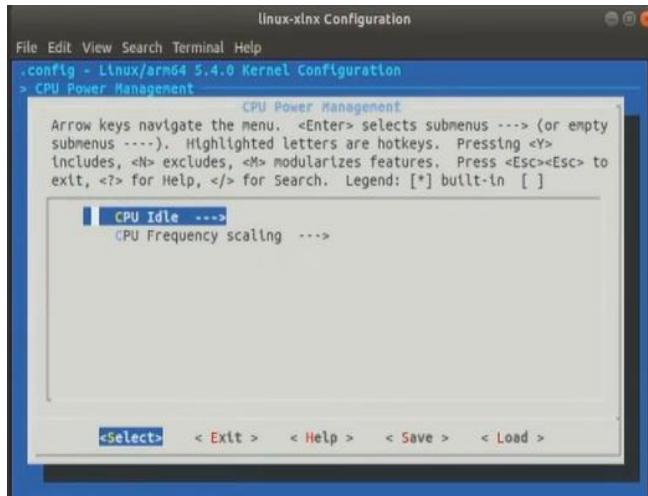
Hình 3.14. Thiết lập các gói package người dùng

- Kích hoạt OpenSSH và vô hiệu hóa Dropbear. Dropbear là công cụ SSH mặc định trong Nền tảng nhúng Vitis IDE. Nếu OpenSSH được sử dụng để thay thế Dropbear, hệ thống có thể đạt tốc độ truyền dữ liệu nhanh hơn gấp 4 lần qua ssh (ug1144). Vì các ứng dụng Vitis-AI có thể sử dụng tính năng hiển thị từ xa để hiển thị kết quả học máy nên việc sử dụng OpenSSH có thể cải thiện trải nghiệm hiển thị.



Hình 3.15. Kích hoạt SSH và vô hiệu hóa Dropbear

- Vô hiệu hóa CPU IDLE trong Kernel. CPU IDLE sẽ khiến bộ xử lý chuyển sang trạng thái IDLE (trạng thái chờ) khi bộ xử lý không được sử dụng. Khi JTAG được kết nối, môi trường thiết kế phần cứng (hardware server) trên Vitis IDE sẽ thường xuyên trao đổi với bộ xử lý. Nếu nó giao tiếp với bộ xử lý ở trạng thái IDLE, hệ thống sẽ bị treo do các giao dịch AXI chưa hoàn tất. Do đó, nên tắt tính năng CPU IDLE trong giai đoạn cài đặt dự án.



Hình 3.16. Vô hiệu hóa CPU IDLE

- Từ bất kỳ thư mục nào trong dự án PetaLinux “**zcu102_platform**”, sử dụng lệnh: `petalinux-build`

Câu lệnh “*petalinux-build*” dùng để xây dựng một hệ thống nhúng chạy hệ điều hành Linux hoạt động trên Xilinx Zynq Ultrascale+ MPSoC - ZCu102. “*petalinux-build*” xây dựng các kernel Linux, hệ thống các tập tin, phần mềm ứng dụng và các thành phần cần thiết từ mã nguồn và cấu hình đã được cài đặt ở trên. Cụ thể quá trình xây dựng được mô tả như sau:

- **Biên dịch Kernel Linux:** Sử dụng công cụ biên dịch chéo Cross-Compiler để biên dịch mã nguồn của Kernel Linux thành mã máy cho ZCU102 có thể hiểu được. Kết quả là tạo được file có tên “image”.
- **Xây dựng hệ thống tập tin Rootfs:** PetaLinux tạo ra một hệ thống tệp rootfs hoàn chỉnh, bao gồm các thư mục, tập tin và phân quyền cần thiết để chạy một hệ thống Linux hoàn chỉnh. Quá trình này bao gồm việc thêm các thư viện, ứng dụng và tệp cấu hình được yêu cầu bởi Kernel.
- **Xây dựng BootLoader và Firmware:** Xây dựng và biên dịch mã nguồn của bootloader (U-boot) và các tệp firmware (device tree) dành cho board mạch.

Quá trình xây dựng dự án tốn khá nhiều thời gian (khoảng 2 giờ), các tập tin cần thiết được tạo khi quá trình xây dựng kết thúc được lưu trong thư mục **/images/linux/**

3.3.1.4. Tiềm xử lý hệ thống siêu phân giải ảnh

Phần này thực hiện trên phần mềm Vitis AI được Xilinx phát triển. Mô hình thuật toán siêu phân giải ảnh được cung cấp trên kho thư viện Vitis AI Model Zoo.



Hình 3.17. Kho thư viện Vitis AI Model Zoo

Vitis AI Model Zoo được xây dựng trên các framework học sâu phổ biến như TensorFlow và PyTorch. Model Zoo cung cấp một nguồn tài nguyên phong phú cho nhà phát triển và nghiên cứu để triển khai các ứng dụng trí tuệ nhân tạo trên các thiết bị Xilinx.

❖ Tải mô hình siêu phân giải ảnh

Model Zoo cung cấp mô hình siêu phân giải ảnh được định dạng theo cấu trúc của thư viện Pytorch có tên là **pt-OFA-rcan_DIK2K_360_640_45.7G_2.5**. Sau khi tải thư viện Vitis AI từ Github và truy cập bằng công cụ docker, ta có môi trường làm việc của Vitis-AI như sau:

```
=====
Vitis-AI
=====

Docker Image Version: 2.5.0.1260 (CPU)
Vitis AI Git Hash: 502703c
Build Date: 2022-06-12

For TensorFlow 1.15 Workflows do:
  conda activate vitis-ai-tensorflow
For PyTorch Workflows do:
  conda activate vitis-ai-pytorch
For TensorFlow 2.8 Workflows do:
  conda activate vitis-ai-tensorflow2
For WeGo Tensorflow 1.15 Workflows do:
  conda activate vitis-ai-wego-tf1
For WeGo Tensorflow 2.8 Workflows do:
  conda activate vitis-ai-wego-tf2
For WeGo Torch Workflows do:
  conda activate vitis-ai-wego-torch
vitis-AI /workspace > conda activate vitis-ai-pytorch
(vitis-ai-pytorch) vitis-AI /workspace > ls
docker docker_run.sh docs examples index.html LICENSE model_zoo RCAN README.md reference_design setup src third_party
(vitis-ai-pytorch) vitis-AI /workspace > cd model_zoo/model_list/pt_OFA-rcan_DIV2K_360_640_45.7G_2.5
(vitis-ai-pytorch) vitis-AI /workspace/model_zoo/model_list/pt_OFA-rcan_DIV2K_360_640_45.7G_2.5 > ls
model.yaml
```

Hình 3.18. Môi trường làm việc Vitis AI trên Docker

- Trong bài báo cáo này, sử dụng phiên bản Vitis AI 2.5. Các phần mềm PetaLinux, Vivado và Vitis sử dụng phiên bản được phát hành vào năm 2022.2.
- Truy cập vào môi trường làm việc của thư viện pytorch, sau đó truy cập vào thư mục có tên *pt_OFA-rcan-DIV2K_360_640_45.7G_2.5*.
- Nội dung của file ***model.yaml*** chứa thông tin về các gói thư viện RCAN dùng cho từng phần cứng khác nhau và liên kết để tải.
- Sau khi tải về, mô hình RCAN có cấu trúc được mô tả trong hình dưới đây:

```
(vitis-ai-pytorch) Vitis-AI /workspace/RCAN/pt_OFA-rcan_DIV2K_360_640_45.7G_2.5 > tree -L 2
.
├── code
│   ├── data
│   ├── dataloader.py
│   ├── __init__.py
│   └── LICENSE
│   └── loss
│       ├── metric.py
│       └── model
│           ├── option.py
│           ├── run_qat.sh
│           ├── run_quant.sh
│           ├── run_test.sh
│           ├── run_train.sh
│           ├── template.py
│           ├── test.py
│           ├── trainer.py
│           ├── train.py
│           └── train_qat.py
│   └── utility.py
└── data
    └── float
        └── model_float.pt
            └── pretrained
                └── quantized
                    ├── best_quantized_model.pth
                    ├── bias_corr.pth
                    ├── deploy_latest.pth
                    ├── Model_int.xmodel
                    ├── Model.py
                    ├── param.pth
                    └── __pycache__
                        └── qat_result
                            └── quant_info.json
    └── readme.md
    └── requirements.txt
```

Hình 3.19. Các tập tin có trong mô hình RCAN

- + **code:** Thư mục này chứa tất cả mã nguồn của dự án. Các thư mục con được mô tả trong

Bảng 3.7. Các thư mục con trong file code của model RCAN

data	Thư mục này có thể chứa dữ liệu đào tạo và dữ liệu thử nghiệm.
dataloader.py	Tệp này chứa mã để tải dữ liệu vào mô hình.
__init__.py	Tệp này chứa mã khởi tạo cho thư mục “code”.
LICENSE	Tệp này chứa thông tin về giấy phép sử dụng mã nguồn mở.
loss	Thư mục này chứa các hàm mất mát (loss functions) của mô hình.
metric.py	Tệp này chứa mã để tính các độ đo (metrics) cho mô hình.
model	Thư mục này chứa file model RCAN được viết bằng python.
option.py	Tệp này chứa các tùy chọn cấu hình cho dự án.
run_qat.sh, run_quant.sh, run_test.sh, run_train.sh	Các tệp shell script để chạy các công đoạn khác nhau của dự án (lượng tử, kiểm thử, huấn luyện vv.). Quá trình huấn luyện sẽ chạy trên Kaggle. Quá trình lượng tử và biên dịch sẽ chạy trên Vitis-AI.
template.py	Chương trình mẫu để bắt đầu viết mô hình mới.
test.py	Chương trình để thử nghiệm mô hình.
train.py, train_qat.py	Chương trình để huấn luyện mô hình hoặc mô hình được tối ưu hóa (quantized-aware training).
utility.py	Các hàm tiện ích hoặc hàm hỗ trợ cho dự án.

-  **data:** Thư mục này chứa dữ liệu huấn luyện và dữ liệu thử nghiệm. Tập tin dùng để huấn luyện là tập dữ liệu DIV2K và tập tin dùng để thử nghiệm là benchmark.

-  **float:** Thư mục này chứa các tệp tin liên quan đến mô hình dự đoán không tối ưu hóa.

Bảng 3.8. Các thư mục con trong file float của model RCAN

model_float.pt	Mô hình được huấn luyện ban đầu (không tối ưu hóa).
pretrained	Thư mục này chứa các mô hình được đào tạo trước (pretrained) từ nguồn dữ liệu khác.

-  **quantized:** Thư mục này chứa các tệp liên quan đến mô hình dùng để lượng tử hóa.

Bảng 3.9. Các thư mục con trong file float của model RCAN

best_quantized_model.pth	Mô hình được tối ưu hóa tốt nhất.
bias_corr.pth	Các tham số bias được điều chỉnh sau quá trình tối ưu hóa.
deploy_latest.pth	Mô hình đã được tối ưu hóa sẵn sàng triển khai.
Model_int.xmodel	Mô hình tối ưu hóa dạng file của Xilinx.
Model.py	Chương trình của mô hình đã được tối ưu hóa.
param.pth	Các tham số của mô hình sau quá trình tối ưu hóa.
qat_result	Kết quả sau quá trình huấn luyện tối ưu hóa trước quantization-aware (QAT).
quant_info.json	Thông tin về quá trình tối ưu hóa quantization.

-  **readme.md:** Tệp này chứa thông tin về dự án, hướng dẫn cài đặt, và các hướng dẫn sử dụng.

- ❖ **frequirements.txt:** Tệp này chứa danh sách các thư viện và phiên bản cần thiết để chạy dự án.

❖ Huấn luyện mô hình

Dữ liệu dùng cho huấn luyện mô hình là tập dữ liệu DIV2K. Đây là một bộ dữ liệu được sử dụng rộng rãi trong lĩnh vực xử lý ảnh và thị giác máy tính để đánh giá và so sánh hiệu suất của các thuật toán siêu phân giải hình ảnh. Tên "DIV2K" được viết tắt từ "Diverse 2K resolution dataset", tương tự như "Bộ dữ liệu đa dạng độ phân giải 2K". Bộ dữ liệu này bao gồm 2.000 hình ảnh, được chia thành tập huấn luyện và tập kiểm tra. Mỗi hình ảnh trong DIV2K được chia thành các tập hợp riêng biệt để đảm bảo tính đa dạng và độ phong phú của dữ liệu.

Việc huấn luyện mô hình cần nhiều tài nguyên của máy tính, nên trong báo cáo này, sẽ thực hiện huấn luyện mô hình trên Cloud, cụ thể là trên môi trường Kaggle. Thông số cho quá trình huấn luyện được mô tả trong bảng 3.10 dưới đây:

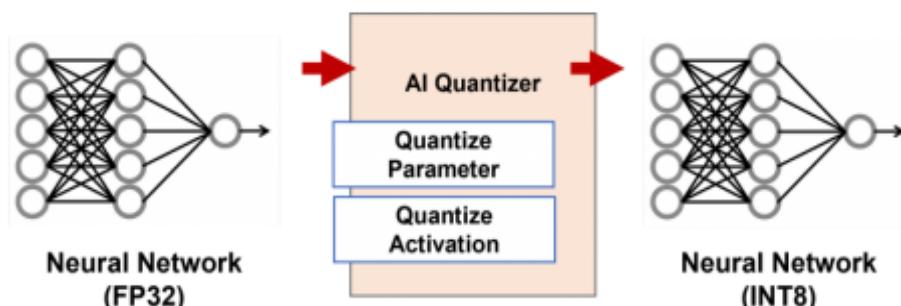
Bảng 3.10. Thông số quá trình huấn luyện mạng RCAN

Thông số	Giá trị	Mô tả
LR (Learning Rate)	3e-4 (0.003)	Tốc độ học của mô hình. Kích thước của bước điều chỉnh trọng số trong mỗi lần epoch được cập nhật.
BETAS	0.9 , 0.999	Tham số beta cho thuật toán của quá trình tối ưu hóa Adam. Đại diện cho hệ số giảm dần của các trung bình động bậc nhất (Momentum) và bậc hai của gradient.

EPS (Elipson)	1e-8	Đây là một giá trị nhỏ được thêm vào để ngăn chặn lỗi chia cho 0 trong tính toán.
STEP_SIZE	10	Cứ mỗi sau 10 epoch thì tốc độ học sẽ được điều chỉnh theo hệ số GAMMA.
GAMMA	0.67	Đây là hệ số làm giảm tốc độ học, chỉ định tỷ lệ mà tốc độ học sẽ bị giảm sau mỗi STEP_SIZE. Cụ thể, tốc độ học sẽ giảm 67% sau mỗi 10 epoch.
NUM_EPOCHS	100	Tổng số epoch mà mô hình sẽ huấn luyện.
DEVICE	cuda	Chọn thiết bị mà mô hình sẽ được huấn luyện. Được đặt là cuda (GPU). Nếu không có GPU thì sẽ huấn luyện trên CPU.

❖ Lượng tử hóa mô hình

Sau khi huấn luyện mô hình, các trọng số không quan trọng được loại bỏ bằng phương pháp cắt tỉa (Prune) nhưng mô hình lúc này vẫn rất phức tạp. Để mô hình hoạt động trên SoC (System on chip) thì mô hình cần được đơn giản nhưng vẫn giữ nguyên được hiệu suất hoạt động. Quá trình đơn giản hóa mô hình được gọi là lượng tử hóa.



Hình 3.20. Mô tả chức năng của quá trình lượng tử

Lượng tử hóa là quá trình chuyển đổi trọng số từ giá trị Dấu phẩy động 32 bit FP32 (Floating Point 32 bit) sang biểu diễn Số nguyên 8 bit (INT8). Đây là bước bắt buộc vì DPU chỉ chấp nhận các giá trị biểu diễn số nguyên 8bit. Để xử lý tác vụ này, Vitis AI cung cấp một công cụ `vai_q_pytorch`.

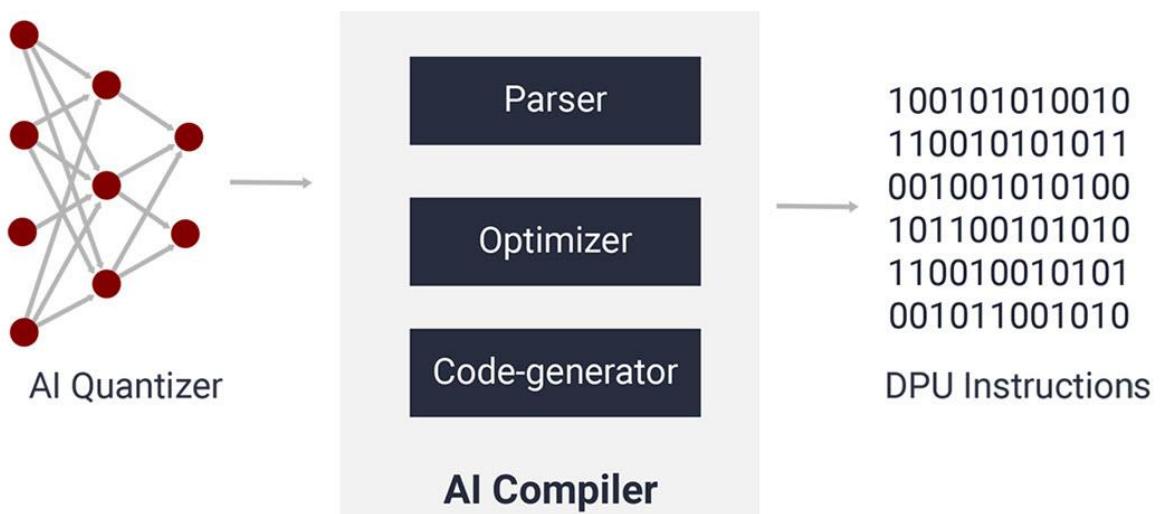
Tiến hành chạy file `run_quant.sh` với các thông số được cài đặt như sau:

- scale : được đặt là 2.
- n_resgroups: được đặt là 1.
- epochs: Vòng lặp được đặt là 250.

Công cụ `vai_q_pytorch` sẽ tiến hành lượng tử mô hình có các thông số trong tập tin `run_quant.py`. Sau khi quá trình lượng tử hoàn tất sẽ tạo ra một file có tên là **`Model_int.xmodel`**. Tập tin này chứa mô hình đã được định dạng thành INT8, có thể triển khai trên DPU.

❖ Biên dịch mô hình mạng

Sau khi tiến hành lượng tử hóa mô hình, bước cuối cùng là tiến hành biên dịch mô hình mạng.



Hình 3.21. Trình biên dịch Vitis AI

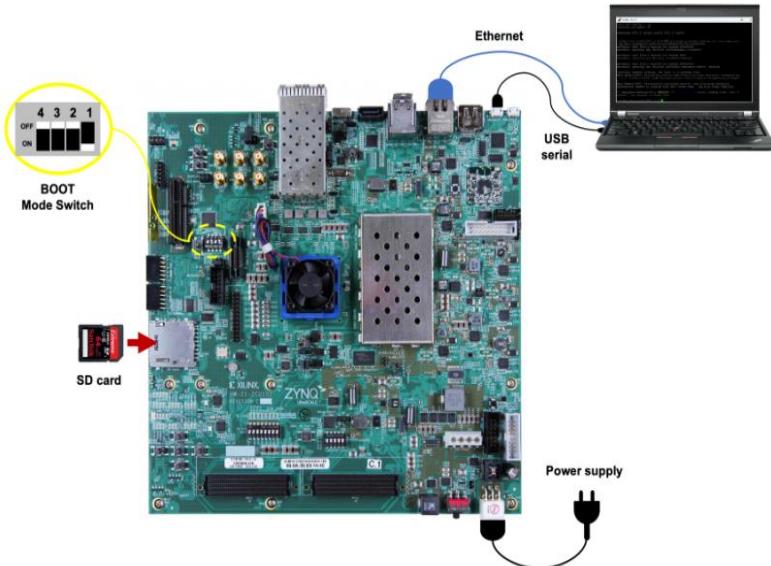
Trình biên dịch mô hình mạng sẽ biên dịch mô hình đã được lượng tử thành đoạn mã bitstream và các tập lệnh có thể chạy trên DPU. Tiến hành biên dịch mô hình mạng RCAN bằng dòng lệnh dưới đây:

```
vai_c_xir -x /workspace/RCAN/pt_OFA-rcan-
DIV2K_360_640_45.7G_2.5/quantized/Model_int.xmodel -a
/vitis_ai/compiler/arch/DPUCZDX8G/ZCU102/arch.json --
output_dir /workspace/RCAN/pt_OFA-rcan-
DIV2K_360_640_45.7G_2.5/compile -n rcan
```

Đoạn mã trên sẽ sử dụng công cụ vai_c_xir-x để biên dịch mô hình đã được lượng tử có tên Model_int.xmodel và file .json mô tả kiến trúc phần cứng là ZCU102. Sau khi biên dịch hoàn tất, VAI_C sẽ tạo tập tin có tên rcan.xmodel để triển khai trên board mạch.

3.3.2. Nền tảng phần cứng

Thiết kế phần cứng áp dụng những kết quả của quá trình phát triển nền tảng phần mềm, được mô tả ở hình 3.23 dưới đây:

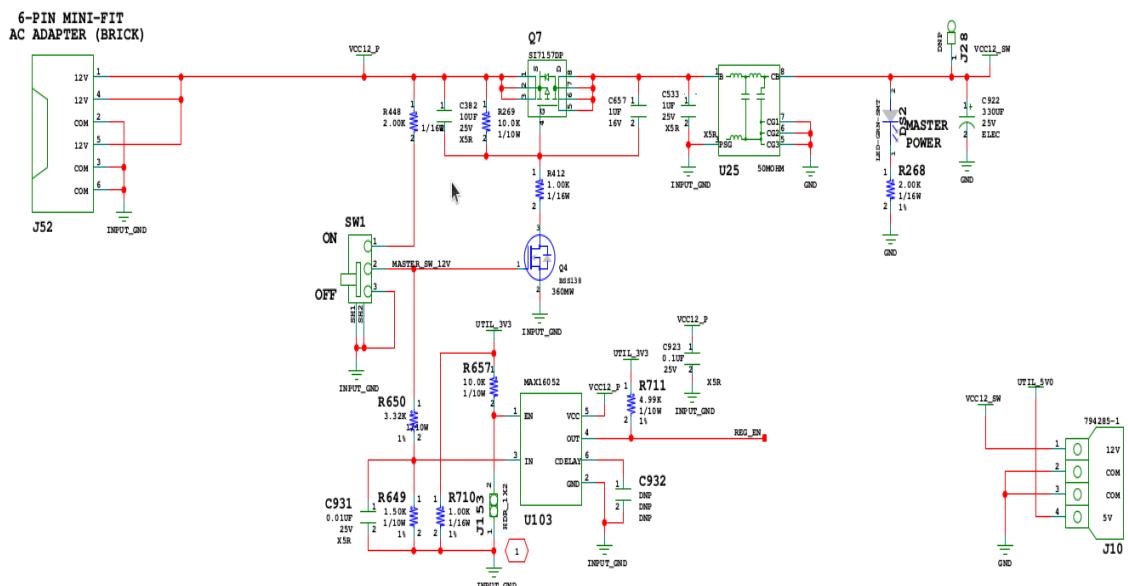


Hình 3.22. Quy trình thiết kế phần cứng

- Xác định kết nối truyền dữ liệu giữa máy tính và phần cứng ZCU102 qua giao diện Ethernet.
- Ghi các tập tin được xây dựng bởi công cụ PetaLinux vào thẻ SD để tiến hành khởi động ZCU102 ở chế độ SD Boot.
- Tiến hành thiết lập các kết nối phần cứng như nguồn, USB UART,
- Triển khai mô hình siêu phân giải ảnh trên ZCU102 sử dụng mạng nơ-ron tích chập RCAN.

3.3.2.1. Khối nguồn

Zynq Ultrascale+ MPSOC - ZCU102 sử dụng nguồn điện một chiều có sơ đồ kết nối được mô tả ở hình 3.20 dưới đây:

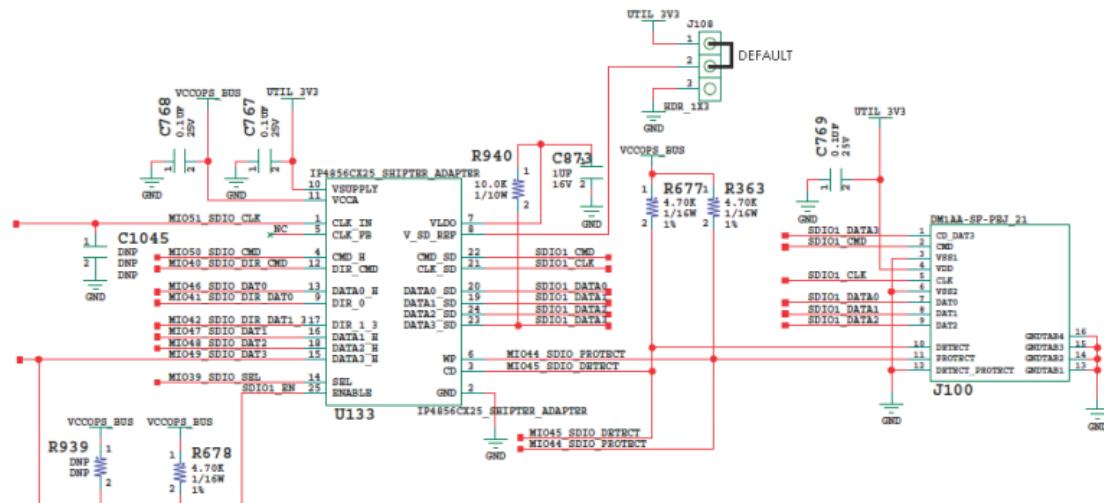


Hình 3.23. Kết nối nguồn cho ZCU102

Công tắc nguồn của bo mạch ZCU102 là SW1. Trượt công tắc từ vị trí Tắt (OFF) sang Bật (ON) sẽ cấp nguồn 12V DC từ J52 là đầu nối mini 6 chân. Đèn LED DS2 màu xanh lục sáng khi nguồn của ZCU102 được bật. Ngoài ra hệ thống còn có IC điều khiển MOSFET SI7157DP có chức năng chống đảo pha, bảo vệ điện áp, quá dòng và IC MAX16052 giúp quản lý nguồn điện vào ra.

3.3.2.2. Khối giao diện SD

Thẻ SD là nơi lưu trữ và truy cập dữ liệu cho ZCU102. Do đó thẻ cần được định dạng đúng với yêu cầu làm việc và được kết nối chính xác. Sơ đồ kết nối giữa thẻ SD với ZCU102 được mô tả ở hình 3.21 dưới đây:



Hình 3.24. Sơ đồ kết nối thẻ SD

Các tín hiệu SDIO được kết nối với XCZU9EG MPSoC PS Bank501 [1] có nguồn điện được cung cấp cho I/O (VCCMIO) được đặt thành 1,8V. Mỗi sáu chân MIOxx_SDIO_* đều có điện trở nối tiếp 30 ohm ở nguồn. Bộ chuyển đổi mức điện áp tương ứng là NXP IP4856CX25 SD 3.0 U133 nằm ở giữa ZCU102 và đầu nối thẻ SD (J100). Thiết bị NXP IP4856CX25 U133 cung cấp khả năng SD3.0 với hiệu suất SDR104. Kết nối chi tiết hiển thị trong bảng dưới đây:

Bảng 3.11. Các kết nối giữa giao diện thẻ SD với XCZU9EG (U1)

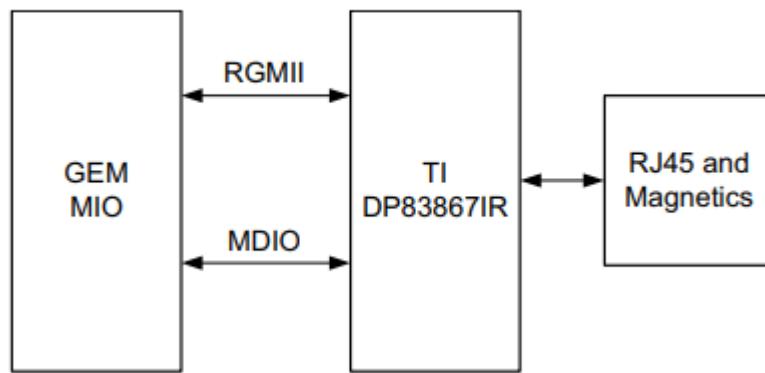
XCZU9EG (U1)	Tên chân	U133 IP4856CX25 Adapter	
		Sô chân	Tên chân
N23	MIO39_SDIO_SEL	14	SEL

M23	MIO40_SDIO_DIR_CMD	12	DIR_CMD
J24	MIO41_SDIO_DIR_DAT0	9	DIR_0
M24	MIO42_SDIO_DIR_DAT1 _3	17	DIR_1_3
J25	MIO46_SDIO_DAT0	13	DATA0_H
L25	MIO47_SDIO_DAT1	16	DATA1_H
M25	MIO48_SDIO_DAT2	18	DATA2_H
K25	MIO49_SDIO_DAT3	15	DATA3_H
P25	MIO50_SDIO_CMD	4	CMD_H
N25	MIO51_SDIO_CLK	1	CLK_IN
N24	MIO44_SDIO_PROTECT	6	WP
P24	MIO45_SDIO_DETECT	3	CD

3.3.2.3. Khối ngõ vào ETHERNET PHY LED

Ngoài việc nhận dữ liệu cục bộ từ thẻ SD, ZCU102 có thể nhận dữ liệu trực tiếp thông qua giao diện GEM3 Ethernet.

Gigabit Ethernet MAC (GEM) triển khai giao diện Ethernet có tốc độ 1=10/100/1000 Mb/s, được hiển thị trong Hình 3.22, kết nối với TI DP83867IRPAP Ethernet RGMII PHY trước khi được định tuyến đến đầu nối Ethernet RJ45. RGMII Ethernet PHY được khởi động ở địa chỉ PHY 5'b01100 (0x0C) và Auto Negotiation được đặt thành Bật.

**Hình 3.25. Sơ đồ khái kết nối Ethernet**

Phần cứng ZCU102 sử dụng TIDP83867IRPAP tại U98 [1] để sử dụng Ethernet ở tốc độ 10Mb/s, 100Mb/s hoặc 1000Mb/s. Các kết nối Ethernet từ XCZU9EG MPSoC U1 tới thiết bị PHY DP83867IRPAP tại U98 được liệt kê trong Bảng 3.6 dưới đây.

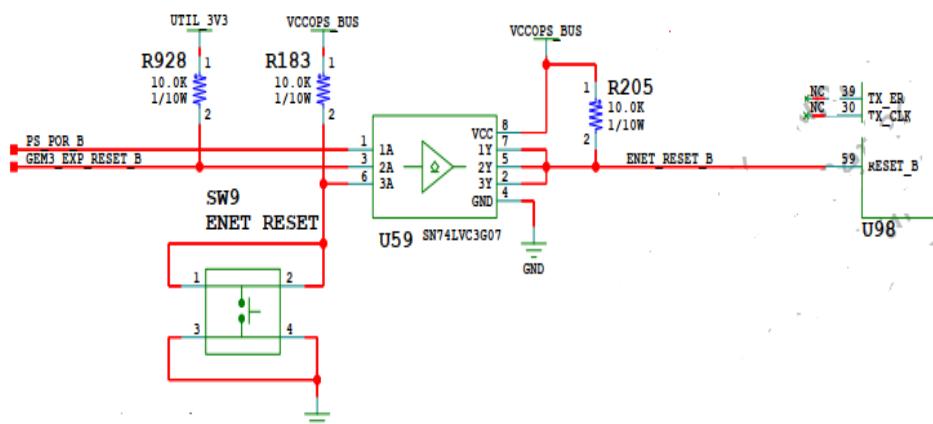
Bảng 3.12. Các kết nối Ethernet, XCZU9EG MPSoC với thiết bị PHY

XCZU9EG (U1)	Tên chân	DP83867 PHY (U98)	
		Số chân	Tên chân
A25	MIO64_ENET_TX_CLK	40	GTX_CLK
A26	MIO65_ENET_TX_D0	38	TX_D0
A27	MIO66_ENET_TX_D1	37	TX_D1
B25	MIO67_ENET_TX_D2	36	TX_D2
B26	MIO68_ENET_TX_D3	35	TX_D3
B27	MIO69_ENET_TX_CTR L	52	TX_EN_TX_CT RL
C26	MIO70_ENET_RX_CLK	43	RX_CLK

C27	MIO71_ENET_RX_D0	44	RX_D0
E25	MIO72_ENET_RX_D1	45	RX_D1
H24	MIO73_ENET_RX_D2	46	RX_D2
G25	MIO74_ENET_RX_D3	47	RX_D3
D25	MIO75_ENET_RX_CTR L	53	RX_DV_RX_CT RL
H25	MIO76_ENET_MDC	20	MDC
F25	MIO77_ENET_MDIO	21	MDIO

Chế độ Reset Ethernet PHY: Mạch Reset DP83867IRPAP PHY (U98) được hiển thị trong hình 3.23 dưới đây. Có thể Reset DP83867IRPAP bằng nút nhấn SW9 (U59.6), thiết bị Reset POR phía PS MAX16025 (U22) MPSoC (U59.1) hoặc cổng mở rộng (U97) TCA6416A I/O được kết nối I2C0 P06 chân số 10 (U59. 3).

[1]



Hình 3.26. Sơ đồ kết nối mạch Reset

Giao diện LED DP83867IRPAP PHY U98 (LED_0, LED_2) sử dụng hai đèn LED được nhúng trong khung đầu nối (P12) RJ45. Mô tả chức năng LED được hiển thị trong bảng 3.7 dưới đây:

Bảng 3. 13. Chức năng các đèn led của giao diện Ethernet

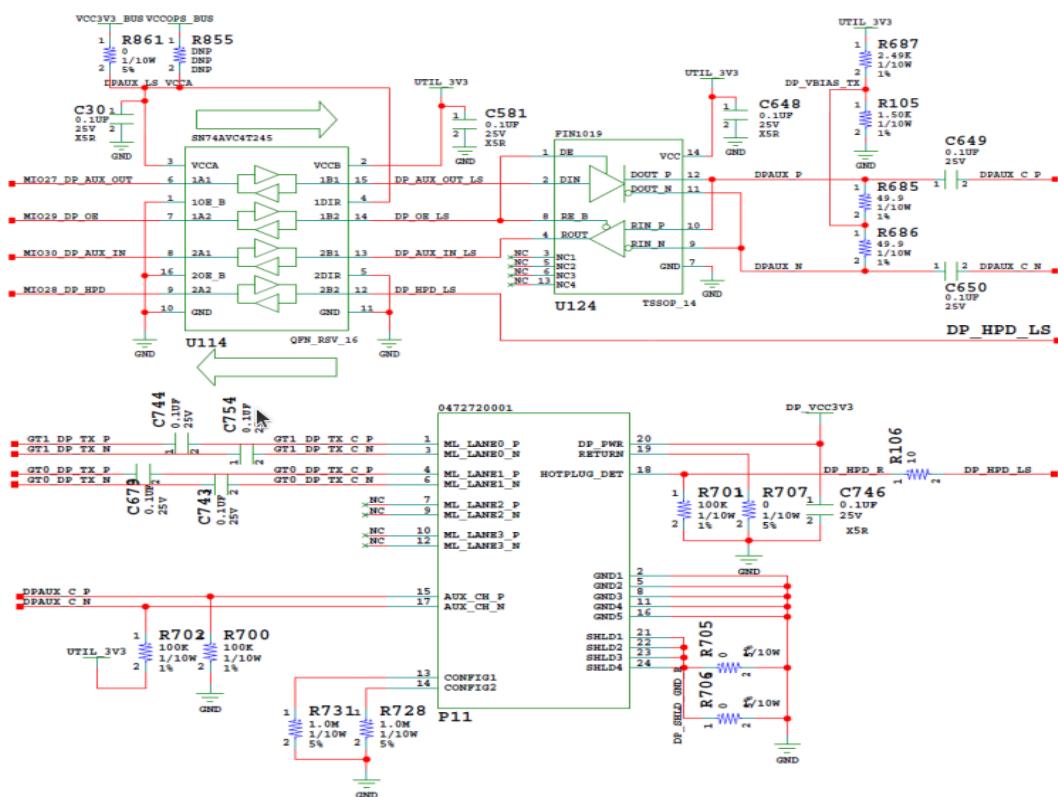
Thiết bị		Loại kết nối	Mô tả
Tên chân	Số chân		
LED_2	61	S, I/O, PD	Chân này biểu thị hoạt động nhận hoặc truyền dữ liệu.
LED_1	62	S, I/O, PD	Chân này cho biết liên kết 100BASE-T đã được thiết lập.
LED_0	63	S, I/O, PD	Chân này cho biết tất cả liên kết đã được thiết lập.

3.3.2.4. Khôi hiển thị VESA (DPAUX MIO 27 -30) [1]

Zynq UltraScale+ MPSoC cung cấp bộ điều khiển VESA DisplayPort 1.2 hỗ trợ tối đa hai luồng dữ liệu liên kết chính ở tốc độ 1.62Gb/s, 2.70Gb/s hoặc 5.40Gb/s. Chuẩn DisplayPort xác định kênh phụ trợ (auxiliary channel) sử dụng tín hiệu LVDS ở tốc độ dữ liệu 1 Mb/s, được dịch từ tín hiệu MIO một đầu sang kênh DisplayPort Aux khác, DPAUX (xem Bảng 3.8) và sơ đồ nguyên lý mạch DisplayPort được hiển thị trong Hình 3.24.

Bảng 3.14 Kết nối giữa DPAUX

XCZU9EG (U1)	Tên chân	Level Shifter (U114)	
		Số chân	Tên chân
L21	MIO30_DP_AUX_IN	8	2A1
K22	MIO29_DP_OE	7	1A2
N21	MIO28_DP_HPD	9	2A2
M21	MIO27_DP_AUX_OUT	6	1A1

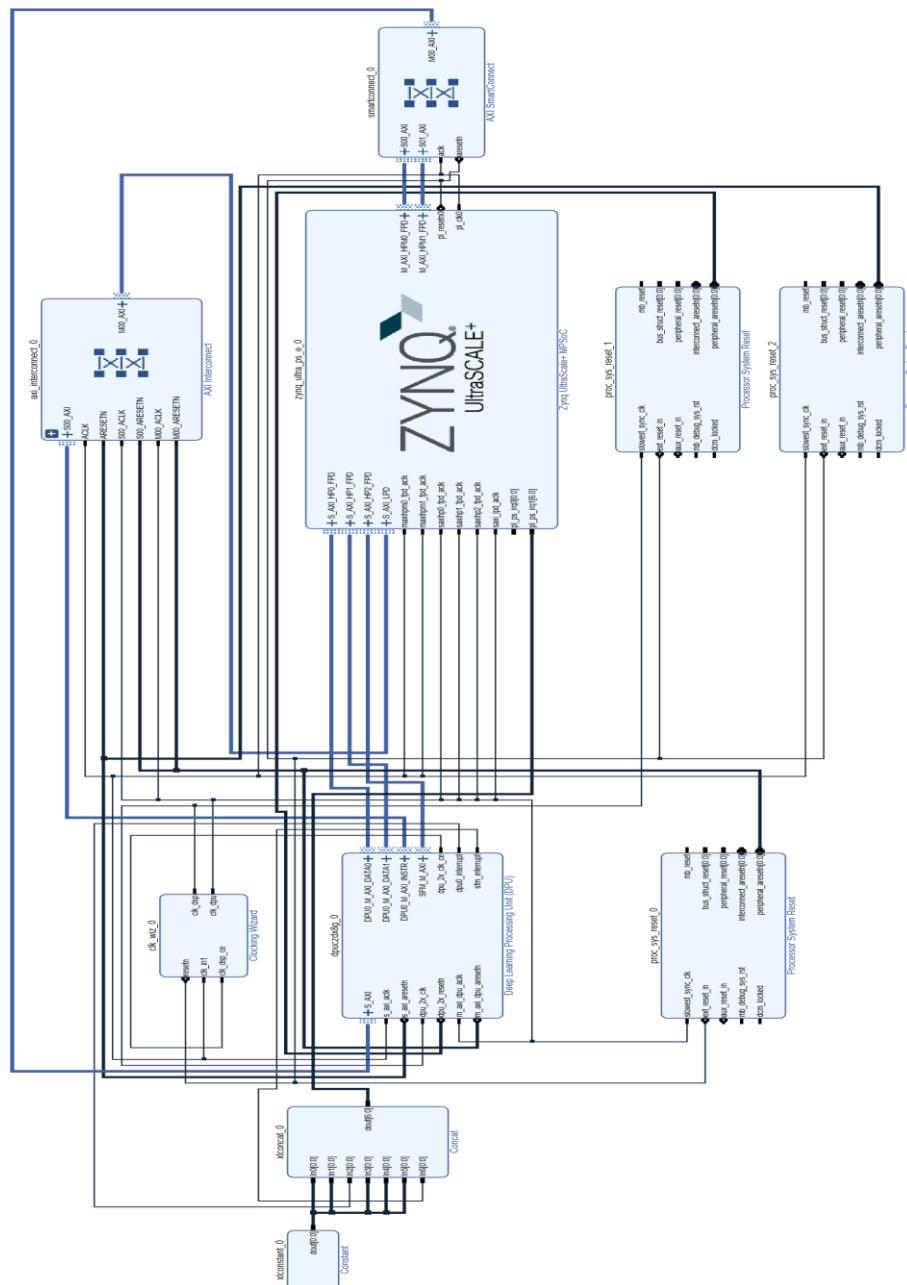
**Hình 3.27. Sơ đồ kết nối của khối hiển thị VESA**

Các tín hiệu từ Zynq Ultrascale+ MPSOC - ZU102 được truyền lên màn hình quan sát. Tuy nhiên, mỗi màn hình quan sát đều có mức điện áp khác nhau tùy thuộc nhà sản xuất. Do đó, các tín hiệu cần được hiển thị sẽ đi qua IC Level Shifter SN74LVC8T245 (U114). IC này có chức năng chuyển đổi mức điện áp từ ZCU102 sao cho phù hợp. Ngoài ra IC này còn có khả năng truyền dẫn dòng điện cao lên đến 35mA nên cho phép kết nối trực tiếp với nhiều thiết bị mà không cần bộ khuếch đại. Cuối cùng, các tín hiệu này sẽ đi qua FIN1019 (U124), IC này có chức năng lọc nhiễu nguồn, lọc nhiễu tín hiệu và lọc nhiễu tín hiệu clock.

CHƯƠNG 4: KẾT QUẢ

4.1. Kết nối IP đầy đủ

Hình 4.1 dưới đây mô tả toàn bộ kết nối IP của hệ thống gồm các IP chính như Zynq Ultrascale+ MPSoC - ZCU102, Deep Learning Processor Unit (DPU), Clock Wiward, Processor System Reset.



Hình 4. 1. Sơ đồ các kết nối IP đầy đủ của hệ thống

Hệ thống các lõi IP được thiết kế với 4 lớp mạng (Network) được kết nối giữa Deep Learning Processor Unit (DPU) (Master) với Zynq Ultrascale+ MPSOC (Slave) bằng chuẩn giao diện AXI.

Sau khi quá trình tổng hợp (Synthesis) kết thúc, các lõi IP được chuyển thành các phần tử logic mà FPGA có thể hiểu được. Các thông tin về các phần tử này được mô tả chi tiết trong bảng 4.1 dưới đây:

Bảng 4.1. Tài nguyên của các khối IP khi tổng hợp

Constraints	LUT	FF	BRAM	URAM	DSP
DPU_zynq_ultra_ps_e_0_0	264	0	0	0	0
DPU_clk_wiz_0_0	1	0	0	0	0
DPU_proc_sys_reset_0_0	19	40	0	0	0
DPU_proc_sys_reset_0_1	24	40	0	0	0
DPU_dpuczdx8g_0_0	61893	107918	259	0	724

Thành phần PS của ZCU102: Mức sử dụng tài nguyên là 264 LUT. Cho thấy khối IP này chủ yếu sử dụng các logic tổ hợp (combinational logic) mà không yêu cầu quá nhiều các yếu tố lưu trữ như FF hoặc bộ nhớ.

Khối IP Clock Wizard: DPU_clk_wiz_0_0: Mức sử dụng tài nguyên là 1 LUT.

Khối IP Processing System Reset: Mức sử dụng tài nguyên lần lượt là: 19 LUT, 40 FF và 24 LUT, 40 FF.

Khối IP Deep Leanring Processor Unit: Mức sử dụng tài nguyên là 61893 LUT, 107918 FF, 259 BRAM, 724 DSP. IP này cần sử dụng nhiều tài nguyên để thực hiện các phép tính phức tạp liên quan đến mạng học sâu.

Bảng 4.2. Thông số các giao diện và đường địa chỉ thiết kế IP học sâu

Chương 4. Kết quả

Name	Interface	Slave Segment	Master Base Address	Phạm vi địa chỉ	Master High Address
Network 0/dpuczdx8g_0/DPU_M_AXI_DATA0					
zynq_ultra_ps_e_0/SAXIGP2	S_AXI_HP0_FPD	HP0_LPS_OCM	0x00_FF00_0000	16M	0x00_FFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP2	S_AXI_HP0_FPD	HP0_QSPI	0x00_C000_0000	512	0x00_DFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP2	S_AXI_HP0_FPD	HP0_DDR_LOW	0x00_0000_0000	2G	0x00_7FFF_FFFF
zynq_ultra_ps_e_0/SAXIGP2	S_AXI_HP0_FPD	HP0_PCIE_LOW	0x00_E000_0000	256M	0x00_EFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP2	S_AXI_HP0_FPD	HP0_DDR_HIGH	0x08_0000_0000	32G	0x0F_FFFF_FFFF
Network 1/dpuczdx8g_0/DPU_M_AXI_DATA0					
zynq_ultra_ps_e_0/SAXIGP3	S_AXI_HP1_FPD	HP1_DDR_HIGH	0x08_0000_0000	32G	0x0F_FFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP3	S_AXI_HP1_FPD	HP1_QSPI	0x00_C000_0000	512	0x00_DFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP3	S_AXI_HP1_FPD	HP1_PCIE_LOW	0x00_E000_0000	256M	0x00_EFFF_FFFF

zynq_ultra_ps_e_0/SAXIGP3	S_AXI_HP1_FPD	HP1_LPS_OCM	0x00_FF00_0000	16M	0x00_FFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP3	S_AXI_HP1_FPD	HP0_DDR_LOW	0x00_0000_0000	2G	0x0F_7FFF_FFFF
Network 3/dpuczdx8g_0/DPU_M_AXI_INSTR					
zynq_ultra_ps_e_0/SAXIGP4	S_AXI_HP2_FPD	HP2_DDR_LOW	0x00_0000_0000	2G	0x00_7FFF_FFFF
zynq_ultra_ps_e_0/SAXIGP4	S_AXI_HP2_FPD	HP2_QSPI	0x00_C000_0000	512M	0x00_DFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP4	S_AXI_HP2_FPD	HP2_PCIE_LOW	0x00_E000_0000	256M	0x00_EFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP4	S_AXI_HP2_FPD	HP2_LPS_OCM	0x00_FF00_0000	16M	0x00_FFFF_FFFF
zynq_ultra_ps_e_0/SAXIGP4	S_AXI_HP2_FPD	HP2_DDR_HIGH	0x08_0000_0000	32G	0X0F_FFFF_FFFF
Network 4/zynq_ultra_ps_e_0/Data					
dpuczdx8g_0/S_AXI	S_AXI	reg0	0x04_0000_0000	16M	0x04_00FF_FFFF
dpuczdx8g_0/S_AXI	S_AXI	reg0	0x04_0000_0000	16M	0x04_00FF_FFFF

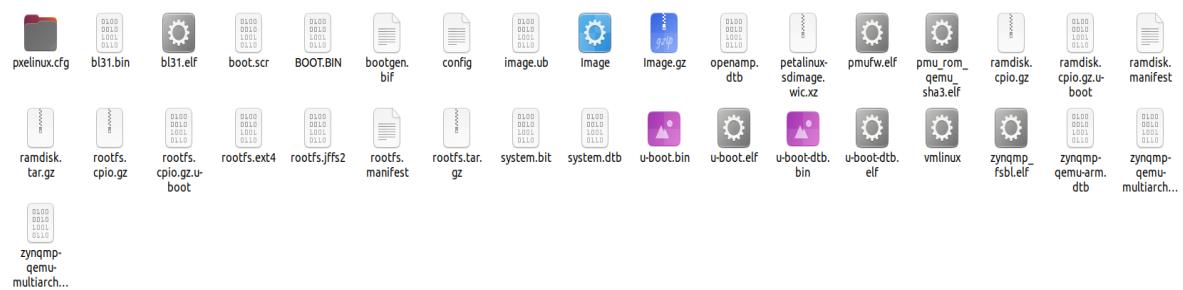
Bảng 4.2 ở trên mô tả chi tiết các đường địa chỉ của các lớp mạng được sử dụng. Trong trường hợp này sử dụng 4 mạng lưới khác nhau (Network 0, 1, 2, 3), mỗi mạng lưới chứa các IP blocks khác nhau.

Các thông số như : Interface mô tả giao diện tương tác của IP block, Slave Segment chỉ ra chức năng của IP block (ví dụ: HP0_PCIE_LOW,HP1_LPS_OCM). Master Base Address và Range xác định vùng địa chỉ mà khối IP có thể truy cập Cuối cùng là Master High Address cho biết giới hạn trên của dải địa chỉ.

Sau khi thực hiện các bước đóng gói (Wrapper) và kiểm tra (Validate) các kết nối của hệ thống, tiến hành tổng hợp (Synthesis) , triển khai (Implementation) và tạo bitstream. Cuối cùng là tạo file nền tảng phần cứng (hardware platform) có đuôi .xsa chứa các thông tin vừa tạo để tiến hành cho bước tiếp theo là xây dựng hệ thống nhúng cho ZCU102.

4.2. Các tập tin xây dựng hệ thống nhúng

Kết quả của quá trình xây dựng nhân chạy hệ điều hành Linux (Build Kernel) chứa nền tảng phần cứng được tạo từ Vivado là các file dùng để khởi động (boot.src, BOOT.BIN,...) và các file cấu hình (pxelinux.cfg, rootfs.tar,...).



Hình 4. 2. Các tập tin được tạo sau khi build kernel

Các tập tin này được ghi vào thẻ SD (lớn hơn 16GB) và gắn vào J100 trên ZCU102 để khởi động. Quá trình khởi động được mô tả ở hình 4.3 dưới đây:



Hình 4.3. ZCU102 vừa bật nguồn



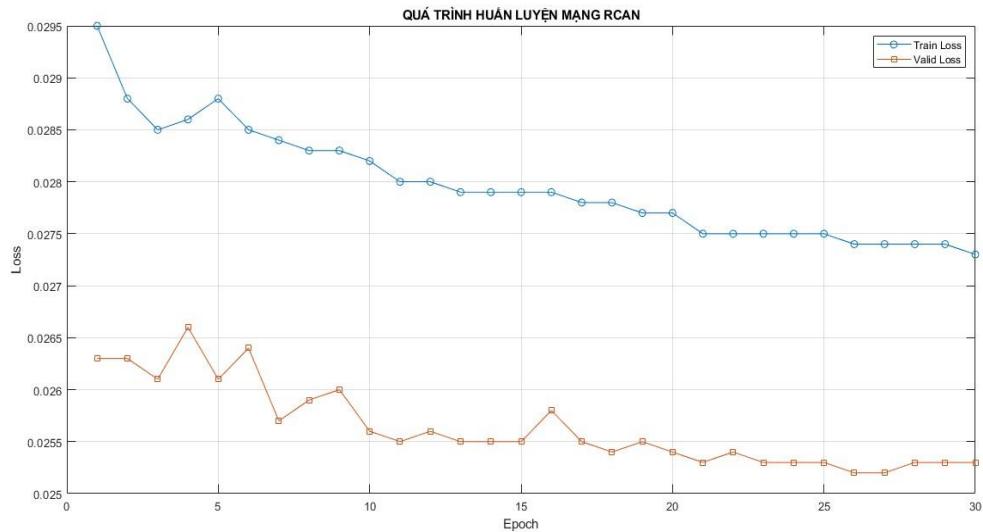
Hình 4.4. ZCU102 khởi động hoàn tất

Khi bật nguồn, ZCU102 sẽ lấy các tập tin cần thiết từ thẻ SD đã được cắm vào J100 để tiến hành khởi động, lúc này đèn led báo trạng thái INT_B hiển thị màu đỏ. Sau khi khởi động thành công, led INT_B chuyển thành màu xanh, chứng tỏ ZCU102 đã sẵn sàng.

4.3. Triển khai mô hình siêu phân giải ảnh

4.3.1. Huấn luyện

Sử dụng tập dữ liệu “*benchmark*” để tiến hành chạy mô hình siêu phân giải ảnh. Tập dữ liệu này gồm có 4 tập dữ liệu con lần lượt là B100, Set14, Set5, Urban100. Tập dữ liệu Set14 gồm có 14 tấm ảnh có dung lượng từ 100KB đến 800KB và có kích thước dao động từ 300 đến 800 pixels. Tập dữ liệu Set5 gồm có 5 tấm ảnh có dung lượng từ 100KB đến 300KB và có kích thước dao động từ 280 đến 500 pixels. Cuối cùng là. Các tập dữ liệu này gồm có ảnh gốc có độ phân giải cao và ảnh có độ phân giải thấp được tạo ra bằng phép nội suy bicubic với các scale khác nhau.

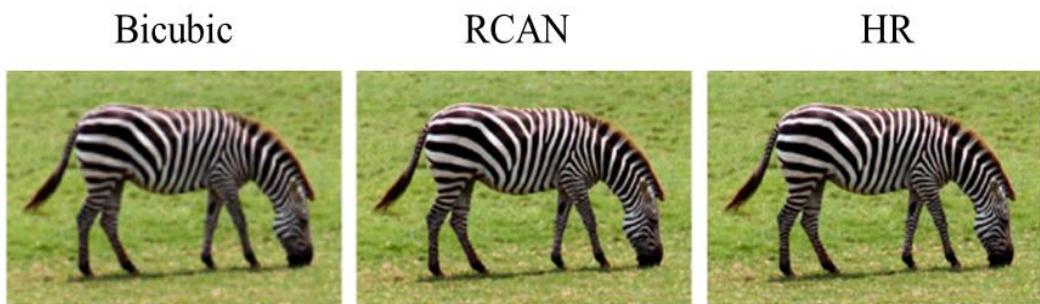


Hình 4.5. Biểu đồ giá trị mất mát trên 30 epoch của quá trình huấn luyện mạng siêu phân giải RCAN

Dựa vào hình 4.5. giá trị mất mát (Loss) của tập dữ liệu Train và tập dữ liệu Valid đều giảm dần theo từng epoch. Sự chênh lệch giữa Train Loss và Valid Loss không lớn, chứng tỏ mô hình không bị quá khớp (Overfitting) với tập dữ liệu huấn luyện.

4.3.2. Chạy mạng siêu phân giải ảnh trên phần cứng

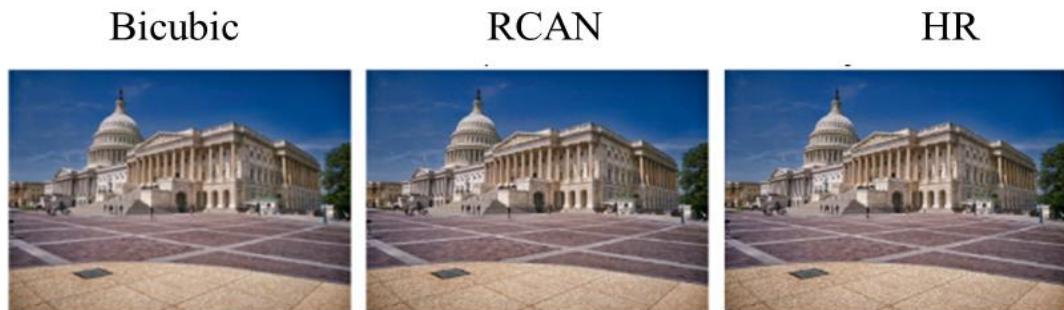
Tiến hành chạy mô hình RCAN đã được huấn luyện trên trên Zynq Ultrascale+ MPSoC - ZCU102.



Hình 4.6. Kết quả siêu phân giải ảnh 253027.png của tập dữ liệu B100

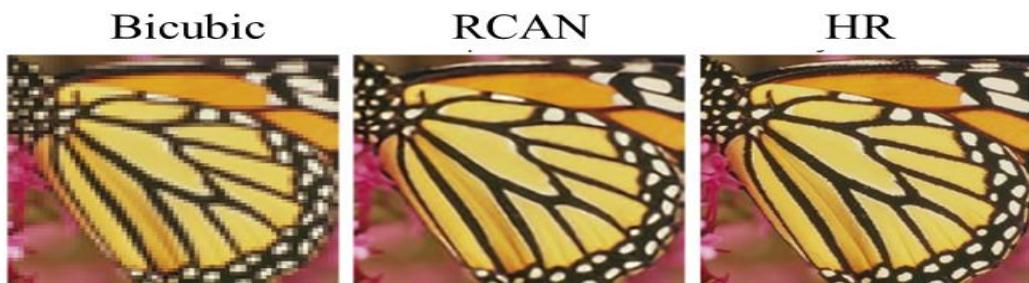
Chương 4. Kết quả

Ảnh 253027.png được lấy từ tập dữ liệu B100. Tập dữ liệu B100 gồm có 100 tấm ảnh có dung lượng từ 150 KB đến 300KB và có kích thước dao động từ 300 đến 500 pixels.



Hình 4.7. Kết quả siêu phân giải ảnh img070.png của tập dữ liệu Urban100

Ảnh img070.png được lấy từ tập dữ liệu Urban100. Tập dữ liệu Urban100 gồm có 100 tấm ảnh có dung lượng từ 900KB đến 1.2MB và có kích thước dao động từ 600 đến 1200 pixels.



Hình 4.8. Kết quả siêu phân giải ảnh butterfly.png của tập dữ liệu Set5

Ảnh butterfly được lấy từ tập dữ liệu Set5. Tập dữ liệu Set5 gồm có 5 tấm ảnh có dung lượng từ 100KB đến 300 KB và có kích thước dao động từ 256 đến 512 pixels.

Cuối cùng là ảnh coastguard được lấy từ tập dữ liệu Set14. Tập dữ liệu Set14 gồm có 14 tấm ảnh có dung lượng từ 100KB đến 800 KB và có kích thước dao động từ 256 đến 712 pixels.



Hình 4.9. Kết quả siêu phân giải ảnh coastguard.png của tập dữ liệu Set14

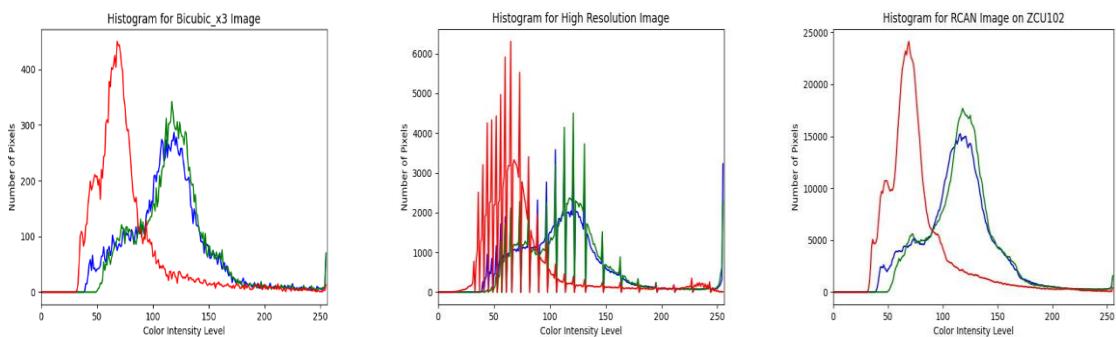
Hình 4.6, 4.7, 4.8, 4.9 là kết quả của quá trình chạy mô hình siêu phân giải RCAN trên phần cứng Zynq Ultrascale+ MPSoC – ZCU102. Ở mỗi bức hình gồm có 3 thông tin: Bicubic, RCAN và HR. Bicubic là ảnh dùng phép nội suy Bicubic với scale = 3, đây là ảnh chất lượng thấp (LR). HR (High Resolution) là ảnh gốc có chất lượng cao. Cuối cùng là RCAN, đây là ảnh được tạo của quá trình chạy mô hình siêu phân giải ảnh RCAN trên thành phần PL của Zynq Ultrascale+ MPSoC – ZCU102. Sự khác nhau về kích thước và dung lượng được mô tả trong bảng 4.3 dưới đây:

Bảng 4.3. Kích thước và dung lượng ảnh

		Bicubic x3	HR	RCAN
253027	Kích thước (pixels)	160×107	481×321	1280×720
	Dung lượng	34.7 KB	290 KB	259 KB
img070	Kích thước (pixels)	341×256	1024×769	1280×720
	Dung lượng	160 KB	1.47 MB	1.02 MB
butterfly	Kích thước (pixels)	85 × 85	256×256	1280×720
	Dung lượng	16.9 KB	124 KB	101KB
coastguard	Kích thước (pixels)	117×96	352×288	1280×720
	Dung lượng (KB)	19.6KB	150 KB	89.8 KB

Dựa vào bảng 4.3 ở trên, kích thước của tất cả các ảnh đều được xử lý về mức 1280×720 pixel. Kích thước này là cố định, nhưng có thể điều chỉnh trong bước tiền xử lý hệ thống siêu phân giải ảnh được giới thiệu ở chương 3. Dung lượng của các bức ảnh được cải thiện gấp 5 đến 8 lần của ảnh Bicubic.

Hình 4.10, hình 4.11, hình 4.12, hình 4.13 bên dưới lần lượt là biểu đồ histogram của bốn ảnh ở bốn tập dữ liệu trên bảng 4.3.



Hình 4.10. Biểu đồ histogram ảnh 253027 dùng bicubic_x3

Hình 4.11. Biểu đồ histogram ảnh HR 253027

Hình 4.12. Biểu đồ histogram ảnh 253027 dùng RCAN

Hình 4.10, 4.11 và 4.12 lần lượt biểu diễn sự phân bố cường độ màu của các kênh màu đỏ, xanh lá, và xanh dương trong từng ảnh bicubic_x3, HR và RCAN của ảnh 253027 trong tập dữ liệu B100.

➤ **Biểu đồ histogram của ảnh 253027 dùng phép nội suy bicubic_x3**

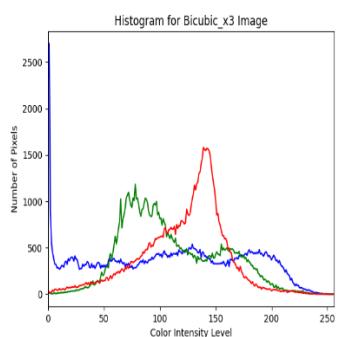
- Kênh đỏ: Đỉnh cường độ cao nhất nằm trong khoảng 30-60, cho thấy rằng có rất nhiều pixels có giá trị cường độ màu đỏ ở mức thấp. Cường độ màu đỏ giảm mạnh sau khoảng 60 và rất ít pixel có cường độ màu đỏ ở mức cao.
- Kênh xanh lá và xanh dương: Cả hai kênh này có sự phân bố tương đối giống nhau, với các đỉnh cường độ màu tập trung trong khoảng 80-120. Các giá trị cường độ màu cao hơn cũng có một số lượng pixel nhất định, nhưng giảm dần sau khoảng 150.

➤ **Biểu đồ histogram của ảnh gốc HR 253027**

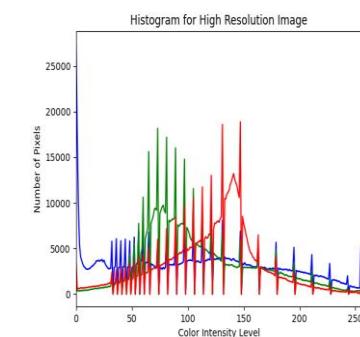
- Kênh đỏ: Đỉnh cường độ rất rõ ràng và cao, tập trung ở các giá trị trong khoảng 30-60.
- Kênh xanh lá: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 80-120, với một số đỉnh cao hơn ở khoảng 100-150.
- Kênh xanh dương: Tương tự như kênh xanh lá, có một số đỉnh tập trung ở khoảng 80-120. Nhưng số lượng pixels lại thấp hơn kênh đỏ và xanh với đỉnh khoảng 2000.

➤ **Biểu đồ histogram ảnh 253027 mạng siêu phân giải RCAN trên ZCU102**

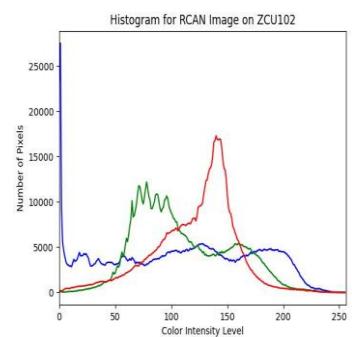
- Kênh đỏ: Đỉnh cường độ cao nhất nằm trong khoảng 30-60, tương tự như ảnh Bicubic nhưng với số lượng pixel cao hơn. Cường độ màu đỏ giảm mạnh sau khoảng 60 và rất ít pixel có cường độ màu đỏ ở mức cao, nhưng vẫn nhiều hơn so với ảnh Bicubic.
- Kênh xanh lá và xanh dương: Các đỉnh cường độ màu tương tự như ảnh Bicubic, tập trung ở khoảng 80-120. Tuy nhiên, số lượng pixel có cường độ màu cao hơn (khoảng 150-200) cũng nhiều hơn, cho thấy sự cải thiện về chi tiết màu sắc.



Hình 4.13. Biểu đồ histogram ảnh img070 dùng bicubic_x3



Hình 4.14. Biểu đồ histogram ảnh HR



Hình 4.15. Biểu đồ histogram ảnh img070 dùng RCAN

Hình 4.13, 4.14 và 4.15 lần lượt biểu diễn sự phân bố cường độ màu của các kênh màu đỏ, xanh lá, và xanh dương trong từng ảnh bicubic_x3, HR và RCAN của ảnh img070 trong tập dữ liệu Urban100.

➤ Biểu đồ histogram của ảnh img070 dùng phép nội suy bicubic_x3

- Kênh đỏ: Cường độ màu đỏ tăng dần từ 0 đến 100 và tăng mạnh đến 150. Đỉnh cường độ cao nhất nằm trong khoảng 130-150, cho thấy rằng có rất nhiều pixels có giá trị cường độ màu đỏ ở mức trung bình. Cường độ màu đỏ giảm mạnh sau khoảng 150 và rất ít pixel có cường độ màu đỏ ở mức cao.
- Kênh xanh lá: Kênh này có sự phân bố tương đối đồng đều, với các đỉnh cường độ màu tập trung trong khoảng 50 – 100 và 130 – 160. Chú ý kên xanh lá được phân bố đều trên bức ảnh.
- Kênh xanh dương: Có số lượng pixels khoảng 400 – 500 và được phân bố đều trên toàn trực cường độ.

➤ Biểu đồ histogram của ảnh gốc HR img070

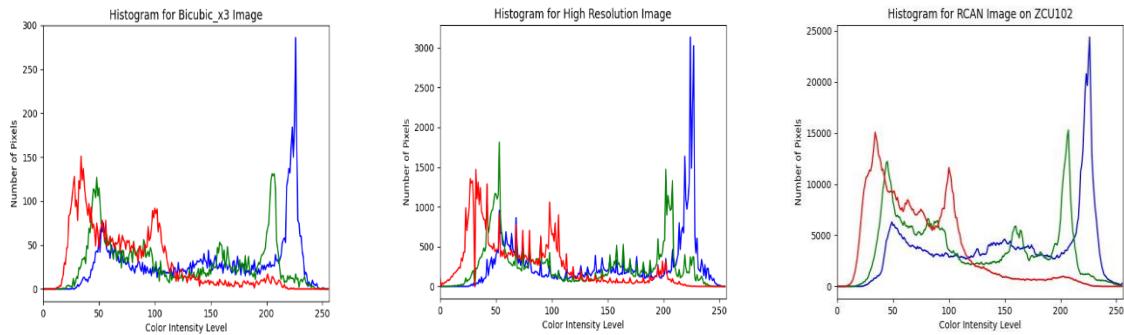
- Kênh đỏ: Đỉnh cường độ rất rõ ràng và cao, đỉnh cao nhất tại 130 và 150 với số lượng pixels là 19000, tập trung ở các giá trị trong khoảng 50 - 170.
- Kênh xanh lá: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 50 - 120, với một số đỉnh có số lượng pixels ở mức 17000. Chứng tỏ kên xanh nằm ở vùng tối.
- Kênh xanh dương: : Có số lượng pixels khoảng 6000 và được phân bố đều trên toàn trực cường độ.

➤ Biểu đồ histogram ảnh img070 dùng mạng siêu phân giải RCAN chạy trên ZCU102

- Kênh đỏ: Đỉnh cường độ cao nhất nằm trong khoảng 100 - 170, tương tự như ảnh Bicubic nhưng với số lượng pixel cao hơn có số lượng pixels ở đỉnh khoảng 17500.
- Kênh xanh lá: Các đỉnh cường độ màu tương tự như ảnh Bicubic, tập trung ở khoảng 60 – 120 và 150 – 170, dải phân bố của kên xanh lá đã dịch sang

phải. Số lượng pixel trong khoảng này là 12500, cho thấy sự cải thiện về chi tiết màu sắc.

- Kênh xanh dương: Các đỉnh cường độ màu được phân bố đều trên toàn trực cường độ, nhưng có số lượng pixels lớn, dao động từ 3000 – 5000.



Hình 4.16. Biểu đồ histogram ảnh butterfly dùng bicubic_x3

Hình 4.17. Biểu đồ histogram ảnh HR butterfly

Hình 4.18. Biểu đồ histogram ảnh butterfly dùng RCAN

Hình 4.16, 4.17 và 4.18 lần lượt biểu diễn sự phân bố cường độ màu của các kênh màu đỏ, xanh lá, và xanh dương trong từng ảnh bicubic_x3, HR và RCAN của ảnh butterfly trong tập dữ liệu Set5.

➤ Biểu đồ histogram của ảnh butterfly dùng phép nội suy bicubic_x3

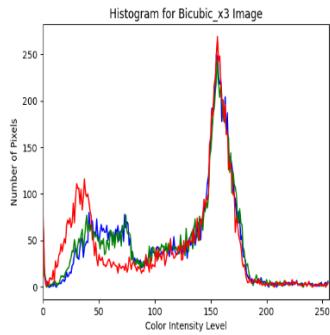
- Kênh đỏ: Cường độ kênh đỏ tập trung ở mức thấp (bên trái), chủ yếu trong khoảng 30 - 110. Đỉnh cường độ cao nhất nằm trong khoảng 35 - 45, cho thấy rằng có rất nhiều pixels có giá trị cường độ màu đỏ ở mức thấp. Cường độ màu đỏ giảm mạnh sau khoảng 115 và rất ít pixel có cường độ màu đỏ ở mức cao.
- Kênh xanh lá: Kênh này có sự phân bố tương đối đồng đều, với 2 đỉnh cao nhất tập trung lần lượt tại 50 và 200. Với số lượng pixels tại các vị trí này khoảng 125. Chú ý kênh xanh lá được phân bố đều trên bức ảnh.
- Kênh xanh dương: Tập trung chủ yếu ở mức cao (bên phải). Với số lượng pixels tại đỉnh là 270.

➤ **Biểu đồ histogram của ảnh gốc HR butterfly**

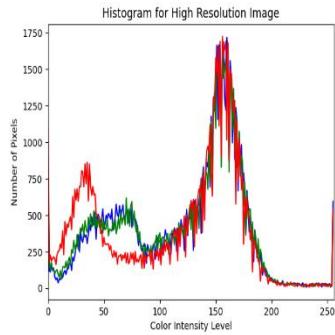
- Kênh đỏ: Đỉnh cường độ tương đối rõ ràng, đỉnh cao nhất tại 30 với số lượng pixels là 1450.
- Kênh xanh lá: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 30 – 100 và 150 - 220, với một số đỉnh có số lượng pixels ở mức 1800. Chứng tỏ kênh xanh được phân bố đều.
- Kênh xanh dương: : Có số lượng pixels tại đỉnh khoảng 3200 và được phân bố lệnh về bên phải trực cường độ.

➤ **Biểu đồ histogram ảnh butterfly dùng mạng siêu phân giải RCAN chạy trên ZCU102**

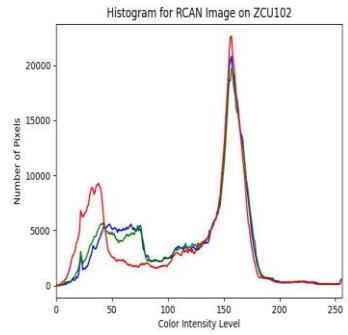
- Kênh đỏ: Đỉnh cường độ cao nhất nằm trong khoảng 40 - 120, tương tự như ảnh Bicubic nhưng với số lượng pixel rất cao. Số lượng pixels ở đỉnh khoảng 17500 (cao hơn ảnh gốc HR).
- Kênh xanh lá: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 40 – 120 và 140 - 230. Dải phân bố số lượng pixels của kênh xanh lá đã dịch sang phải. Số lượng pixel tại đỉnh là 15000, cho thấy sự cải thiện về chi tiết màu sắc.
- Kênh xanh dương: Các đỉnh cường độ màu được phân bố lệch về bên phải trên trực cường độ, nhưng có số lượng pixels rất cao, tại đỉnh đạt tới 24000.



Hình 4.19. Biểu đồ histogram ảnh coastguard dùng bicubic_x3



Hình 4.20. Biểu đồ histogram ảnh HR coastguard



Hình 4.21. Biểu đồ histogram ảnh coastguard dùng RCAN

Hình 4.19, 4.20 và 4.21 lần lượt biểu diễn sự phân bố cường độ màu của các kênh màu đỏ, xanh lá, và xanh dương trong từng ảnh bicubic_x3, HR và RCAN của ảnh coastguard trong tập dữ liệu Set14.

➤ **Biểu đồ histogram của ảnh coastguard dùng phép nội suy bicubic_x3**

- Kênh đỏ: Cường độ kênh đỏ tập trung chủ yếu trong khoảng 20 - 40 và từ 150 - 170. Đỉnh cường độ cao nhất nằm trong khoảng 252. Cường độ màu đỏ giảm mạnh sau khoảng 165.
- Kênh xanh lá và xanh dương: Hai kênh này có sự phân bố tương đối giống nhau và giống với kênh đỏ, cường độ kênh tập trung lần lượt tại 40 – 70 và 150 - 170. Với số lượng pixels tại đỉnh khoảng 250 (gần bằng kênh đỏ).

➤ **Biểu đồ histogram của ảnh gốc HR coastguard**

- Kênh đỏ: Đỉnh cường độ tương đối rõ ràng, đỉnh cao nhất tại 155 với số lượng pixels là 1700.
- Kênh xanh lá và xanh dương: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 30 – 75 và 140 - 160, với một số đỉnh có số lượng pixels ở mức 1700 (bằng với kênh đỏ).

➤ **Biểu đồ histogram ảnh coastguard dùng mạng siêu phân giải RCAN chạy trên ZCU102**

- Kênh đỏ: Đỉnh cường độ cao nhất nằm trong khoảng 20 – 50 và 150 - 180, tương tự như ảnh Bicubic nhưng với số lượng pixel rất cao. Số lượng pixels ở đỉnh khoảng 23500 (cao hơn ảnh gốc HR).
- Kênh xanh lá và xanh dương: Các đỉnh cường độ cũng tương đối rõ ràng, tập trung ở các giá trị khoảng 40 – 80 và 130 - 180. Dải phân bố số lượng pixels của kênh xanh lá và xanh dương đã dịch sang phải. Số lượng pixel tại đỉnh là 19000, cho thấy sự cải thiện về chi tiết màu sắc.

Dựa vào lược đồ histogram và so sánh kích thước và dung lượng ảnh, ảnh được tạo bằng mạng siêu phân giải RCAN trên ZCU102 có chất lượng tốt trên các tập dữ liệu Set5, Set14 và 100. Đặc điểm chung là dung lượng các ảnh trong những tập dữ liệu này đều ở mức thấp từ 100 KB đến 700KB và có kích thước nhỏ hơn 800x800 pixels. Đối với tập dữ liệu Urban100 với ảnh có dung lượng lớn ($\geq 1\text{MB}$) thì ảnh tạo ra không được hiệu suất cao.

4.3.3. Chạy mạng siêu phân giải ảnh trên Cloud (Kaggle)

Bảng 4.1 dưới đây sẽ hiển thị các thông số so sánh ảnh một cách trực quan về chất lượng của ảnh chạy trên cloud (Kagggle) và ZCU102.

Bảng 4.4. Thông số PSNR và SSIM của ảnh

Tên ảnh	Tập dữ liệu	Cloud (Kagggle)		ZCU102	
		PSNR	SSIM	PSNR	SSIM
253027	B100	25.76	0.70	26.17	0.74
img070	Urban100	23.34	0.74	23.82	0.80
butterfly	Set5	29.04	0.85	32.64	0.91
coastguard	Set14	25.82	0.73	28.87	0.79

Bảng 4.4 cho thấy các giá trị PSNR và SSIM của ảnh siêu phân giải RCAN thực hiện trên phần cứng Zynq Ultrascale+ MPSoC – ZCU102 có giá trị lớn hơn chạy trên cloud. Cho thấy ảnh siêu phân giải RCAN thực nghiệm trên phần cứng có chất lượng tốt hơn chạy trên Kaggle.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Đồ án đã đạt được một số thành công nhất định. Thành công tạo được nền tảng phần cứng từ các IP, xây dựng được hệ điều hành Linux và triển khai mạng siêu phân giải RCAN chạy trên ZCU102. Dựa vào kết quả đạt được ở chương 4, đồ án đã cung cấp các thông tin cần thiết chứng tỏ cho việc triển khai thuật toán mạng học sâu xử lý hình ảnh trên nền tảng phần cứng cụ thể là Zynq Ultrascale+ MPSoC tốt hơn trên nền tảng phần mềm. Các chỉ số thu được như PSNR, SSIM, lược đồ histogram trong chương trước có thể chứng tỏ được hình ảnh đầu ra trên nền tảng Kaggle có chất lượng thấp hơn trên ZCU102.

Tuy nhiên, đồ án vẫn còn một số điểm cần phải cải thiện. Việc huấn luyện mạng trên máy ảo Ubuntu chạy hệ điều hành Linux gặp lỗi và không thành công nên phải huấn luyện trên môi trường Kaggle. Ngoài ra, ảnh siêu phân giải RCAN chỉ tạo ra một kích thước nhất định là 720x1280. Ảnh siêu phân giải được tạo ra có chất lượng càng cao khi ảnh có dung lượng càng thấp. Do đó, mô hình chỉ hợp lý cho các hình ảnh có dung lượng thấp, khi chạy mô hình cho ảnh có dung lượng lớn, thì ảnh đầu ra sẽ không đạt được kết quả mong muốn.

5.2. Hướng phát triển

Hệ thống siêu phân giải ảnh RCAN cần được tối ưu và hoàn thiện hơn nữa. Đầu tiên, cần phải cải thiện chất lượng ảnh ngõ ra theo ý muốn. Giúp mô hình linh hoạt hơn trong thực tiễn. Triển khai các mạng siêu phân giải khác như GAN, VDSR, ... và so sánh với RCAN, từ đó suy ra phương pháp nào tối ưu nhất.

Phần cứng Zynq Ultrascale+ MPSoC - ZCU102 là một SoC mạnh mẽ với nhiều tiềm năng phát triển. Ngoài việc ứng dụng Deep Learning cho ZCU102, có thể ứng dụng cho nhiều lĩnh vực khác nhau như điều khiển các thiết bị ngoại vi bằng các chuẩn giao tiếp CAN, SPI,... Ứng dụng trong IoT cho việc nhận các giá trị từ cảm biến, hoặc trong các bài toán siêu cao tần,....

TÀI LIỆU THAM KHẢO

- [1] N. H. N. Uyen, "Implementation Of Sobel Filter Base On The Zynq-7000 SoC Development Board," Ho Chi Minh, 2022.
- [2] G. K. Ravi, "Deploying Deep learning Image Super-Resolution Models in Xilinx Zynq MPSoC ZCU102," 2020.
- [3] C. C. L. K. H. X. T. Chao Dong, "Learning a Deep Convolutional Network for Image Super-Resolution," 2014.
- [4] Jiwon Kim, Jung Kwon Lee, Kyoung mu Lee, "Accurate Image Super-Resolutioon Using Very Deep Convolutional Networks," 2016.
- [5] Jwon Kim, Jung Kwon Lee, Kyoung Mu Lee, "Deep-Recursive Convolutional Network for Image Super-Resolution," 2016.
- [6] Yulun Zhang, Kunpeng Li, Kai li, Lichen Wang, Bineng Zhong, Yun Fu, "Image Super-Resolution Using Very Deep Residual Channel Attention Networks," 2018.
- [7] Louise H. Crockett, David Northcote, Craig Ramsay, Fraser D. Robinson, Robert W. Stewart, Exploring Zynq® MPSoC With PYNQ and Machine Learning Applications, Scotland, 2019.
- [8] Xilinx, ZCU102 Evaluation Board User Guide (Ug1182) (v.17), 2023.
- [9] ARM, Learning the architecture - An introduction to AMBA AXI (v3.0), 2022.

- [10] Xilinx, DPUCZDX8G for Zynq UltraScale+ MPSoCs - PG338 (v4.1), 2023.
- [11] Xilinx, Zynq Ultrascale+ MPSoC Processing System v3.5 (PG201), 2023.
- [12] Xilinx, Clocking WIzard v6.0 (PG065), 2022.
- [13] Xilinx, "Processor System Reset Module v5.0 (PG164)," 2015, p. 13.
- [14] Xilinx and University of Strathclyde Glasgow, The Zynq Book: Embedded Processing with the ARM CortexA9 on the Xilinx Zynq-7000 All Programmable SoC, UK: Louise H. Crockett; Ross A. Elliot; Martin A. Enderwitz; Robert W. Stewart;, 2014.

PHỤ LỤC

Code chương trình mạng siêu phân giải RCAN, code vẽ biểu đồ histogram và hướng dẫn các bước chạy mô hình siêu phân giải ảnh RCAN trên Zynq Ultrascale+ MPSoC – ZCU102 nằm trong link Github được cung cấp bên dưới.

<https://github.com/phuoc0402/Residual-Channel-Attention-Networks-based-on-FPGA.git>

Nang cao do phan giao anh bang tri tue nhan tao ZCU102

BÁO CÁO ĐỘC SẮC

29%

CHỈ SỐ TƯƠNG ĐỒNG

26%

NGUỒN INTERNET

10%

ẤN PHẨM XUẤT BẢN

14%

BÀI CỦA HỌC SINH

NGUỒN CHÍNH

1	Submitted to Ho Chi Minh University of Technology and Education Bài của Học sinh	8%
2	tailieu.vn Nguồn Internet	2%
3	www.ctu.edu.vn Nguồn Internet	1%
4	www.upm.es Nguồn Internet	1%
5	china.xilinx.com Nguồn Internet	1%
6	www.slideshare.net Nguồn Internet	1%
7	www.xilinx.com Nguồn Internet	1%
8	arxiv.org Nguồn Internet	1%
9	www.farnell.com Nguồn Internet	<1%

10	text.123docz.net Nguồn Internet	<1 %
11	Submitted to Vietnam Maritime University Bài của Học sinh	<1 %
12	mientayvn.com Nguồn Internet	<1 %
13	Submitted to National Economics University Bài của Học sinh	<1 %
14	Submitted to Foreign Trade University - Ho Chi Minh Campus Bài của Học sinh	<1 %
15	Submitted to Nha Trang University Bài của Học sinh	<1 %
16	Submitted to Ho Chi Minh City Open University Bài của Học sinh	<1 %
17	www.researchgate.net Nguồn Internet	<1 %
18	doc.edu.vn Nguồn Internet	<1 %
19	cloud.z.com Nguồn Internet	<1 %
20	portal.ptit.edu.vn Nguồn Internet	<1 %

21	ictnews.vietnamnet.vn Nguồn Internet	<1 %
22	luanvan.co Nguồn Internet	<1 %
23	vi.wikipedia.org Nguồn Internet	<1 %
24	sites.google.com Nguồn Internet	<1 %
25	Submitted to Obudai Egyetem Bài của Học sinh	<1 %
26	text.xemtailieu.net Nguồn Internet	<1 %
27	is.muni.cz Nguồn Internet	<1 %
28	tdsi.gov.vn Nguồn Internet	<1 %
29	www.tuancuong.vn Nguồn Internet	<1 %
30	doandientu.net Nguồn Internet	<1 %
31	anchor.fm Nguồn Internet	<1 %
32	tudr.thapar.edu:8080 Nguồn Internet	<1 %