VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

NBA Accreditation for B.Tech. CE, EEE, ME, ECE, CSE, EIE, IT Programmes
Approved by AICTE, New Delhi, Affiliated to JNTUH, NIRF 135th Rank in Engineering Category
Recognized as "College with Potential for Excellence" by UGC
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Estd. 1995

SEVEN HABITS OF SUCCESSFUL PEOPLE

- **Be Proactive**
- **Begin with the End in Mind**
- **Put First Things First**
- **Think Win-Win**
- **Seek first to understand, Then to be Understood**
- **Synergize**
- **Sharpen the Saw**

We have followed the above Seven habits during the course of our project work.

Swetha Goparaju **ROLL No:20071A04K9**

Sohan Kuchangari **ROLL No:20071A04N2**

Manish Vemula **ROLL No:20071A04Q9**

A SYNTHETIC VISION SYSTEM: INTELLIGENT OBJECT PERCEPTION WITH YOLO ON PYNQ-Z2

A PROJECT WORK SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS & COMMUNICATION ENGINEERING
UNDER THE SUPERVISION OF

B.B.Shabarinath
ASSISTANT PROFESSOR,
Department of ECE

Submitted By

Swetha Goparaju	**ROLL No:20071A04K9**
Sohan Kuchangari	**ROLL No:20071A04N2**
Manish Vemula	**ROLL No:20071A04Q9**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY
NBA Accreditation for B.Tech. CE, EEE, ME, ECE, CSE, EIE, IT Programmes
Approved by AICTE, New Delhi, Affiliated to JNTUH, NIRF 135th Rank in Engineering Category
Recognized as "College with Potential for Excellence" by UGC
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India.

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY
NBA Accreditation for B.Tech. CE, EEE, ME, ECE, CSE, EIE, IT Programmes
Approved by AICTE, New Delhi, Affiliated to JNTUH, NIRF 135th Rank in Engineering Category
Recognized as "College with Potential for Excellence" by UGC
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled "A Synthetic Vision System: Intelligent Object Perception with YOLO on PYNQ-Z2" that is being submitted by **Goparaju Swetha Sahithi (20071A04K9), Kuchangari Sohan (20071A04N2), Vemula Manish (20071A04Q9)** in partial fulfillment for the award of **Bachelor of Technology in Electronics & Communication Engineering** of the VNR VJIET, Hyderabad during the academic year **2023-2024** is a record of bonafide work carried out **by them** under my guidance and supervision.

Certified further that to the best of my knowledge, the work presented in this thesis has not been submitted to any other University or Institute for the award of any Degree or Diploma.

SUPERVISOR

B.B.Shabarinath
Assistant Professor
Department of ECE
VNR VJIET
Hyderabad

HEAD OF THE DEPARTMENT

Dr. S. Rajendra Prasad
Professor and Head
Department of ECE
VNR VJIET
Hyderabad

SIGNATURE OF THE EXTERNAL EXAMINER

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY
NBA Accreditation for B.Tech. CE, EEE, ME, ECE, CSE, EIE, IT Programmes
Approved by AICTE, New Delhi, Affiliated to JNTUH, NIRF 135th Rank in Engineering Category
Recognized as "College with Potential for Excellence" by UGC
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DECLARATION

We do declare that the Seminar Report entitled "A Synthetic Vision System: Intelligent Object Perception with YOLO on PYNQ-Z2" submitted in the department of Electronics and Communication Engineering (ECE), Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Electronics & Communication Engineering** is a bonafide record of my work carried out under the supervision of **B.B.Shabarinath,** Assistant Professor, VNRVJIET.

Also, we declare that the matter embodied in this thesis has not been submitted by me in full or in any part thereof for the award of any degree/diploma of any other institution or university previously.

Place: Hyderabad **Date:** 28-03-2024

SWETHA GOPARAJU **SOHAN KUCHANGARI** **MANISH VEMULA**

ACKNOWLEDGEMENT

We are thankful to the Principal **Dr. C.D.NAIDU**, VNRVJIET, Hyderabad, for giving me permission to carry out this project.

Our sincere thanks to **Dr. S. RAJENDRA PRASAD**, Professor and Head of the Department, ECE, VNR VJIET for his esteemed guidance and encouragement provided during the course of my project.

We would like to express our sincere thanks to **MR. B.B.Shabarinath**, Assistant Professor, VNR VJIET for her precious guidance and kind co-operation at every step of this project work.

We are thankful to all the staff members of the ECE department, VNR VJIET for helping me during this project.

We are thankful to all the project committee members of the ECE department, VNR VJIET for helping me during this project.

Finally, we are very thankful to our family members and friends for their great moral support.

SWETHA GOPARAJU

SOHAN KUCHANGARI

MANISH VEMULA

ABSTRACT

Object detection using deep learning has emerged as a powerful solution for various applications due to the high accuracy of its models. However, the need for more efficient and compact implementations has increased due to the resource-intensive nature of deep learning models. When paired with Central Processing Units (CPUs), Programmable Logic Devices (PLDs)—like Field Programmable Gate Arrays (FPGAs)—offer a potentially effective option for object detecting applications. These gadgets make it possible to implement operations in hardware, enabling quick and low-power execution. Well-known FPGA manufacturers such as Xilinx offer tools such as the Deep Neural Network Development Kit (DNNDK) that make it easier to deploy and compress neural networks across several boards with diverse architectures. The goal of this project is to use the Xilinx PYNQ-Z2 development board to construct an object detector. To fit the board's constraints, two models—YOLOv3 and Tiny YOLO—are chosen and compressed with DNNDK's help. Tiny YOLO achieves a mean Average Precision (mAP) of 0.0542, while YOLOv3 achieves a mAP of 0.4036, according to the evaluation results. These results show promise for further improvements and advancements in object identification technology development on an affordable FPGA platform.

INDEX

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION:

In the realm of embedded systems, the PYNQ-Z2 board emerges as a versatile and powerful platform, capable of accelerating deep learning algorithms and facilitating edge computing. This project aims to harness the PYNQ-Z2's potential by implementing the YOLO (You Only Look Once) algorithm for real-time object detection. YOLO, with its unique approach to bounding box prediction and class probability assignment, offers a swift and efficient method for object perception.

The PYNQ-Z2 board, equipped with a Zynq-7020 FPGA, provides a conductive environment for deploying the YOLO algorithm. By leveraging the FPGA's parallel processing capabilities, we can achieve significant improvements in inference speed, making it suitable for real-time applications. The project encapsulates the entire workflow, from the DPU (Deep Learning Processing Unit) implementation to the deployment of the optimized YOLO model, ensuring that the system is not only accurate but also power-efficient.

This endeavor not only demonstrates the feasibility of running complex neural networks on resource-constrained devices but also paves the way for innovative applications in various fields such as autonomous vehicles, surveillance, and smart IOT devices. With this project, we step closer to a future where intelligent object perception is ubiquitous, opening doors to endless possibilities.

1.2 DESIGN APPROACH:

The You Only Look Once (YOLO) architecture was developed to create a one step process for detection and classification. The image is divided into a fixed grid of uniform cells and bounding boxes are predicted and classified within each cell. This architecture enables faster object detection.

Designing an intelligent object perception system with YOLO on the PYNQ-Z2 board involves several key steps to ensure efficient and accurate detection.

DPU Implementation: The first step is to create a hardware layout capable of processing neural networks, including YOLO, called the Deep Learning Processing Unit (DPU). The DPU is tailored to the PYNQ-Z2's capabilities, and you'll need to create an SD card image with the DPU and other software configurations.

Model Optimization: The YOLO model will be compressed and optimized to fit the PYNQ-Z2 board's constraints using tools like DNNDK v3.1. This involves quantization of the model and compilation, resulting in a file that can communicate with the DPU to form the compressed YOLO network.

Deployment: The final step is setting up the board to see the results. This involves organizing and compiling the files into an executable file capable of running YOLOv3 inference on a desired image.

1.2 PROBLEM STATEMENT:

The project's goal is to create a real-time object detection system using YOLO (You Only Look Once) on the PYNQ-Z2 board, optimizing the YOLOv3 model for FPGA deployment while maintaining high inference speed and accuracy within resource restrictions. Integration of the DPU version 3.0 into the system is critical, necessitating the use of software tools such as Vivado, DNNDK, and Petalinux for model compilation and system configuration. The major goal is to smoothly load the optimized model onto an SD card image, allowing for effective real-time object identification with low latency. Evaluation criteria like as inference speed, detection accuracy, and resource utilization provide complete insights into the viability of FPGA-based systems for object identification applications. The project tackles the growing demand for real-time object detection systems in a variety of sectors, including surveillance, self-driving vehicles, and industrial automation, by proposing potential solutions to improve safety, efficiency, and productivity.

1.2 OBJECTIVES:

1. Implementation of YOLO on PYNQ-Z2:

Integrate YOLO object detection algorithm on the PYNQ-Z2 board, leveraging FPGA acceleration for optimized performance.

2. Real-time Object Detection:

Enable real-time object detection by integrating camera input with the YOLO model on the FPGA-accelerated PYNQ-Z2 platform.

3. Performance Optimization and Evaluation:

Optimize the system for efficiency and accuracy, conducting thorough evaluations to ensure robust object detection on the PYNQ-Z2

CHAPTER 2- OVERVIEW OF BASICS:

2.1 WHAT IS PYNQ-Z2:

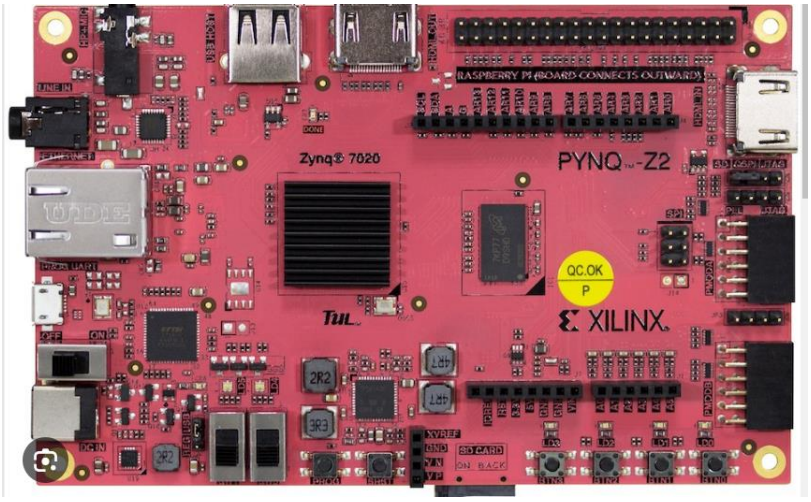

Fig.2.1.PYNQ-Z2 Board

The PYNQ-Z2 board is a versatile development platform for the Xilinx Zynq-7000 SoC (System on Chip). It makes it easier to construct embedded systems with programmable logic and microprocessors by letting users utilize the Python language and libraries to create complicated electronic systems. Here are some main features of the PYNQ-Z2 board:

SoC: Zynq Z7020

Memory: 512MB DDR3

Storage: Micro SD

Video: HDMI In & Out ports

Audio: ADAU1761 codec with HP + Mic, Line in

Network: 10/100/1000 Ethernet

Expansion: USB host (PS), GPIO, Arduino Header, Pmod, Raspberry Pi header.

Other I/O: User LEDs, Pushbuttons, Dip switches

Dimensions: 3.44” x 5.51” (87mm x 140mm)

The PYNQ-Z2 development board is based on the Xilinx Zynq-7000 SoC (System on Chip), which combines the hardware programmability of an FPGA (Field Programmable Gate Array) with the software flexibility of a CPU. This is how it works.

SoC Architecture: The PYNQ-Z2 is powered by the Zynq SoC, which combines the programmable logic of an FPGA with the computing capability of an ARM Cortex-A9 processor. This enables high-performance computing and flexible hardware configuration.

Python Productivity: The PYNQ framework, which stands for Python Productivity for Zynq, allows developers to use Python and its libraries to interact with the board. This makes it easy to program the FPGA and conduct operations such as image and video processing.

Peripheral Support: The PYNQ-Z2 supports a variety of interfaces and connections, including HDMI, USB, Ethernet, and headers for Arduino and Raspberry Pi. This enables the attachment of numerous peripherals and the increase of the board's functionality.

Development tools: Tools like Xilinx Vivado can be used to generate and synthesis hardware designs, which are then put into the board to operate components like LEDs and switches.

2.2 WHAT IS YOLO:

The YOLO (You Only Look Once) method is a cutting-edge, real-time object recognition system that uses a single neural network to process the entire image. This network separates the image into regions and calculates bounding boxes and probabilities for each one. These bounding boxes are weighted according to the estimated probabilities.

YOLO is extremely fast since it processes the entire image at test time, which informs its predictions based on the image's global context. It is also extremely precise, making it an ideal choice for real-time applications. On the PYNQ-Z2 board, YOLO can be used to take advantage of the Zynq-7020 FPGA, which speeds up the inference process.

The FPGA's parallel processing capabilities enable YOLO to run effectively, making it ideal for real-time object identification applications. The combination of YOLO's speed and accuracy with the PYNQ-Z2 board's computing capability creates an effective tool for intelligent object perception in a variety of applications.

HOW YOLO DETECTS OBJECTS:

The YOLO (You Only Look Once) algorithm predicts bounding boxes as part of the object detection process. Here's a simple description of how it works:

Grid Division: YOLO begins by dividing the input image into a grid of cells, usually a (S \times S) grid.

Bounding Box prediction: Each cell in the grid anticipates a specific number of bounding boxes. For each bounding box, the model predicts the following.

Box coordinates: They are the center ((x, y)), width ((w)), and height ((h)) of the box.

Confidence Score: The confidence score measures a box's possibility of containing an object as well as its accuracy.

Non-Max Suppression: To reduce duplication, YOLO uses a technique known as non-max suppression, which retains only the bounding boxes with the greatest confidence ratings while eliminating overlapping boxes with lower scores.

The bounding box coordinates are predicted based on the grid cell's position, and the dimensions are normalized to the image's size. This method enables YOLO to recognize objects at various scales and aspect ratios effectively.

The PYNQ-Z2 board, with its FPGA capabilities, can speed up this process, allowing for real-time object detection by rapidly running the YOLO algorithm.

CHAPTER 3- RESEARCH BACKGROUND :

3.1-LITERATURE SURVEY:

Ákos Mándi and Jeney Máté utilized the FPGA-based PYNQ-Z2 board for hardware acceleration in image processing. Their study attempted to create a working prototype of a controller and processing unit for an autonomous vehicle that integrated a laser range finder for depth sensing and a camera for visual input. The study demonstrated the promise of the PYNQ-Z2 platform in autonomous vehicle technology by focusing on improving inference acceleration for self-driving automobiles through FPGA-based hardware acceleration. Tanvir Ahmad and Belal Ahmad used a modified YOLO neural network to study object detection. By modifying the loss function, adding a spatial pyramid pooling layer, and integrating an inception model with 1 * 1 convolution kernels, they modified the YOLOv1 network. Their research shows how well the modified network can extract rich characteristics from photos, which leads to better object detection capabilities and reliable categorization and detection of different item classes.

Tausif Diwan and G. Anirudh conducted an extensive review of single-stage object detection algorithms, with a focus on the YOLO series. They analysed underlying architectures, regression formulation, and comparative performance metrics. The study underscores the significance of single-stage object detectors and highlights the superior performance of YOLOs in terms of detection accuracy and inference time, positioning them as a viable option for real-time object detection applications. Philip J, Patil S and Philip J. examined YOLO algorithm-based real-time object detection. They examined previous research on object detection methods, including conference proceedings, journal publications, and studies; they paid special attention to the YOLO algorithm. Their investigation demonstrated how well the YOLO algorithm performs in obtaining high levels of accuracy while achieving real-time object detection capabilities. The paper offers a thorough summary of the developments and uses of the YOLO method in image processing and computer vision.

Wenhao Li and Huaixiang Hu designed and tested a hardware network acceleration circuit using the Pynq-Z2 development platform and the YOLO-v2 paradigm. To lower hardware resource consumption and keep network detection accuracy, they used adder optimization and floating-point quantization. They reduced power consumption and increased processing power, performance, and circuit latency through parallel and array optimization. The study's hardware acceleration strategy proved effective as seen by its 80.6% mean average precision, 27.1 GOP/s overall computing power, and 2.6W power consumption.

3.2 SOFTWARE TOOLS USED:

3.2.1 Vivado:

Vivado is a package of tools for developing hardware for Xilinx FPGAs. You can basically make hardware in three ways:

High Level Synthesis (HLS): Vivado HLS is a tool that lets you to build hardware with high-level programming languages such as C, C++, or OpenCL. This enables the developer to express desired functionality at a more abstract level, making the hardware development process more straightforward and accessible.

Hardware Description Languages (HDL): The traditional method of hardware development involves writing code in hardware description languages such as VHDL or Verilog. Vivado provides an Integrated Development Environment (IDE) for writing, testing, verifying, and debugging these projects.

Block Design: For complex projects, Vivado provides a graphics-based system design tool that allows users to link pre-projected functional blocks and adjust the system's behaviour. These blocks are known as Intellectual Property (IP) and were developed by Xilinx and other creators.

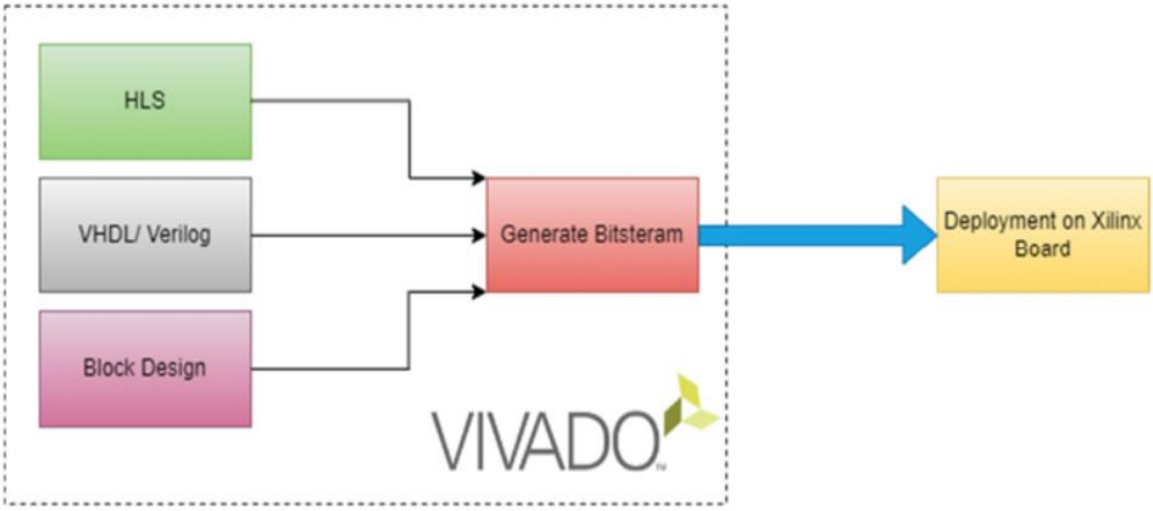

Fig.3.2.1.Vivado Development

3.2.2 Petalinux:

Petalinux is an open-source development platform from Xilinx that allows you to construct, customize, and implement Linux operating systems (OS) on Xilinx devices. As a result, it is possible to customize libraries and packages on the board's operating system, as well as specify the kernel, I/Os, drivers, and other components. In addition to software, this is the platform that connects the hardware designed in Vivado to the board. After all configurations and identifying the binary file relative to the necessary hardware, we may copy particular files to a micro SD card and boot the board from it.

3.2.3 DNNDK:

DNNDK stands for Deep Neural Network Development Kit. This is a series of tools created by Xilinx to help expedite the creation and implementation of Deep Neural Networks (DNNs) on Xilinx FPGA

devices. This development kit was created specifically to take advantage of the capability of parallel processing and the FPGA's programmable flexibility to expedite machine learning operations and DNN inference.

CHAPTER 4- IMPLEMENTATION :

4.1 BLOCK DIAGRAM:

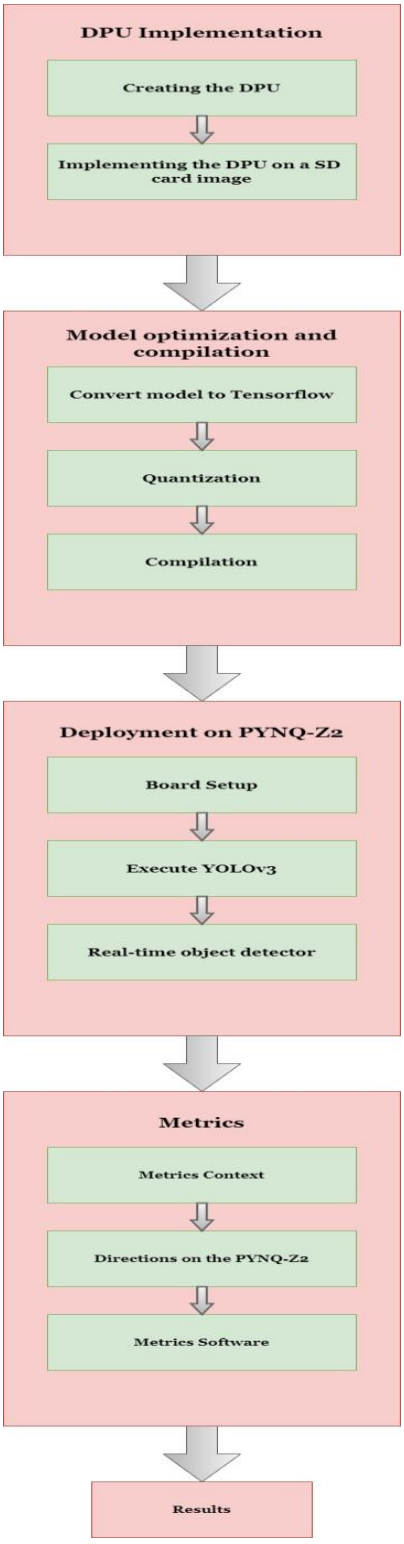

Fig.4.1.Block diagram of Object Detection using YOLO on PYNQ-Z2 Board

We started our YOLO on PYNQ-Z2 project by incorporating the Deep Learning Processing Unit (DPU) version 3.0 into our system. This required a rigorous process of setting up and configuring the DPU to enable a smooth deployment onto an SD card image. Our trip began with the necessary model optimization and compilation stages. First, we converted the YOLOv3 model to Tensor flow format, which is critical for compatibility with our setup. We then quantized the model to improve inference efficiency before deploying it on the PYNQ-Z2 board. Using Vivado and DNNDK, we meticulously constructed the optimized model to achieve optimal hardware performance.

We started with real-time object detection after assembling the board and creating the YOLOv3 model. We conducted detailed studies utilizing the PYNQ-Z2 platform, concentrating on critical metrics such as inference speed and detection accuracy. We gained profound insights into the capabilities and limitations of our YOLOv3 implementation on the PYNQ-Z2 platform by doing thorough investigation that included both software and hardware components. This extensive assessment offered a thorough understanding of the final outcomes, allowing us to tweak our methodology and enhance performance for future deployments.

4.2 METHDOLOGY:

Setup and Dependencies:

Ensure the PYNQ-Z2 board is properly configured with the necessary software tools, and dependencies like PYNQ library and DPU implementation by developing a Hardware to run on the PYNQ-Z2 capable of simulating YOLO

YOLO Integration:

Make use of the FPGA capabilities of the PYNQ-Z2 to expedite the YOLO inference procedure. Use the DNNDK API to integrate the YOLO algorithm with the PYNQ framework. Utilize its parallel processing capabilities for object detection. Develop the YOLO inference and display the results in a window along with the detection's

Performance Evaluation:

Evaluate the object detection performance on the PYNQ-Z2 board by measuring metrics such as IOU, Precision and Recall in such a way we can inference speed and accuracy. Fine-tune parameters and optimize the implementation for real-time applications, demonstrating the feasibility of YOLO on resource-constrained embedded systems.

4.3 PROCEDURE:

Step 1: Develop a DPU implementation for the PYNQ-Z2 to simulate neural networks such as YOLO.
Step 2: Use Petalinux, a Xilinx tool, to integrate DPU hardware onto the board and build an SD card image with the DPU implementation.
Step 3 involves optimizing and compiling the YOLO model using DNNDK.
Step-4 Deployment on PYNQ-Z2, where we Tie everything together to effectively conduct YOLO inference on an image
Step 5: Metrics: Measure YOLO model parameters after compression.

4.4 IMPORTANT METRICS FOR OBJECT DETECTION:

Intersection Over Union (IOU):

Intersection over union (IOU) is a measure for determining the intersection of two bounding boxes. This requires both the ground truth bounding boxes (the ones that actually confine the object) and the predicted bounding boxes (the ones predicted by the model).

$$\mathbf{IOU}=\frac{\boldsymbol{area}\ (\boldsymbol{gt}\cap\boldsymbol{pd})}{\boldsymbol{area}(\boldsymbol{gt}\cup\boldsymbol{pd})}$$

$$\mathbf{IOU}=\frac{\boldsymbol{area\ of\ overlap}}{\boldsymbol{area\ of\ union}}$$

IoU ranges from 0 to 1, with 0 denoting no intersection between the ground truth and forecast bounding boxes and 1 indicating perfect overlapping. This measure will be required to discriminate between correct and incorrect detections; hence it must be coupled with a threshold such as 50%, 75%, or 95%. The following concepts use the distinction depending on the threshold value.

True Positives (TP): Correct detection. IOU is more or equal to the threshold.
False Positive (FP): Denotes incorrect detection. IOU is below or equal to the threshold.
True Negative (TN): Indicates the bounding boxes that should not be identified on the image. This does not apply to object detection because there are too many options.
False Negative (FN): The ground truth is unnoticed.

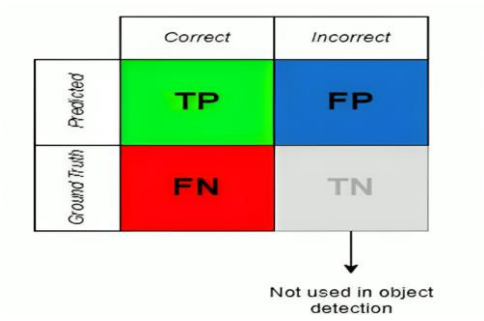

Fig.4.4.1.Confusion matrix on object detection

Precision:
Precision refers to the model's ability to detect only relevant things. This represents the percentage of correct detections.

$$\textbf{Precision} = \frac{TP}{TP+FP} = \frac{TP}{all\ detections}$$

Recall:
It indicates the model's capacity to identify ground truths. In other words, it indicates the number of true positives (TP) discovered based on a set of ground truths.

$$\textbf{Recall} = \frac{Tp}{TP+FN} = \frac{TP}{all\ ground-truths}$$

Precision X Recall Curve:
A graph that links Precision with Recall for each class is an effective approach to assess an object detector's performance. A good object detector is one whose Precision remains high as the Recall grows, indicating that even when the confidence threshold is changed, the Precision and Recall remain high.

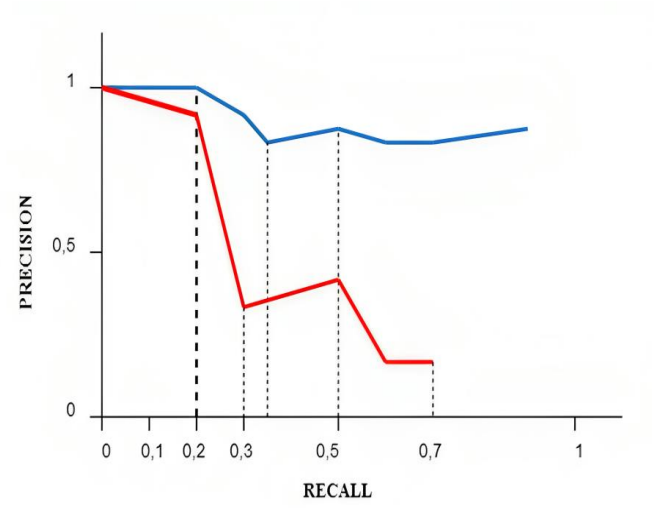

Fig.4.4.2Precision X Recall Curve

Blue a good object detector and in Red a less good object detector.

Average Precision (AP):
Average Precision (AP) is simply the area under the Precision X Recall curve.

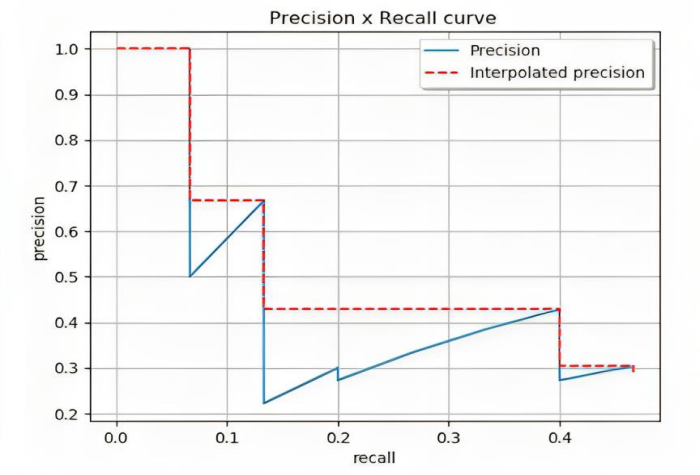

Fig.4.4.3.Average Precision

Mean Average Precision (mAP):

This represents the average AP for the total set of classes.

CHAPTER 5- RESULTS :

5.1.Input:

Fig.5.1.Input image

5.2.Output:

Fig.5.2.Output image

5.3.Accuracy:

The accuracy of model is as follows. Here Average Precision (AP) is simply the area under the Precision X Recall curve. Mean Average Precision (mAP) represents the average AP for the total set of classes.

Platform	AP	AP50	AP75	APS	APM	APL	mAP
YOLOv3	33,00	57,90	34,40	18,30	35,40	41,90	55,30
PYNQ-Z2 YOLOv3	21,41	40,10	20,53	06,07	20,46	32,19	40,36

Table.5.3.Accuracy table of YOLOv3 and PYNQ-Z2 YOLOv3

- YOLOv3 model on the PYNQ-Z2 is 23 times faster and 27% less accurate than the original YOLO model
- This goes to show that FPGA's are indeed a really good way to embed a Neural Network.
- The Hardware makes the execution time of the Neural Network fast and the accuracy still very acceptable.
- Also, the PLD has low power consumption - about 4,166W which is less than some LED light bulb.

CHAPTER-6 CONCLUSION:

6.1 Conclusion

In conclusion, implementing object detection using YOLO on the PYNQ-Z2 board involves setting up the board, downloading YOLO model weights, and configuring dependencies. The algorithm efficiently detects objects in real-time, making it suitable for embedded systems. Further optimization and experimentation with different YOLO configurations can be undertaken to achieve a balance between accuracy and computational efficiency. Overall, the PYNQ-Z2 board offers a capable platform for deploying YOLO-based object detection applications.

6.2 Future Scope:

To reach even greater levels of speed and accuracy in the future, additional optimization and improvement of the YOLO implementation on the PYNQ-Z2 can be investigated. Furthermore, broadening the application scope to include many fields including autonomous systems, robotics, and surveillance has a lot of promise for solving practical issues. Working together with researchers and industry partners can spur creativity and open up new avenues for utilizing YOLO on embedded devices with limited resources.

CHAPTER-7 REFERENCES

[1] Mándi Á, Máté J, Rózsa D, Oniga S. Hardware accelerated image processing on FPGA based PYNQ-Z2 board. *Carpathian Journal of Electronic and Computer Engineering*. 2021; 14(1):20-23. doi:https://doi.org/10.2478/cjece-2021-0004

[2] Diwan T, Anirudh G, Tembhurne JV. Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications. 2022; 82:9243-9275. doi:https://doi.org/10.1007/s11042-022-13644-y

[3] Haritha Sai L, Harshini M, Patil S, Philip J. Real Time Object Detection using YOLO Algorithm. *2022* 6th International Conference on Electronics, Communication and Aerospace Technology. Published online December 1, 2022. doi:https://doi.org/10.1109/iceca55336.2022.

[4] Li W, Hu H. FPGA-based Object Detection Acceleration Architecture Design. *Journal of Physics: Conference Series*. 2022; 2405(1):012011. doi:https://doi.org/10.1088/1742-6596/2405/1/012011

[5] Agarwal S, Terrail JO, Jurie F (2018) Recent advances in object detection in the age of deep convolutional neural networks. arXiv preprint arXiv: 1809.03193. https://doi.org/10.48550/arXiv.1809.03193

[6] Albelwi S, Mahmood A (2017) A framework for designing the architectures of deep convolutional neural networks. Entropy 19(6):242

[7] Bengio Y, Courville AC, Vincent P (2012) Unsupervised feature learning and deep learning: a review and new perspectives. CoRR, abs/1206.5538, 1(2665)

[8] T. F. Gonzalez, "Handbook of approximation algorithms and metaheuristics," Handb. Approx. Algorithms Metaheuristics, pp. 1–1432, 2007, doi: 10.1201/9781420010749.