DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

September 10-11, 2025

Radisson Blu, Marathahalli, Bangalore

FPGA Based CNN Accelerator For Real Time Gun Sound Detection Using Systolic Array and Adaptive Processing

LALITH KUMAR R
MADHUSUDHANAN K
MANOJ KUMAR A

Team Name: VOICE VIBES
Submission Number: 3627
Affiliation: Anna University

Table of Contents

List of Tables

I. Introduction

Our project focuses on developing a custom hardware accelerator to enhance the performance of a CNN-based gunshot classification model. The system is designed to process audio signals from multiple microphones, converting them into spectrograms for analysis. These spectrograms are then fed into a CNN model, which determines whether the detected sound is a gunshot. Once classified, the information is passed to a localization algorithm that analyzes the amplitude and determines the direction of arrival (DOA) of the gunshot.

To achieve high efficiency, we implement an optimized CNN accelerator featuring a systolic array architecture, zero-sparse computation, and an adaptive kernel to maximize performance while operating within the constraints of an FPGA. Our approach differs from traditional spectrogram computation methods, introducing enhanced techniques that improve processing efficiency when combined with CPU and accelerator hardware.

II. Background Research

Gunshot detection and localization systems are crucial for public safety, security enforcement, and military applications. Traditional methods rely on software-based signal processing techniques, which often suffer from high latency and power consumption, making them unsuitable for real-time embedded applications. The rise of deep learning, particularly **Convolutional Neural Networks** (CNNs), has enabled more accurate and efficient sound classification. However, deploying CNNs on conventional processors or GPUs results in significant computational overhead and energy inefficiency, limiting their practical deployment in resource-constrained environments.

To address these limitations, FPGA-based CNN accelerators have gained traction due to their ability to provide parallel processing, low power consumption, and real-time inference capabilities. Our research focuses on designing a custom FPGA accelerator tailored for real-time gunshot detection, integrating advanced hardware optimization techniques such as **systolic array-based convolution, sparse computation, and adaptive kernel selection.** By leveraging these techniques, our approach aims to reduce computational redundancy, optimize resource utilization, and enhance inference speed, making it suitable for real-time embedded security systems.

The proposed FPGA accelerator offers a novel approach to achieving high-speed, low-power inference, making it ideal for deployment in military defense strategies

III. Goal and Objectives

Goal: The Primary goal of this research is to develop a FPGA-Based CNN accelerator optimized for real-time gunshot detection. By leveraging hardware-efficient deep learning techniques, the proposed system aims to achieve high-speed inference, reduced power consumption, and enhanced computational efficiency in comparison to traditional software-based approaches.

Objectives:

1. **Develop a CNN-based classification model** capable of accurately identifying gunshot sounds from spectrogram representations of audio signals.
2. **Implement a systolic array-based convolution** engine to accelerate CNN computations, ensuring parallel processing and efficient data flow.
3. **Integrate sparse computation techniques** to eliminate redundant calculations and optimize memory usage.
4. **Incorporate adaptive kernel selection** to dynamically adjust processing based on input signal characteristics, improving accuracy and power efficiency.
5. **Design and implement the entire accelerator in Verilog** HDL to ensure efficient FPGA deployment and real-time operation.
6. **Evaluate performance metrics** such as inference speed, power consumption, and classification accuracy to validate the effectiveness of the proposed hardware accelerator.

IV. Design Process

i. Refined Solution

Comparing to Stage 1 design, now we have aimed to increase the performance by **loop unrolling** few loops which also optimizes the resource utilization and improved the computation performance of Processing elements of Systolic array by adding **Wallace tree multiplier and Kogge stone adders** for efficient matrix multiplication for convolution of matrices.

ii. Functional Specification

The spectrogram data is streamed through the **FMC connector**, ensuring seamless integration with external audio preprocessing modules.

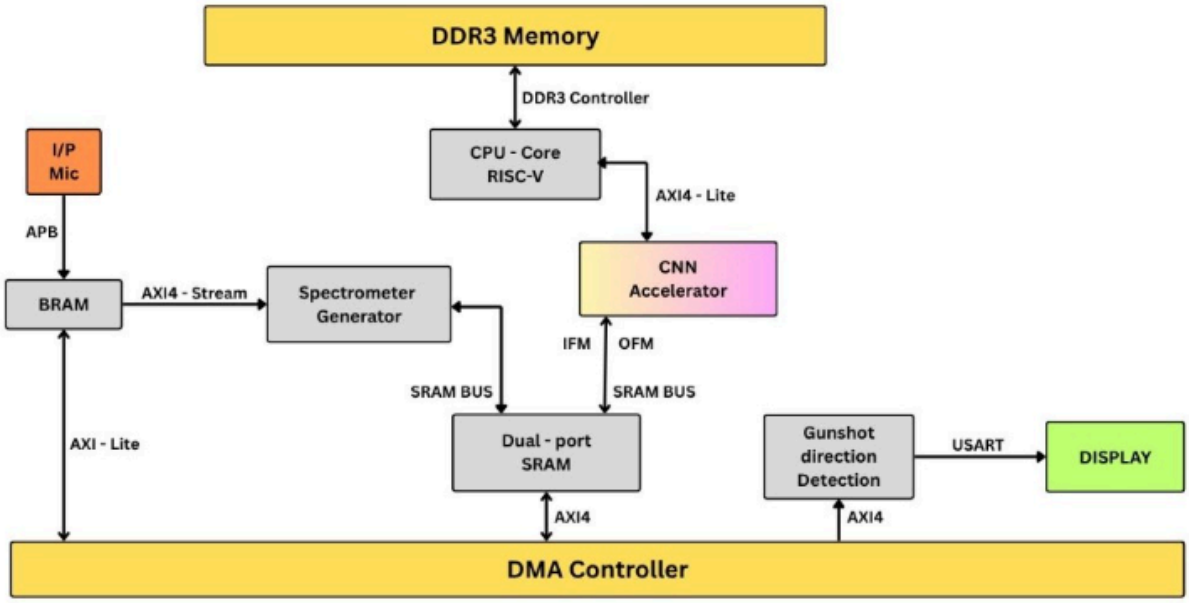
A fully-pipelined **8-bit systolic array**, optimized for parallel execution of convolution operations. To maximize performance, the systolic architecture integrates **Wallace tree multipliers for fast dot-product calculations and a Kogge-Stone adder** to accelerate summation within accumulation layers, reducing latency in matrix multiplications. Intermediate feature maps and activation outputs are stored in distributed RAM and block RAM, preventing excessive external memory accesses.

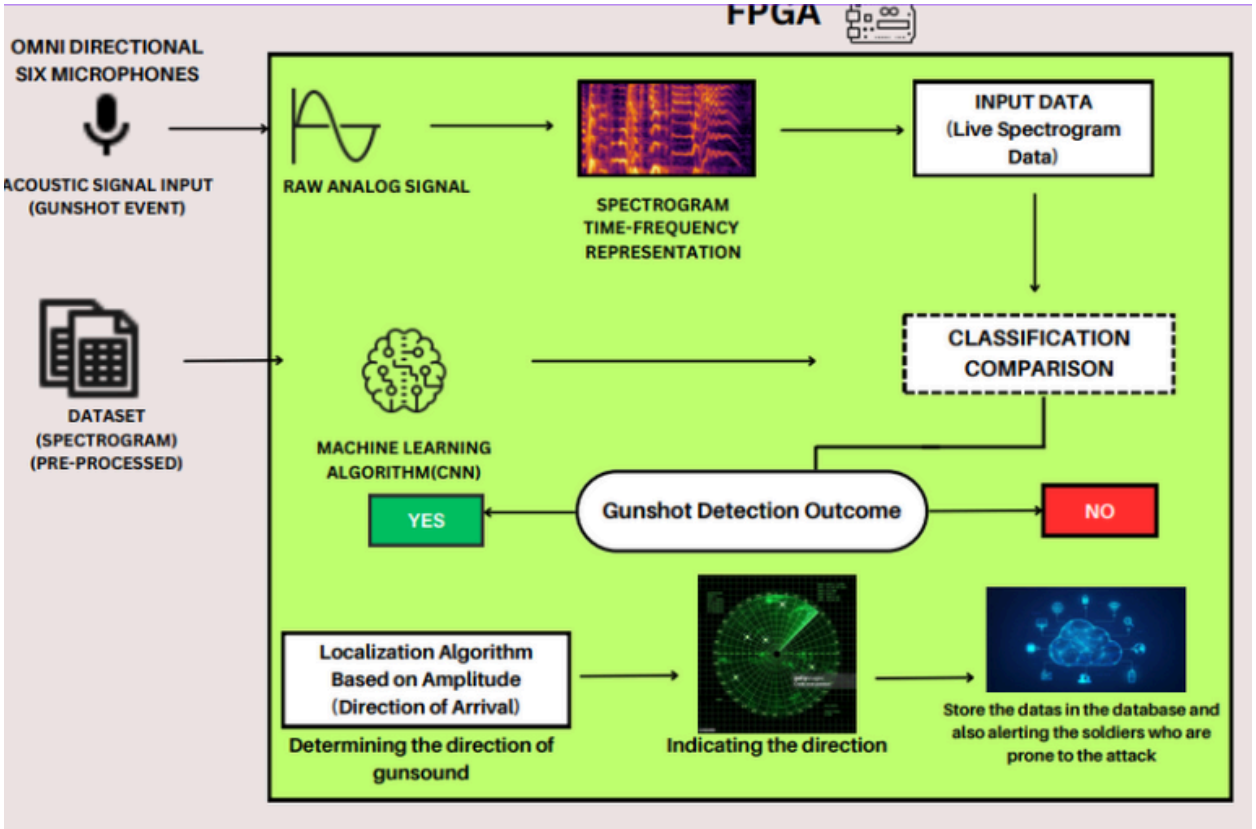
The final classification result and direction of arrival (DOA) estimation are stored in a dedicated output buffer, which is mapped to the **AXI interface** for direct DesignProcess.

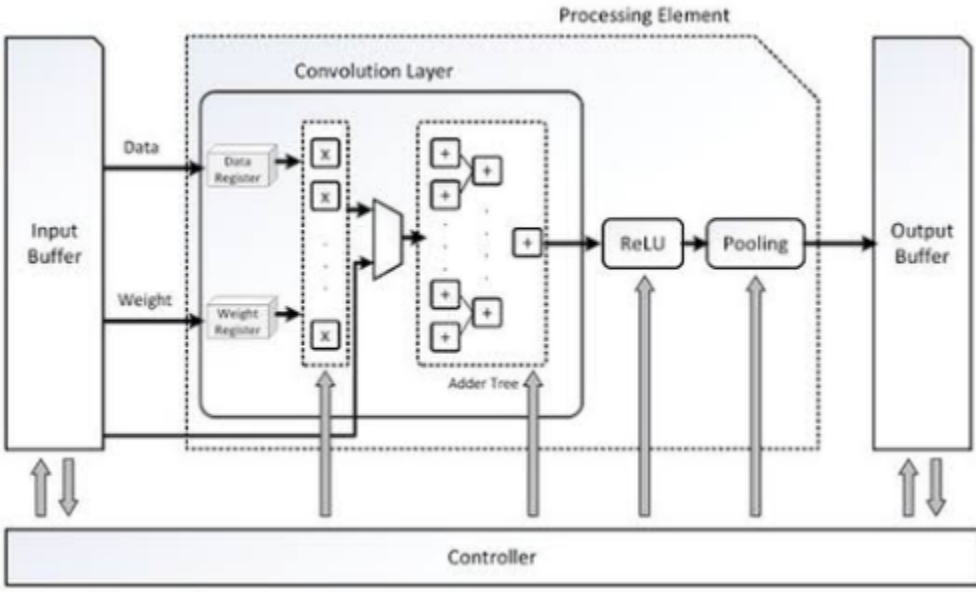
The challenge is to develop a hardware-accelerated CNN that can efficiently classify gunshot sounds from spectrograms in real time while consuming minimal power.

Our solution aims to design and implement an optimized CNN accelerator on the Genesys FPGA with the CDAC Vega processor, leveraging a systolic array architecture, sparse computation techniques, and adaptive kernel selection for efficient deep learning inference communication with the CDAC Vega AT1051 processor. The AT1051 retrieves the processed results through a high-bandwidth AXI DMA controller, enabling low-latency data transfer.

iii. SoC Design (SoC level block diagram with interfaces/sensors used)



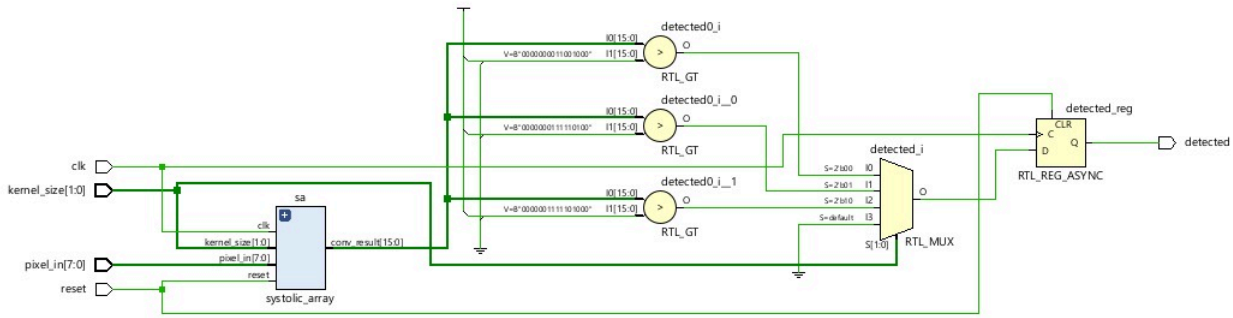

FPGA
OMNI DIRECTIONAL SIX MICROPHONES
ACOUSTIC SIGNAL INPUT (GUNSHOT EVENT)
RAW ANALOG SIGNAL
SPECTROGRAM TIME-FREQUENCY REPRESENTATION
INPUT DATA (Live Spectrogram Data)
DATASET (SPECTROGRAM) (PRE-PROCESSED)
MACHINE LEARNING ALGORITHM(CNN)
CLASSIFICATION COMPARISON
YES
Gunshot Detection Outcome
NO
Localization Algorithm Based on Amplitude (Direction of Arrival)
Determining the direction of gunsound
Indicating the direction
Store the datas in the database and also alerting the soldiers who are prone to the attack

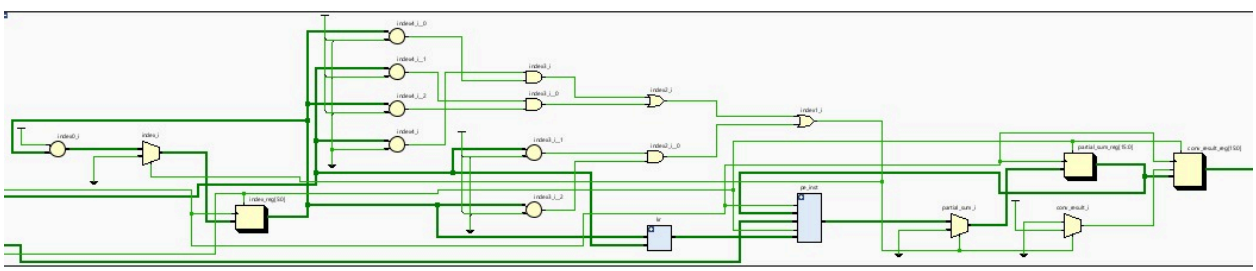

Processing Element
Convolution Layer
Input Buffer
Data
Weight
Data Register
Weight Register
Adder Tree
ReLU
Pooling
Output Buffer
Controller

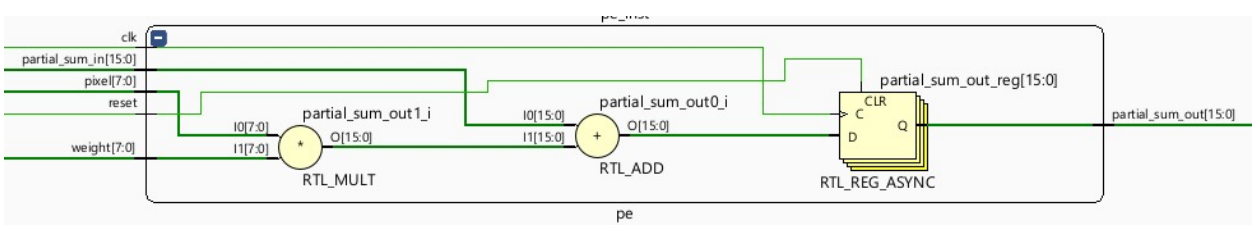
iv. Accelerator Design Implementation

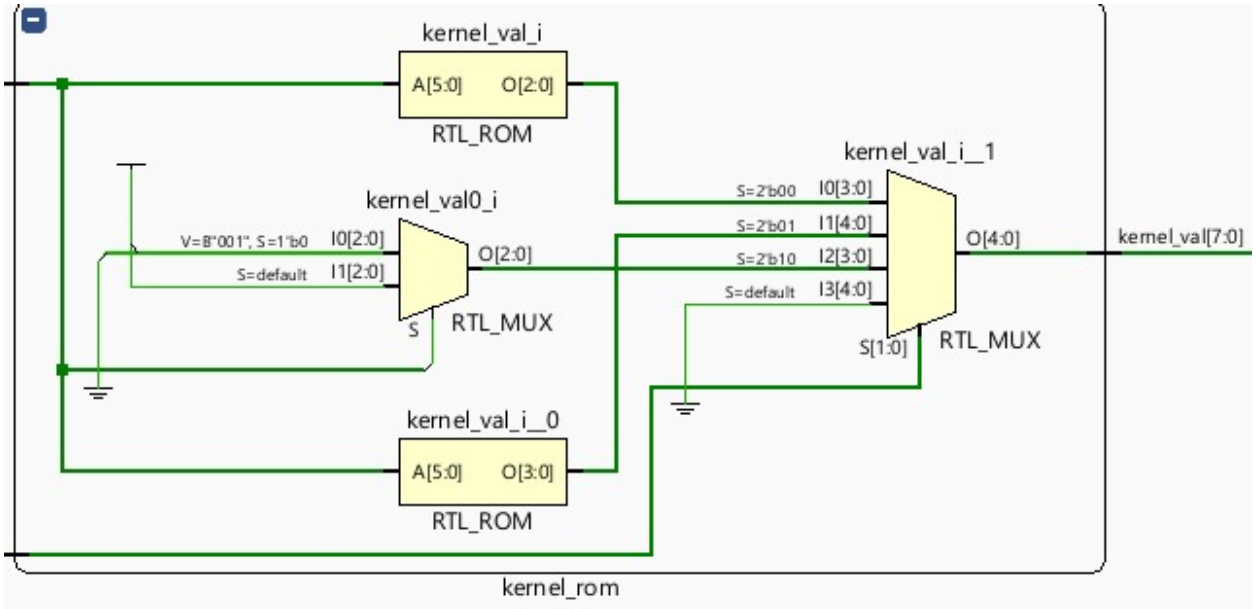
The Accelerator is designed in Xilinx Vivado using verilog and simulation results are obtained.

RTL of Accelerator:









First RTL->RTL of entire Accelerator(Top Module) with the internal Modules Minimized.
Second RTL->RTL of Systolic Array.
Third RTL->RTL of Processing Elements of Systolic Array
Fourth RTL->RTL of Adaptive Kernel Selector.

v. Test Plan/Test cases

CELL USAGE Report:

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	116	0	0	8000	1.45
LUT as Logic	116	0	0	8000	1.45
LUT as Memory	0	0	0	5000	0.00
Slice Registers	55	0	0	16000	0.34
Register as Flip Flop	55	0	0	16000	0.34
Register as Latch	0	0	0	16000	0.00
F7 Muxes	0	0	0	7300	0.00
F8 Muxes	0	0	0	3650	0.00

* Warning! The Final LUT count, after physical optimizations and full implemer
Warning! LUT value is adjusted to account for LUT combining.

```
Detailed RTL Component Info :
+---Adders :
      16 Input   20 Bit       Adders := 1
       9 Input   20 Bit       Adders := 1
       3 Input   20 Bit       Adders := 1
+---Registers :
                   20 Bit    Registers := 4
                    1 Bit    Registers := 1
+---Muxes :
       4 Input    4 Bit        Muxes := 3
       4 Input    3 Bit        Muxes := 6
       4 Input    2 Bit        Muxes := 8
       3 Input    1 Bit        Muxes := 3
       4 Input    1 Bit        Muxes := 2
-----------------------------------------------
Finished RTL Component Statistics
```

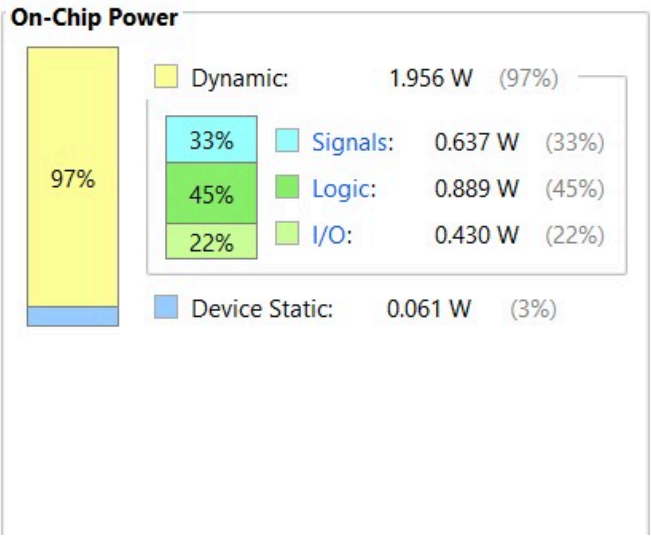
Ref Name	Used	Functional Category
LUT2	91	LUT
FDCE	55	Flop & Latch
LUT6	36	LUT
CARRY4	16	CarryLogic
LUT4	12	LUT
IBUF	12	IO
LUT5	9	LUT
LUT3	4	LUT
LUT1	2	LUT
OBUF	1	IO
BUFG	1	Clock

```

Report Cell Usage:
+------+-------+------+
|      |Cell   |Count |
+------+-------+------+
|1     |BUFG   |     1|
|2     |CARRY4 |    16|
|3     |LUT1   |     2|
|4     |LUT2   |    91|
|5     |LUT3   |     4|
|6     |LUT4   |    12|
|7     |LUT5   |     9|
|8     |LUT6   |    36|
|9     |FDCE   |    55|
|10    |IBUF   |    12|
|11    |OBUF   |     1|
+------+-------+------+

Report Instance Areas:
+------+------------+---------------+------+
|      |Instance    |Module         |Cells |
+------+------------+---------------+------+
|1     |top         |               |   239|
|2     |  sa        |systolic_array |   224|
|3     |    pe_inst |pe             |   126|
+------+------------+---------------+------+
-----------------------------------------------------------------
Finished Writing Synthesis Report : Time (s): cpu = 00:00:22 ;
```

Power Usage Report:

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

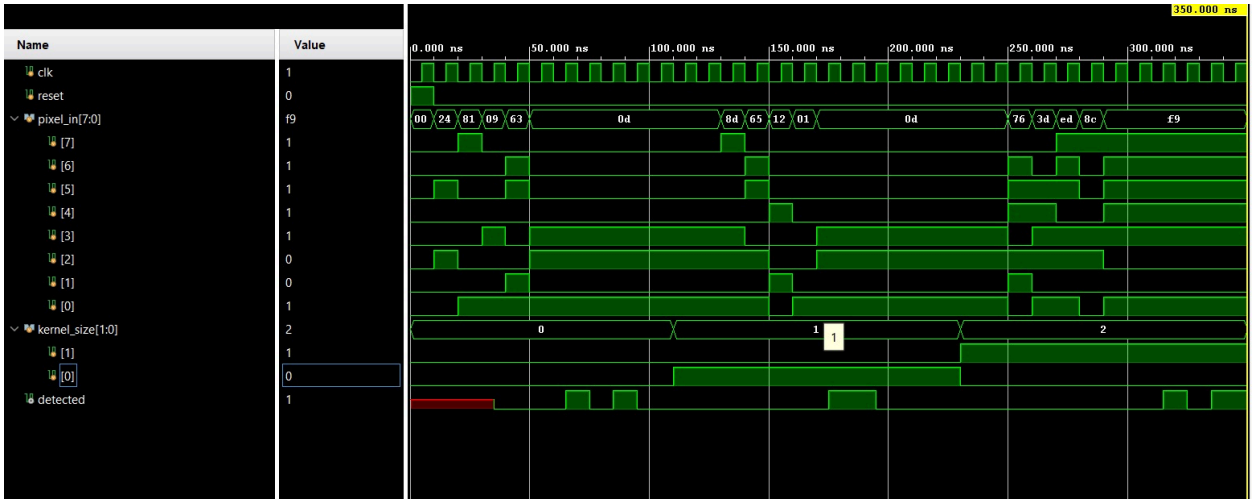
Total On-Chip Power:	**2.018 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**35.6°C**
Thermal Margin:	64.4°C (12.2 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.3°C/W
Power supplied to off-chip devices:	0 W



Utilization	Name	Signals (W)	Data (W)	Clock Enable (W)	Logic (W)	I/O (W)
1.956 W (97% of total)	cnn_accelerator					
1.445 W (72% of total)	sa (systolic_array)	0.562	0.551	0.011	0.884	<0.001
0.511 W (25% of total)	Leaf Cells (16)					

Utilization	Name
0.43 W (21% of total)	cnn_accelerator
0.388 W (19% of total)	detected
0.03 W (2% of total)	pixel_in
0.004 W (<1% of total)	pixel_in[0]
0.004 W (<1% of total)	pixel_in[1]
0.004 W (<1% of total)	pixel_in[2]
0.004 W (<1% of total)	pixel_in[3]
0.004 W (<1% of total)	pixel_in[4]
0.004 W (<1% of total)	pixel_in[5]
0.004 W (<1% of total)	pixel_in[6]
0.004 W (<1% of total)	pixel_in[7]
0.008 W (1% of total)	kernel_size
0.004 W (<1% of total)	kernel_size[0]
0.004 W (<1% of total)	kernel_size[1]
0.004 W (<1% of total)	clk
0 W	reset

vi. Simulation result of accelerator (along with waveforms)



The simulated Waveform of the Accelerator has been observed and Verified

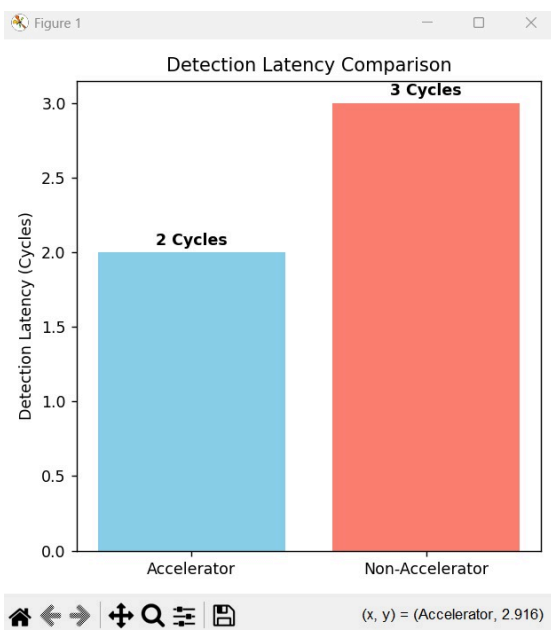
Timing Check	Count
no_clock	355
unconstrained_internal_endpoints	415
no_input_delay	11
no_output_delay	1

13 I/O Pins Used.

V. Results and Discussion

Comparison of performance of CPU and CPU+Accelerator is analysed and we ended up with 1.5x speedup in software simulation and we are expecting around 3x to 4x speedup if it is implemented and analysed with hardware benchmark.
The Comparison is illustrated below:

```
VCD info: dumpfile comparison_wave.vcd opened for output.
|    0 |    76 |       0 |        0 |         0 |          0 |
|    1 |    76 |       0 |        0 |         0 |          0 |
==============================================================
| Cycle | Input | ACC Out | ACC Detected | NoACC Out | NoACC Detected |
--------------------------------------------------------------
|    2 |    69 |      76 |        1 |         0 |          0 |
>> Accelerator detected GUNSHOT at Cycle=2
|    3 |    69 |      69 |        1 |        76 |          1 |
>> Non-Accelerator detected GUNSHOT at Cycle=3
|    4 |    65 |     145 |        1 |        76 |          1 |
|    5 |    65 |     148 |        1 |        76 |          1 |
|    6 |    12 |     210 |        1 |       148 |          1 |
|    7 |    12 |       0 |        0 |       148 |          1 |
|    8 |    14 |       0 |        0 |       148 |          1 |
|    9 |    14 |       0 |        0 |       146 |          1 |
=========================================================
| Accelerator Detection Latency = 2 Cycles (20 ns)
| Non-Accelerator Detection Latency = 3 Cycles (30 ns)
| Accelerator Speedup = 1.50 x
=========================================================
testbench.sv:87: $finish called at 165 (1s)
Done
```



vii.

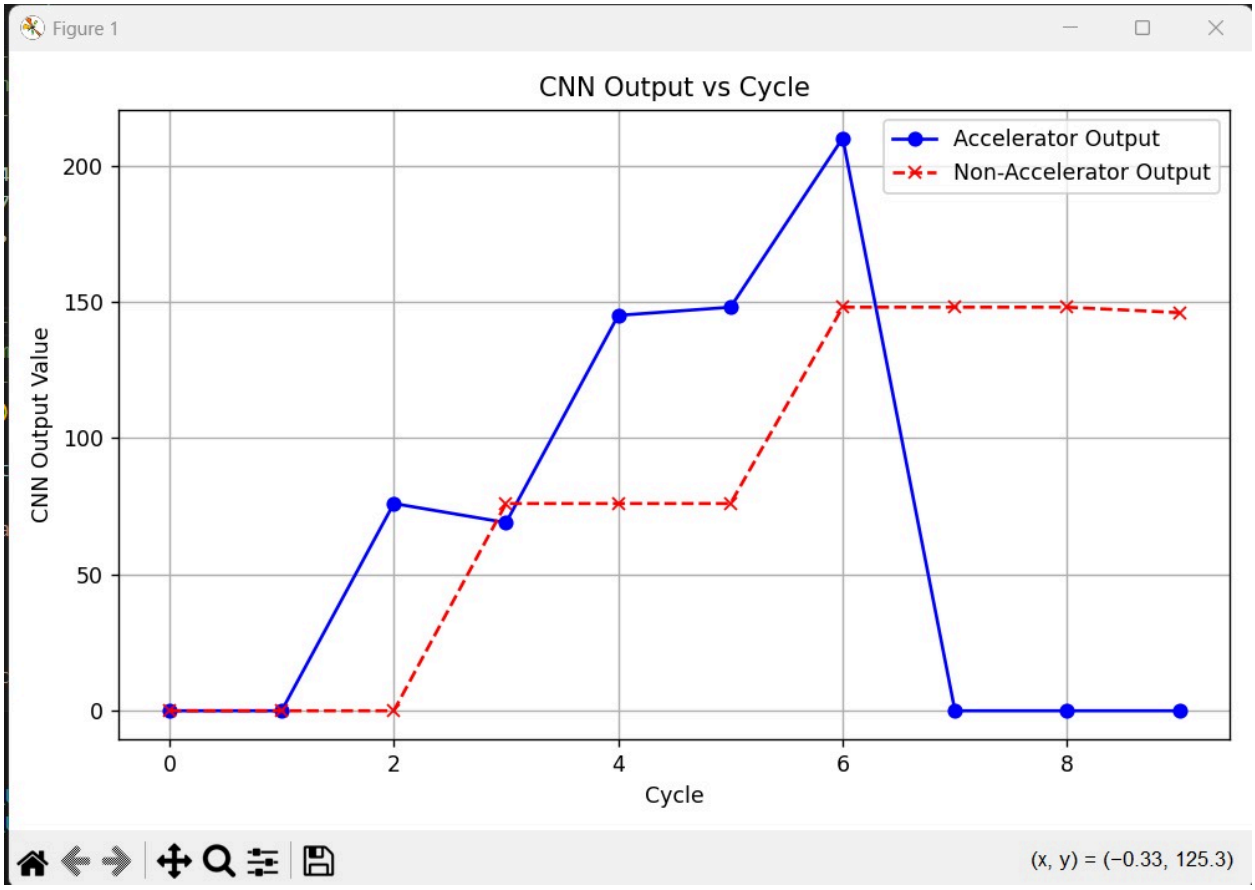


Figure 1
CNN Output vs Cycle
Accelerator Output
Non-Accelerator Output
CNN Output Value
Cycle
0
50
100
150
200
0
2
4
6
8
(x, y) = (−0.33, 125.3)

VI. Conclusion

Thus we got our model classified in 20ns compared to non-accelerated computation time of 30 ns.

We executed the function in a 2 Clock Cycle which is 1.5x faster than the Normal CPU that has a 3 clock cycle.

VII. References

- D. Grespan et al., "Gunshot Detection using Convolutional Neural Networks and Transfer Learning," in IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 2019, doi: 10.1109/MLSP.2019.8918859.
- Y. Chen, T. Luo et al., "Sparse Convolutional Neural Networks on FPGA," in Field-Programmable Custom Computing Machines (FCCM), 2019, doi: 10.1109/FCCM.2019.00034.
- A. Kalra and S. Deb, "High-Speed Wallace Tree Multiplier Design and Implementation on FPGA," in International Conference on Computing, Communication and Automation (ICCCA), 2016, doi: 10.1109/CCAA.2016.7813812.
- P. Ramesh et al., "Brent-Kung Based Parallel-Prefix Adder Design for High-Performance Digital Systems," in International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2018, doi: 10.1109/ICECA.2018.8474719.