High-Speed AXI4-Lite Dynamic MAC Accelerator for RISC-V SoC Designs

Thirunaavukarasan S[1], *Sriram N*[1], and *Rohini G*[2]

[1] Department of ECE, St. Joseph's Institute of Technology
[2] Professor (Department of ECE), St. Joseph's Institute of Technology

Abstract. This design and realization of a high-speed dynamic-precision Multiply–Accumulate (DPMAC) accelerator targets embedded and RISC-V-based System-on-Chip (SoC) platforms. Unlike conventional MAC units that operate with a fixed precision, this design supports runtime configurable 8-, 16-, and 32-bit precision modes. These modes allow accuracy, energy efficiency, and operational behaviour to be tuned according to workload requirements, while the pipeline maintains the same latency for all modes. The dynamic-precision MAC's data-path integrates Radix-4 Booth multiplier and Brent–Kung adder within a two-stage pipeline, ensuring deterministic and low-latency operation regardless of precision selection. The AXI4-Lite interface enables communication with processor subsystems through memory-mapped registers, providing flexible configuration and control of precision, start/enable logic, and accumulator behaviour. Through FPGA synthesis and testing, the proposed AXI-MAC unit demonstrates competitive timing and resource utilization while offering improved numerical flexibility compared to traditional fixed-precision MAC units. The absence of DSP slice requirements further enhances portability across devices. As such, the architecture provides an efficient, high-performance, and scalable solution suitable for next-generation embedded computing and edge-processing applications.

1 Introduction

The MAC unit functions as a core computational block in real-time DSP systems, supporting a wide range of applications such as digital filtering, image processing, embedded systems and edge computing [1]. The primary benefits of incorporating DPS in MAC units include enhanced energy efficiency, improved resource utilization, better performance and improved accuracy. Lower precision reduces switching activity, which lowers dynamic power consumption, making the design more energy efficient. Deep learning algorithms benefit greatly from hardware acceleration techniques, such as dedicated deep learning accelerators or Field-Programmable Gate Arrays (FPGAs). Efficient multipliers are critical components of these hardware accelerators, as they enable high-performance matrix multiplication operations, which are at the core of deep learning computations [3].

Software implementation of Neural Networks offers poor performance. Therefore, most real-time applications require hardware implementation. To handle high performance in computationally intensive applications, FPGAs need flexible and efficient MAC units that offer reconfigurability and power efficiency [2, 6, 7, 13, 15]. To achieve this, a MAC unit is implemented with a parallel BK adder and a signed radix-4 multiplier, which can handle up to 32 bits of data inputs. The MAC is implemented in such a way that it can switch between 8, 16, and 32-bit values based on the necessary applications. For specific workloads like DSP, Machine Learning, and AI, MAC units integrated with RISC-V offer significant benefits mainly driven by the architecture's open and flexible nature, which allows for accessing custom instruction set extensions.MAC units in various systems have shown significant results such as an increase in power and energy efficiency, low latency and high throughput. Our paper describes the importance of a parallel adder DPMAC that is integrated with a RISC-V through AMBA AXI4_Lite as the interconnect to attain such results in computation processes like ML, DL, NN, Image processing and core applications like DSP and telecommunications [4, 10, 17, 19].

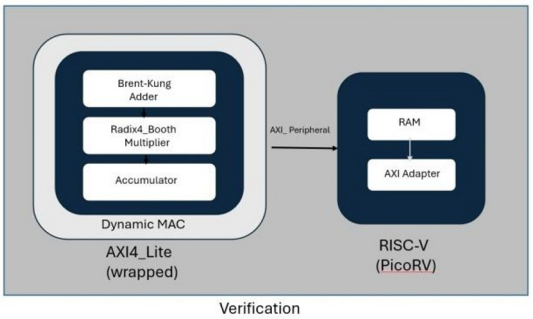

Fig.1. Block Diagram of the proposed system

Earlier approaches provide a strong foundation, but they also come with limitations that become more noticeable when applied to modern, high-demand workloads. These limitations can be crucial when dealing with processes that work at a rapid rate, and a small mistake can cause more damage. Some of these issues that we have addressed are listed below:

[1]Corresponding author: thirurajhan@gmail.com

1. **Dependencies on DSP48 Slices:** Traditional FPGA MAC accelerators heavily rely on DSP48 slices, as they provide significant benefits. However, when the workload exceeds available DSP resources, this may suffer from three major problems:
   - Time Multiplexing is necessary
   - Throughput Drops
   - Performance Collapses

These issues arise primarily from the limited number of DSP48 slices available on most widely used FPGAs, making scalable MAC designs challenging.

2. Absence of Dynamic MAC: In a fixed-precision MAC architecture, both power usage and processing latency remain essentially constant for every operation because the hardware always evaluates the full data width, even when the application does not require that level of precision. This rigidity makes static MACs well-suited for conventional DSP tasks where the precision and workload pattern do not change at run time [2, 5].

However, they perform poorly in workloads such as ML, DL, and advanced DSP applications, where either precision or timing requirements vary dynamically across operations. Since a fixed-precision MAC cannot adapt its computation width or execution flow at run time, it cannot efficiently handle these workloads and often results in unnecessary power consumption, reduced throughput, and limited flexibility.

3. Low Power Efficiencies: Earlier FPGA-based MAC designs have low power efficiency because they mainly rely on DSP48 slices and fixed-precision data paths. Even when workloads only require low precision (e.g., INT8), DSP units always compute at their full internal width (typically 25 x 18 or 27 x 18), resulting in excessive use of transistors for switching when only low precision arithmetic. All operations must use the same high bit-width due to the lack of runtime precision scaling in earlier architectures, which wastes energy across neural network layers with different precision requirements. On inexpensive FPGAs, DSP dependency further restricts parallelism, leading to increased toggle density and further decreased energy efficiency [15].

The suggested MAC architecture addresses the power, flexibility and scalability issues of earlier systems by introducing a complete DSP-free dynamic MAC system that has the following qualities as our novelty:

LUTs and FFs: Rather than depending on DSPs for every process, our MAC, along with AXI, works only with LUTs as they are abundant in FPGA's (around 50000). This helps reduce the impact on the fixed DSP blocks in FPGAs, which helps with achieving energy, power efficiency and scalability [11, 14].

Dynamic MAC (8/16/32) controlled by RISC-V: While traditional MAC units operate in a fixed bit width or as a standalone dynamic MAC, our system embeds the Dynamic MAC with RISC-V through AXI4-Lite, which makes it the optimal system for any computation ranging from low-level DSP operation to high-level neural networks.

Power Consumption: Dynamic power increases slightly (~39%) due to added configurability, but still the DSP-free DPMAC remains energy efficient because static leakage dominates total consumption.

2 Related Work

Several MAC accelerators supporting DSP and Neural Networks have been presented in recent years, where each of them has its own characteristics. Here, we discuss the accelerator architectures most relevant to our work.

2.1P.S. Locatelli et.al "Time-domain multiply–accumulate unit," Apr. 2023

This work presents a fully DSP-free, dynamically reconfigurable MAC unit with AXI4-Lite integration for RISC-V-based machine-learning and DSP workloads in order to overcome the limitations of fixed-precision, DSP-dependent MAC architectures, such as the time-domain design of Locatelli et.al [19].

While remaining compatible with standard SoC design flows, the suggested architecture greatly reduces switching activity and improves energy efficiency by enabling runtime selection of 8, 16, 32 and 40-bit precision modes.

2.2Sabreen Alsayyab et.al, FPGA Implementation of MAC Unit for Artificial Neural Network Accelerators, 2022

For neural network and DSP applications, the study offers a parameterizable floating-point MAC architecture that supports 32-bit and 64-bit IEEE-754 formats. In order to meet floating-point accuracy requirements, the design uses a partial-product tree to speed up mantissa multiplication, followed by exponent alignment, normalization, and rounding stages. The implementation shows feasible resource utilization with an operating frequency of about 46 MHz on an FPGA, and these operations are arranged into a pipelined structure appropriate for real-time processing. The work demonstrates the design considerations involved in implementing IEEE-754 compliant MAC units in reconfigurable hardware and emphasizes the value of floating-point arithmetic for high-precision workloads [15].

2.3Guillaume Devic et. al, Highly Adaptive Mixed Precision MAC Unit for Smart and Low-Power Edge Computing,2021

Recent work has proposed a mixed-precision MAC architecture designed for low-power edge computing that allows for flexible operand bit-widths in powers of two, up to 32 bits. Better use of quantized data is provided by the design, which targets resource-constrained embedded machine learning workloads and supports asymmetric precision multiplication. The MAC, which is synthesized in a 28 nm FD-SOI process, outperforms the native MAC unit of the RI5CY processor by 10% in area and 25% in dynamic power, with energy-efficiency gains of up to 50%. This work emphasizes the need for configurable MAC units that can adjust to different workloads and highlights the advantages of precision-scalable arithmetic in edge-oriented ML accelerators [16].

2.4Rangarao Orugu et al., "FPGA Design and Implementation of Approximate Radix-8 Booth Multiplier," 2024

This work introduces an approximate radix-8 Booth multiplier aimed at reducing computational complexity and improving performance in digital image-processing applications. The design incorporates modified Booth encoding and parallel partial-product generation techniques to accelerate multiplication while lowering power and area requirements compared to conventional multipliers. The architecture is evaluated on an Artix-7 FPGA using Xilinx Vivado, and its effectiveness is demonstrated through integration into a Sobel edge-detection pipeline. Experimental results highlight improvements in delay and resource usage, with the approximate computation maintaining high accuracy, achieving 98.87% edge-detection correctness. This work emphasizes the efficiency benefits of Booth-based multiplier optimizations for FPGA-based signal-processing tasks [18].

3 Proposed System

Looking at the limitations of conventional fixed-precision and DSP-dependent MAC units discussed earlier, the proposed system addresses these challenges by introducing a dynamically configurable precision MAC architecture. Unlike traditional designs, the proposed MAC supports multiple precision modes and employs a DSP-independent data-path, allowing the architecture to adapt efficiently to varying computational requirements. This flexibility makes the design suitable for both machine-learning and general embedded workloads, where precision, performance, and resource utilization must be carefully balanced. The following subsections describe the key building blocks of the system, including the BK adder, radix-4 Booth multiplier, accumulator, AXI4-Lite interface, and the RISC-V integration.

3.1 Dynamic Precision Multiply Accumulate Unit (DPMAC):

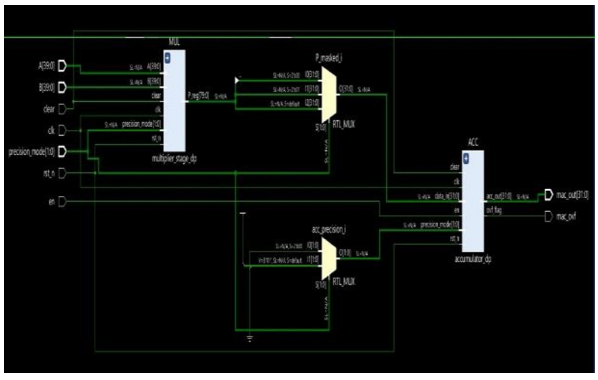

Fig.2. Schematic of DPMAC

For the processor to withstand immense computations, the MAC unit is made dynamic, where it can operate in different data-path widths ranging from 8 to 32, which is perfect to handle all types of computations.

Formula:

$$MAC(i) = \sum_{k=0}^{N-1} X[k].Y[k]$$

$$acc(i+1) = acc(i) + (x_k \,.\, y_k)$$

3.1.1 Brent-Kung Adder:

Brent-Kung adder is a parallel prefix adder, which is used in the MAC unit for making calculations by using generate (G) and propagate (P) signals in a tree-structured layout, which reduces chip area and wiring congestion. This is achieved by minimizing the number of connections between stages, leading to better performance and less area requirement for the implementation on an FPGA[9].
Time Complexity for N bits = $O(\log_2 N)$
Chip size of area for N bits = $O(N\log_2 N)$

3.1.2 Radix-4 Booth Multiplier:

The Booth multiplier was preferred over other multipliers as it reduces the number of partial products, which in turn reduces hardware, power, and the critical path inside the adder tree. In particular, Radix4 cuts the number of partial products in half, providing a good balance between performance and hardware complexity. Radix4 is more suited for FPGA implementations than Radix8 due to simpler recoding and smaller logic footprint in FPGA LUTs (Look up table) mapping. Thus, the speed up of multiplication is achieved by fewer partial products (Booth Radix4) and logarithmic carry propagation (Brent Kung adder).

3.1.3 Accumulator:

For accumulating the results produced by the BK adder during the MAC operation, while providing both intermediate values and final results, P_masked is used between the multiplier and accumulator to align the full-width multiplier output to a fixed 32-bit accumulator input by performing precision-dependent truncation or sign extension. The BK adder is wrapped inside the accumulator for operation efficiency.

Table 1: Timing Summary of the MAC

Metric	Setup	Hold	Pulse Width
Worst Slack(ns)	4.192	0.144	4.500
Total Slack(ns)	0.000	0.000	0.000
Failing Endpoints	0	0	0
Total endpoints	99	99	101

Table 2: Module-level resource utilization

Metric	Slice LUTs	Slice Registers	F7 Muxes	Bonded IOB
Mac_Top	919	100	4	103
Acc	64	68	0	0

Mul-Stage	855	32	4	0
Booth	733	0	4	0

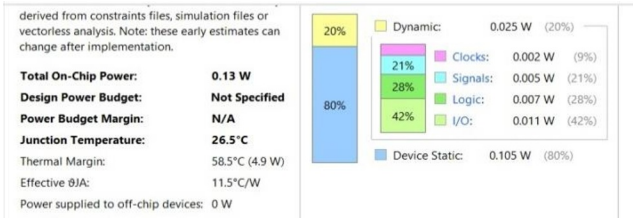

Fig.3. Power Summary of DPMAC

3.1.4 Synthesis Results:

As per the outcomes of the standalone dynamic-precision MAC unit, it can be seen that the timing constraints are comfortably met, with the worst-time slack of +4.19 ns and the critical path measuring 5.81 ns. This performs a maximum operating frequency (F-max) of ~172 MHz. And the area utilization uses 919 LUTs and 100 flip-flops, where most of them are from the multiplier, but noticeably without any DSP48 slices on both the FPGA ZedBoard and Artix-7. Therefore, it makes the design suitable and scalable for low-cost FPGA devices where DSP availability is limited or reserved for other subsystems. The overall power consumption reports about 0.13 W, with 0.105 W attributed to static leakage and only 0.025 W coming from dynamic switching activity. Since we used a two-stage pipeline (one for the multiplier and the other for the accumulator), this can accept new input pairs every cycle once the pipeline is filled. The throughput effectively becomes one MAC operation per clock cycle. At 172 MHz, this translates to approximately 172 million MAC operations per second.

3.2 AXI-Lite 4:

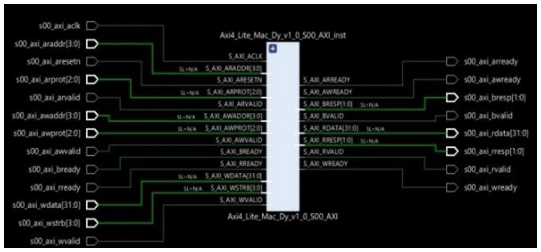

Fig.4. Schematic of AXI4-Lite

A major contribution of the proposed system is that it integrates the MAC unit through an AXI4-Lite interface, making the data path a reusable hardware accelerator compatible with SoC architectures. This AMBA-based bus provides standardized memory-mapped control registers to act as the interface with a RISC-V or any processor, making it a versatile unit to interface with. This processor can have the control to select precision modes, load operands, initiate MAC operations, and retrieve results using simple register accesses. This is enabled by the AXI channels, namely address, write data, read data, and response, which help establish proper handshaking between the MAC and the CPU/SoC or any processor. Also, the AXI4-Lite protocol supports high modularity, portability, and scalability. In addition, AXI is a handshake-driven protocol that ensures reliable operation, low-latency access, and compatibility across systems, making the overall accelerator architecture practical for modern systems that require fast computation with high efficiency.

Table 3: Timing summary of AXI

Metric	Setup	Hold	Pulse Width
Worst Slack(ns)	0.185	0.032	4,500
Total Slack(ns)	0.000	0.000	0.000
Failing endpoints(ns)	0	0	0
Total endpoints(ns)	407	407	343

Table 4: Module Level Resource Utilization

Name	LUT	Register	Mux	IOB
Axi_v1_0(Top module)	659	342	13	88
Axi4_S00_AXI	659	342	13	0
Mac_Top	601	100	13	0
Accumulator	56	68	0	0
Multiplier	545	32	13	0

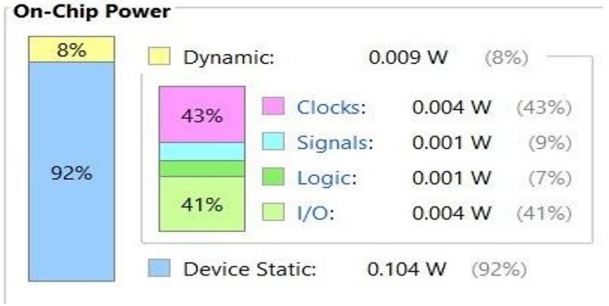

Fig.5. On-Chip Power after AXI-integration

Table 5. AXI4_Lite Address MAC

Address (hex)	Register	R/W	Description
0x00	CTRL	R/W	en, clear, status bits
0x04	A_IN	W	A operand (32-bit signed)
0x08	B_IN	W	B operand (32-bit signed)
0x0c	Result	R	MAC output (32-bit accumulator)
0x10	MODE	R/W	2-bit precision mode

Table 6. 0x10 — PRECISION Register

Bits	Meaning
00	8-bit multiply
01	16-bit multiply
10	32-bit multiply
11	Reserved (for above 32-bit with truncation)

3.2.1 Synthesis result of AXI4_Lite_MAC_Dy:

After integrating the MAC unit with the AXI4-Lite interface, the combined subsystem was synthesized again to check how the AXI wrapper affects timing, area, and power. Under a 10 ns clock constraint (100 MHz), the design meets timing with a setup slack of +0.185 ns and a critical-path delay of 9.815 ns, giving a Fmax of roughly 102 MHz. Hold timing is also clean with a WNS of +0.032 ns, showing that both the AXI control path and the MAC data-path are properly synchronized without any timing violations.

The standalone MAC used around 919 LUTs, but when Vivado synthesized the MAC together with the AXI wrapper, the tool performed cross-module optimizations and reduced the MAC footprint to 601 LUTs. So, the total subsystem now uses 659 LUTs and 342 registers. This reduction is expected because Vivado removes redundant logic only when the entire subsystem is compiled as a single hierarchy. Even after adding the AXI interface, the design still remains DSP-free.

Power also decreases after integration. Dynamic power goes down to 0.009 W because the MAC logic is optimized, and AXI-Lite communication has very low switching activity. The total on-chip power for the subsystem is about 0.113 W.

The MAC still operates with a 2-cycle latency since the pipeline structure is unchanged, and the AXI wrapper does not add extra cycles to the computation path. Once the pipeline is filled, the throughput remains one MAC output per cycle at the achieved frequency of about 102 MHz.

Table 5: MAC vs AXI-Integrated MAC

Metric	MAC-only	AXI + MAC
Latency	2 cycles	2 cycles
WNS	+4.19 ns	+0.185 ns
F- max	172 MHz	102 MHz
LUTs	919	659
Registers	100	342
Power	0.13 W	0.113 W

3.3RISC-V

For the SoC integration, we used the PicoRV32 core obtained from open-source resources, as it supports both AXI4-Lite and Wishbone for integration and instruction passing. For controlling the MAC, such as loading operands, selecting precision modes, and triggering load/store operations, we used firmware executed from external RAM. This minimalistic setup ensures deterministic control of the MAC accelerator and does not add significant overhead to the FPGA implementation. As -900this RV32I core has very low LUT usage (~750-900 LUTs), making it ideal for the SoC and FPGA implementation, which executes single-cycle load/store semantics. Using external RAM also allows firmware updates without resynthesizing the processor subsystem. This minimal RISC-V + MAC setup resembles real SoC-based accelerator architectures used in edge ML chips.

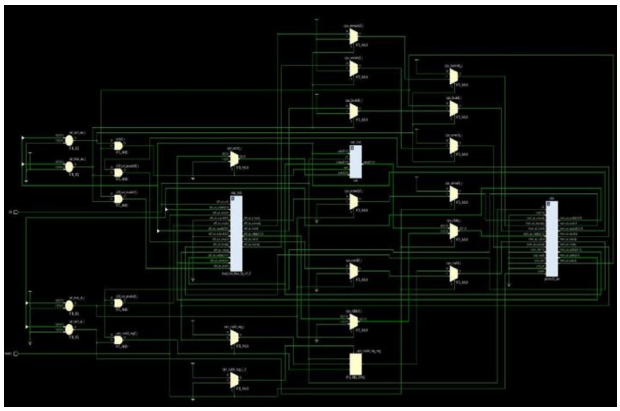

Fig.6. Integration with PicoRV_32 with Ram and Axi_mac

These parameters display the advantages of our proposed system over traditional MAC units. These advantages are crucial in modern cutting-edge technologies like DSP applications, ML, DL, NN, image processing, and many other high computational processes.

4 Simulation Results

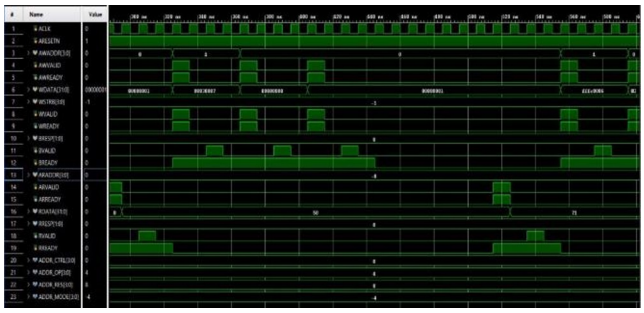

Fig.6. AXI4-Lite and MAC Waveform

From this simulation we can see theAXI4-Lite write and read transactions between the AXI slave interface and the dynamic-precision MAC. The master writes control, operand, and precision-mode registers, and later reads back the accumulated MAC result via the result register. All AXI handshakes (AWREADY, WREADY, BVALID, RVALID) complete successfully, confirming correct MAC register access and data-path response.

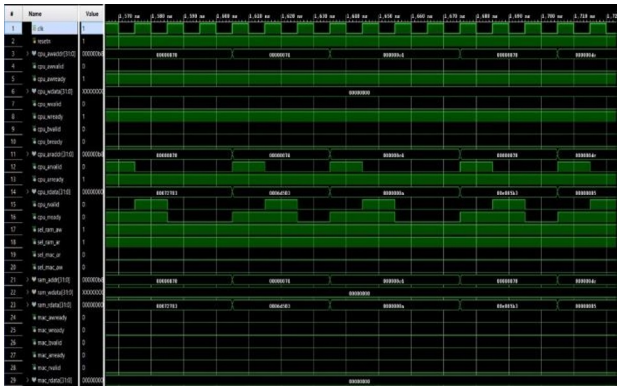

Fig.7. RISC-V instruction fetches via AXI4-Lite:

From the waveform interval between 1570ns to 1710 ns, the PicoRV32 processor preforms read transactions

to AXI4_Lite as it continues fetching instruction from the on-chip RAM. The CPU drives cpu_araddr with the next instruction addresses (0x70, 0x74, 0x0C4, 0x78, 0xDC, etc.), and each read is acknowledged by the AXI interconnect through cpu_arready. The read data returned on cpu_rdata (0x00727803, 0x00064503, 0x00e885b3, etc.) corresponds directly to RISC-V instructions stored in RAM.

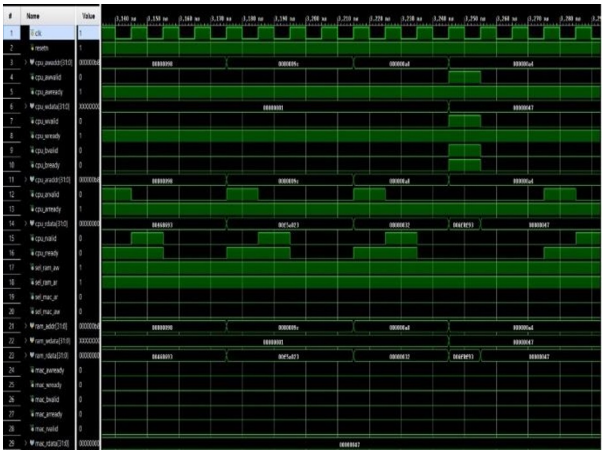

Fig.8. RISC-V Instruction Fetch via AXI4-Lite from On-Chip RAM.

Between the 3140 - 3290 ns the PicoRV32 performs continuous AXI4_Lite read transaction to fetch instruction from the RAM, which is initiated by the firmware (C to hex file). As the address increment sequentially (0x98 → 0x9C → 0xA0 → 0xA4), and each access is acknowledged with valid read data. The returned words (0x00468693, 0x00F5A023, etc.) correspond to RISC-V instructions stored in RAM. As here no MAC activity occurs in this interval which indicates that the SoC/CPU is solely executing instruction in firmware.

5. Performance Comparison between DPMAC and Static MAC

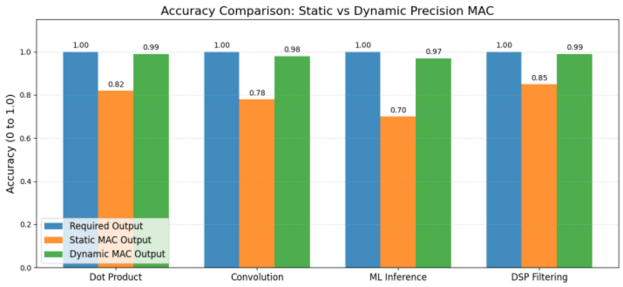

Fig.9. Accuracy Comparison of static vs dynamic mac in different applications

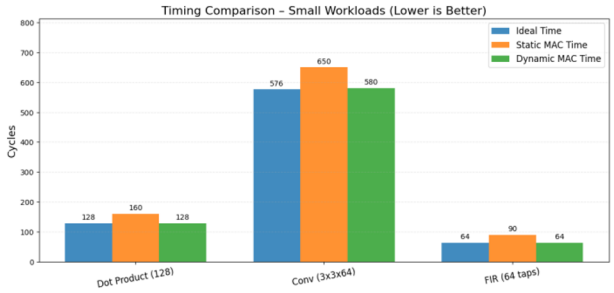

Fig.10. Timing comparison of static vs dynamic MAC in small workloads

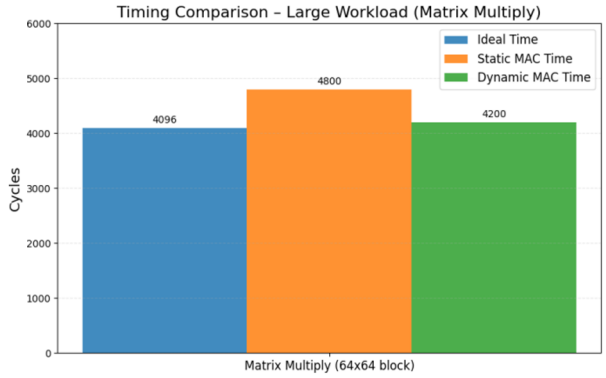

Fig.11. Timing comparison of static and dynamic MAC in large workloads

The comparison results show that the proposed Dynamic Precision MAC achieves output values that closely match the expected results across AI, ML, DSP, and general embedded workloads. By averaging multiple test inputs for dot product, convolution, ML inference, and DSP filtering, the dynamic MAC consistently maintains accuracy near the ideal output, while the static MAC shows noticeable deviations due to its fixed-precision constraints.

The timing graphs further confirm that the dynamic MAC maintains performance close to the ideal cycle counts for both small and large workloads. For matrix multiplication, convolution, FIR filtering, and dot product operations, the dynamic MAC remains significantly faster than the static MAC because it avoids unnecessary full-width computation and operates using the most suitable precision mode.

These results together demonstrate that the DPMAC offers both high accuracy and predictable timing, making it suitable for real-time applications such as drones, robotics, Kalman filtering, and DSP-based signal processing, where precision and speed are equally important.

6. Conclusion

This paper presents the design and implementation of an AXI-based DPMAC unit integrated with the RISC-V architecture for high performance in SoC environments. The use of AXI4-lite helped us achieve seamless integration between RISC-V and the DPMAC, which resulted in the design providing reliable data transfers,

ensuring reliable latency and ease of integration into every SoC architecture existing so far.

The key enhancement of the proposed design is the use of Dynamic Precision MAC over standard MAC, as it adapts its computational precision based on the requirements of the applications. The DPMAC unit present in the core of the accelerator architecture is designed to support high-level computations for applications like DSP and embedded systems. The dynamic precision has a huge impact in applications like ML, NN and adaptive signal processing, where complete precision is not always the requirement. The system reduces operand precision when high accuracy is not required by minimizing switching activity, which leads to lower power consumption without significantly compromising result quality.

The dynamic control of the MAC further strengthens its versatility by configuring operand width, accumulation depth and precision control logic, making the design scalable across a range of performance and resource constraints in both FPGA and ASIC implementation [9].

Verification of the proposed design confirmed the correct operation of AXI, dynamic precision control and MAC data-path functionality. Simulation results validated proper handshaking between RISC-V and the MAC unit as well as accurate computation across different precisions.

References

1. R. Nivetha, K. Ranjani, S. Santhiya, Student attendance system using RFID. *Int. J. Res. Eng. Sci. Manag.* **2**, 234–238 (2020)
2. H. Supriyanto, I. Rokhim, V.E. Polman, Smart helmet for monitoring construction workers using IoT-based ESP32 and LoRa. *Int. J. Comput. Sci.* **12**, 45–52 (2024)
3. M.N.V. Mangala, S. Priyanka, A. Sharmila, IoT based smart helmet for ensuring safety in industries. *Int. J. Eng. Res. Technol.* **7**, 1–5 (2018). https://www.ijert.org/iot-based-smart-helmet-for-ensuring-safety-in-industries
4. B. Swamy, L. Ramesh, K.G. Krishna, IoT-enabled wearable sensor network for real-time industrial worker safety and health monitoring. *J. Emerg. Technol. Trends* **6**, 120–128 (2021)
5. T. Prajapati, N. Bhatt, D. Mistry, A survey paper on wearable sensors based fall detection. *Int. J. Comput. Appl.* **113**, 22–27 (2015)
6. V.I.L. Vishala, R. Srihari, S. Kumar, An IoT based wearable system for the safety of workers in industrial scenario. *Int. J. Adv. Res. Commun. Comput. Eng.* **13**, 56–62 (2024)
7. A. Tiwari, R. Kumar, Wearable IoT-based smart safety helmet for hazardous environment monitoring. *IEEE Sens. J.* **21**, 15920–15928 (2021). https://doi.org/10.1109/JSEN.2021.3076543
8. A.S. Prasad, R. Rao, V. Reddy, Real-time fall detection using IMU sensors and machine learning for worker safety. *IEEE Access* **10**, 120450–120460 (2022). https://doi.org/10.1109/ACCESS.2022.3224567
9. D. Zhang, H. Wang, J. Li, A wearable IoT system for real-time worker health and safety monitoring in smart industry. *IEEE Internet Things J.* (2023). https://doi.org/10.1109/JIOT.2023.3321456
10. P. Mohan, V. Bharathi, C. Nandhini, Smart helmet for industrial safety using IoT. In *Proc. IEEE Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, 480–485 (2020).
11. M.N. Islam, A. Rahman, K. Andersson, A smart worker's helmet using IoT for hazardous event detection.In *Proc. IEEE Int. Conf. Ind. Eng. Eng. Manag. (IEEM)*, 820–825 (2019).
12. K. Ishaq, S. Bibi, IoT based smart attendance system using RFID: A systematic literature review.arXiv (2023). https://arxiv.org/abs/2308.02591
13. T.B. Aderinola, S. Stein, M.A.A. Rodrigues, Watch your step: A cost-sensitive framework for accelerometer-based fall detection in real-world streaming scenarios. arXiv (2025).
14. K.C. Liu, Y.T. Lin, C.W. Hung, Deep learning based signal enhancement of low-resolution accelerometer for fall detection systems. arXiv (2020). https://arxiv.org/abs/2006.12345
15. M. Jalan, RFID-based attendance system design for educational institutes. *Int. J. Res. Mech. Electr. Eng. Technol.* **3**, 1–5 (2015)