Journal of Harbin Engineering University
ISSN: 1006-7043

Vol. 47 No. 07
July 2026

Design of State Transition Graph Based Low Power Sequential Multiplier: RTL-to-GDS II Flow

Aishwarya[1], Sujatha K[2], Dr. Kiran Bailey[3]
Department of Electronics and Communication Engineering,
BMS College of Engineering, Bengaluru, Karnataka, India.
Department of Electronics and Communication Engineering,
BMS College of Engineering, Bengaluru, Karnataka, India.
Department of Electronics and Communication Engineering,
BMS College of Engineering, Bengaluru, Karnataka, India.

Abstract

The reduction of energy has been a significant problem in the design of modern VLSI circuits, in particular for the multiplier circuits that are extensively used in DSP, communication and embedded computing applications. This work proposes a low power Predictive Activity-Gated State Transition Graph (PAG-STG) sequential multiplier that reduces unnecessary switching activity for low dynamic power dissipation. The proposed architecture incorporates an Activity Prediction Unit (APU) and fine grain datapath gating logic to activate the arithmetic blocks only when the computation is required. The multiplier was designed in Verilog HDL and realized through a full RTL-to-GDSII design flow using Cadence Genus and Innovus tools in 45 nm technology. The design was verified through full ASIC implementation flow which includes functional simulation, synthesis, timing verification, physical design and sign-off analysis. Experimental results show that, the PAG-STG multiplier achieves significant power efficiency and silicon utilization improvement without loss of computational accuracy. By minimizing redundant signal transitions, the proposed PAG-STG multiplier achieves improved energy utilization and becomes a practical solution for power-conscious ASIC and VLSI designs.

Keywords: low-power VLSI, sequential multiplier, state transition graph (STG), predictive activity gating (PAG), activity prediction unit (APU), dynamic power reduction, RTL-to-GDSII Flow, ASIC design, fine-grain datapath gating, 45 nm technology.

1. Introduction

Multiplication forms the basis of many arithmetic computations in modern digital hardware. Applications such as DSP processors, communication devices, image-processing systems, and embedded platforms rely heavily on efficient multiplier architectures. Multiplication is a core operation in digital hardware, and the design of the multiplier directly affects processing speed, silicon area, and energy consumption of the overall system. As modern VLSI circuits become more complex and portable electronic devices continue to grow in popularity, achieving lower power operation has emerged as an important design requirement. The conventional sequential multipliers based on State Transition Graph (STG) employ structured control by Finite State Machine (FSM) and use sequential addition and shift operations to carry out multiplication. However, arithmetic datapath blocks are active for each clock cycle, resulting in unnecessary switching activity and higher dynamic power consumption. To overcome this drawback, a Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier is proposed where the arithmetic resources are activated only when the computation is predicted, thereby reducing unnecessary signal transitions. The Activity Prediction Unit (APU) analyzes future computation needs and dynamically manages the activation of arithmetic blocks based on the processing demand. The architecture was designed in Verilog HDL and implemented through full ASIC implementation flow from RTL to GDSII generation using Cadence Genus and Innovus in 45 nm CMOS technology environment. The experimental evaluation shows that the design consumes less energy, while maintaining functional accuracy, and meeting the physical design constraints related to timing and silicon utilization.

2. Literature survey

[1] Sequential multipliers based on State Transition Graphs (STGs) have been utilized extensively in VLSI systems because of its effective hardware consumption and organized control mechanism. The multiplication

process is controlled using a Finite State Machine (FSM), where mathematical procedures are carried out sequentially through state transitions. Although the architecture reduces hardware complexity, arithmetic datapath blocks remain active during every clock cycle, resulting in increased switching activity and higher dynamic power consumption.

[2] Booth multiplier architectures were introduced to improve multiplication speed by reducing the quantity of partial products generated during multiplication. The Booth recoding technique minimizes arithmetic operations and improves computational efficiency. However, the additional encoding and control circuitry increase design complexity and may contribute to higher power consumption in low-power applications.

[3] Dadda and Wallace Tree multipliers provide high-speed multiplication through parallel reduction of partial products. These architectures significantly reduce propagation delay and improve overall performance. However, the use of large numbers of adders and reduction stages increases silicon area utilization and switching activity, making them less appropriate for power-constrained VLSI systems.

[4] Several low-power techniques such as Clock Gating, Operand Separation, as well as the Spurious Power Suppression Method (SPST) have been proposed to reduce dynamic usage of power. These methods reduce unnecessary signal transitions by disabling inactive circuit blocks. Although power efficiency is improved, additional control circuitry introduces design overhead and implementation complexity.

[5] Recent research has centered on activity-aware and power-efficient arithmetic architectures for VLSI applications. Nevertheless, little work has been reported on reducing switching activity in State Transition Graph based sequential multipliers. To address this limitation, the proposed Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier introduces activity prediction and fine-grain datapath gating to selectively enable arithmetic operations only when computation is required, thereby enhancing power efficiency while preserving correct multiplier functionality.

3. Existing STG Multiplier Architecture

The existing State Transition Graph (STG) based sequential multiplier architecture uses a finite state machine. The multiplication process is managed using a finite state machine (FSM). through multiple operational states. The FSM controls operations such as loading inputs, checking multiplier bits, generating partial products, accumulation, shifting, and final output generation. The process of multiplication is carried out sequentially, where one multiplier bit is processed during each clock cycle.

In this architecture, the unit of control checks the multiplier's least significant bit (LSB) during every clock cycle. In the event that the multiplier bit is ‘1’, the multiplicand is added to the accumulator to obtain the partial product. Only the shift operation is carried out and the addition operation is omitted if the multiplier bit is ‘0’. Until all multiplier bits are processed and the final multiplication result is produced, the FSM iteratively goes through several phases. This sequential state-based operation provides organized datapath control and efficient hardware utilization.

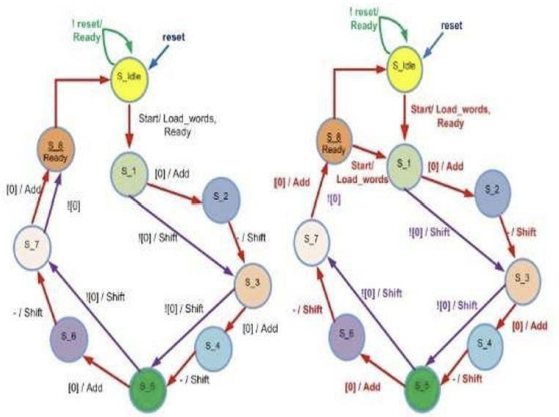

Fig. 1. Existing STG multiplier control unit.

Figure 1 shows the state transition graph representing the sequence of states during multiplication. State transitions are controlled by the value of the current multiplier bit, enabling operations such as addition and shifting. Although this approach provides systematic control, all datapath blocks remain active during computation, leading to increased power usage and switching activity.

A. Limitations of Conventional STG Multiplier

The normal STG multiplier does not distinguish between active and inactive computation cycles. Hence the arithmetic datapath keeps switching even though no useful computation is required. Such unnecessary switching activity increases dynamic power consumption and reduces overall energy efficiency. Also, the architecture does not have any activity prediction or datapath gating mechanism to reduce unnecessary signal activity during multiplication.

4. Proposed Methodology

The proposed work presents a low power VLSI application of a Predictive Activity-Gated State Transition Graph (PAG-STG) sequential multiplier. The proposed architecture aims at reducing the dynamic power consumption by reducing the unnecessary switching activity in multiplication operations. Unlike conventional STG multipliers where all arithmetic datapath blocks are active throughout the multiplication process, in the proposed architecture only the required hardware components are selectively activated according to the predicted computational activity. This method decreases the number of the needless signal transitions and enhances the overall power efficiency, while maintaining the correct multiplication functionality.

A. Proposed PAG-STG Architecture

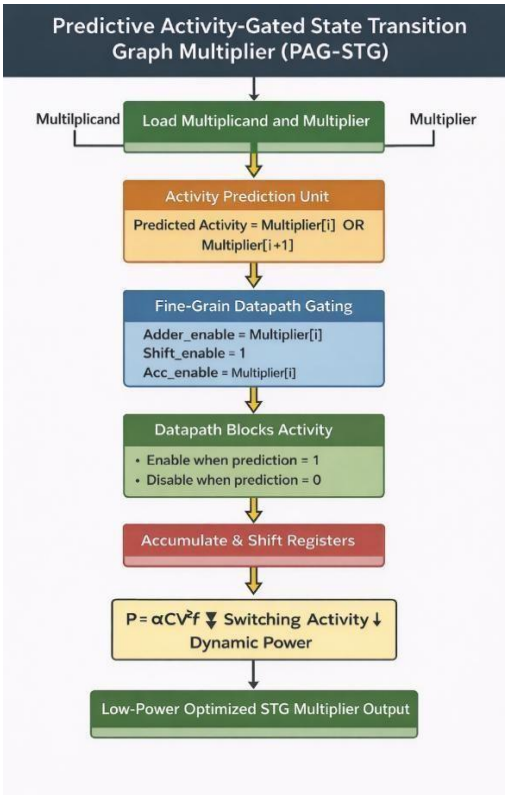

Fig. 2. Proposed PAG-STG multiplier architecture.

Fig. 2. illustrates the architecture of the proposed Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier. The architecture consists of an Activity Prediction Unit (APU), fine-grain datapath gating logic, adder, accumulator, shift registers, control FSM, and output register. The FSM controls the sequential multiplication process, while the Activity Prediction Unit predicts upcoming arithmetic activity and generates control signals for selective datapath activation. Considering the prediction result, arithmetic blocks are only activated when computation is required. This selective activation minimizes unnecessary activity switching and reduces dynamic power usage.

B. Limitations of Conventional STG Multiplier

The process of multiplication begins by loading the multiplicand and multiplier values into their respective registers. These values act in the input operands regarding the sequential procedure of multiplication. The registers store the data required for generating partial products during the multiplication operation.

The multiplication process begins by loading the multiplicand and multiplier values into their respective registers. The multiplication operation is performed through sequential partial product accumulation.

$$product = \sum_{i=0}^{n-1} (B_i \times A \times 2^i)$$

where:

- A= Multiplicand
- B_i= i-th bit of multiplier
- 2^i= Shift operation
- n= Number of multiplier bits

C. Activity Prediction Unit

The activity prediction unit determines whether the arithmetic datapath blocks should be activated during each multiplication cycle. The prediction is generated using the current multiplier bit and the next multiplier bit. Based on the predicted activity, enable signals are generated to control the datapath blocks, thereby reducing unnecessary switching activity and dynamic power consumption.

The equation:

$$P[i] = B[i] \text{ OR } B[i+1]$$

where:

- $P[i]$= Predicted activity signal
- $B[i]$= Current multiplier bit
- $B[i+1]$= Next multiplier bit

The Activity Prediction Unit predicts whether arithmetic computation is necessary in the next clock cycle. Considering the multiplier bit, the datapath blocks are either enabled or disabled.The activity

prediction logic used to generate the prediction signal is summarized in Table I.

Enable = Multiplier Bit where:

Multiplier Bit = 1 → Arithmetic units enabled
Multiplier Bit = 0 → Arithmetic units disabled

Table I. Activity Prediction Logic Used in Proposed Architecture

Activity Prediction Logic		
Multiplier[i]	***Multiplier[i+1]***	***Predicted Activity***
0	0	0
0	1	1
1	0	1
1	1	1

D. Fine-Grain Datapath Gating

To lower dynamic power consumption even more, the proposed architecture employs a fine-grain datapath gating mechanism. Considering the predicted activity signal generated by the Activity Prediction Unit, enable signals are produced for the arithmetic datapath blocks. The adder and accumulator are activated only when arithmetic computation is required, while inactive hardware blocks remain disabled. This significantly reduces unnecessary signal transitions within the datapath.

Considering the prediction outcomes of the APU, fine-grain gating logic selectively enables or disables the datapath components like the adder, accumulator, and shift registers. This gating system guarantees that only the required hardware blocks operate during each clock cycle.

When the multiplier bit is '1', the partial product is generated as:

$$PP_i = A \times B_i$$

If:

$$B_i = 1$$

Then:

$$PP_i = A$$

Otherwise:

$$PP_i = 0$$

This selective generation of partial products performs arithmetic operations only when required. As a result, unnecessary switching activity is reduced.

Fig. 3. shows the fine-grain datapath gating mechanism of the suggested PAG-STG multiplier. Considering the predicted activity signal, enable signals are generated to selectively activate the adder, accumulator, and shift register. Inactive datapath blocks remain disabled, thereby reducing unnecessary signal transitions and lowering dynamic power consumption.

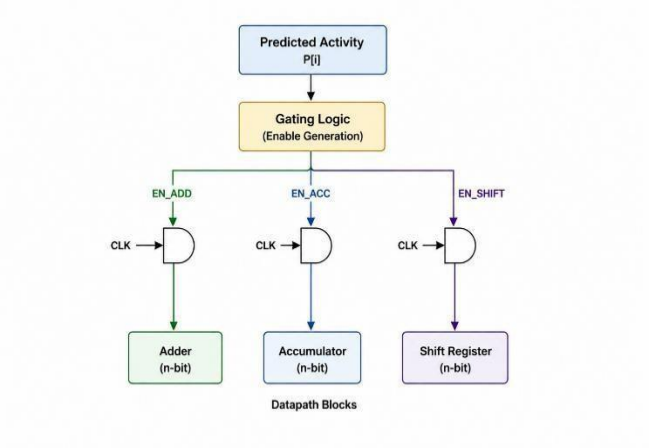

Fig. 3. Fine grain datapath gating mechanism

E. Datapath operation

The datapath blocks are responsible for performing the arithmetic operations required during multiplication. Their operation is managed by the enable signals produced from the activity prediction and gating logic. When computation is required, the adder, accumulator, and shift registers become active and process the data normally. If no useful operation is needed, these blocks remain disabled to prevent unnecessary switching activity. This selective activation increases power efficiency while preserving correct multiplication function.

Accumulation Equation:

The accumulator stores intermediate multiplication results at each clock cycle and is as follows:

$$ACCnew = ACCold + PPi$$

where:

- $ACCold$= Previous accumulator value
- PPi= Current partial product.

F. Shift Operation equation

During each clock cycle, the accumulator stores the partial product, while the shift registers update the multiplier bits. This sequential procedure keeps on until every multiplier bit has been processed and the final product is obtained.

Shift Operation Equation:

After each multiplication cycle, the multiplicand and multiplier registers are shifted as:

$Multiplicand \leftarrow Multiplicand << 1$

$Multiplier \leftarrow Multiplier >> 1$

where:

• (<<1) = Left shift operation

• (>>1) = Right shift operation

The left shift operation aligns the multiplicand for the next partial product generation, while the right shift operation updates bits of the multiplier sequentially. This procedure is repeated until all multiplier bits are processed and the final product is obtained.

Table II. Multiplication Operation

Cycle	*Multiplier Bit*	*Adder*	*Shift*
1	1	ON	YES
2	0	OFF	YES
3	1	ON	YES
4	0	OFF	YES

Table Ⅱ illustrates the operation of the proposed PAG-STG multiplier during successive clock cycles. Based on the multiplier bit, the adder is either enabled or disabled, while the operation of the shift continues in each cycle until the process of multiplication is completed.

G. Dynamic Power Reduction

The primary objective of the suggested architecture is to reduce dynamic power use by minimizing switching activity inside arithmetic datapath blocks. Dynamic power use in CMOS circuits is represented by:

$Dynamic(p) = \alpha CV^2 f$

where:

α = Switching activity factor

C = Load capacitance

V = Supply voltage

f = Operating frequency

By selectively enabling arithmetic units only during active computation cycles, the proposed PAG-STG architecture minimizes the switching activity factor (α). Consequently, unnecessary signal transitions are minimized, resulting in lower dynamic power consumption and improved power efficiency compared to conventional STG multiplier architectures.

5. Results and Analysis

A. RTL Simulation Waveform

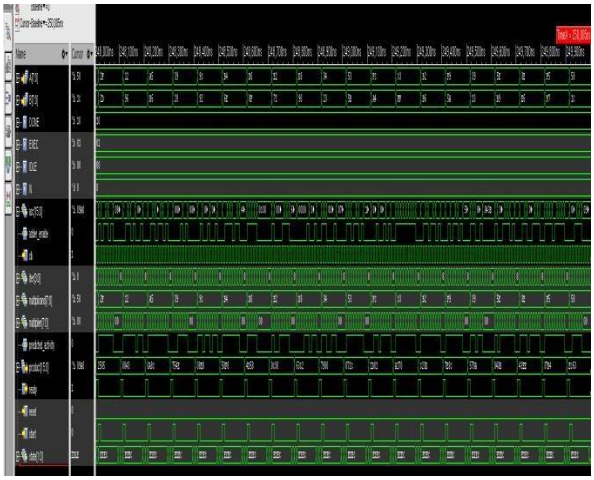

Fig. 4. Functional simulation waveform.

Fig. 4. shows the functional simulation waveform of the proposed PAG-STG multiplier. The reset signal initializes the internal registers, and the FSM is in the IDLE state at first. Upon assertion of the start signal, the multiplicand A[7:0] and multiplier B[7:0] are loaded, the multiplication process is started with the EXEC state. The predicted activity signal is used to control the adder_enable signal in each clock cycle to perform the arithmetic operations only when necessary. The accumulator and the temporary registers are incrementally updated and each multiplier bit is considered individually. During the computation, partial products are accumulated to form the product [15:0] output. When all the multiplier bits are processed, then DONE and ready signals get active which shows the successful completion of multiplication and the correct operation of proposed PAG-STG architecture.

B. Synthesis Results

Fig. 5. illustrates the optimized gate-level netlist of the proposed PAG-STG multiplier obtained after synthesis. The RTL design was converted into a technology-mapped implementation and optimized for timing, power, and area performance. The generated netlist preserves the activity-aware gating functionality and confirms the readiness of the design for physical implementation.

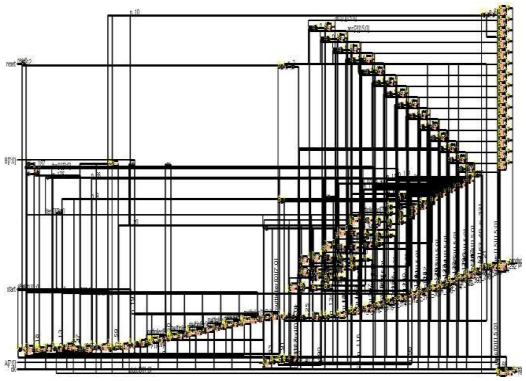

Fig. 5. Optimized gate-level netlist generated after synthesis.

C. Physical Design Implementation

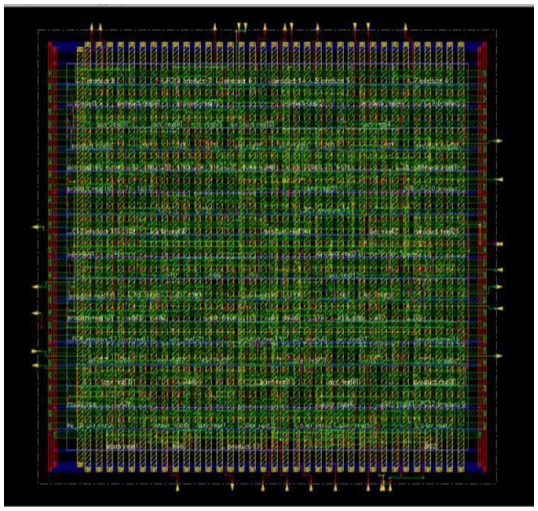

Fig. 6. Final Routed Layout.

Fig. 6. shows the final routed layout of the proposed Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier obtained after placement, clock tree synthesis, and detailed routing. The routing process establishes all signal, clock, and power interconnections required for circuit operation while satisfying physical design constraints. The successful completion of routing confirms that all datapath and control blocks are correctly connected without routing violations. Finally, the generated layout demonstrates the physical realizability of the proposed architecture in 45 nm CMOS technology and confirms its applicability for Application Specific Integrated Circuit (ASIC) implementation.

The detailed routing was performed by NanoRoute routing engine to achieve full signal connectivity and meet the physical design requirements.

D. Post-Layout Timing Analysis

```
Path 1: MET (6460 ps) Setup Check with Pin acc_reg[15]/CK->D
          Group: clk
     Startpoint: (R) multiplicand_reg[0]/CK
          Clock: (R) clk
       Endpoint: (R) acc_reg[15]/D
          Clock: (R) clk

                    Capture       Launch
       Clock Edge:+   10000            0
       Src Latency:+      0            0
       Net Latency:+      0 (I)        0 (I)
           Arrival:=  10000            0

             Setup:-    122
       Uncertainty:-    200
     Required Time:=   9678
      Launch Clock:-      0
         Data Path:-   3218
             Slack:=   6460

#-------------------------------------------------------------------------------------------------
#      Timing Point       Flags    Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                          (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
```

Fig. 7. Timing report.

Fig. 7. shows the post-layout timing analysis report of the proposed Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier. The critical timing path has a required arrival time of 9678 ps and a data path delay of 3218 ps, leading to a positive slack of 6460 ps. The timing status is reported as MET, which means that after physical implementation, all timing constraints are successfully met. The positive slack confirms the successful timing closure and reliable operation of the proposed architecture.

E. Post-Layout Power Analysis

```
Power Unit: W
PDB Frames: /stim#0/frame#0
-------------------------------------------------------------------------------
  Category        Leakage     Internal    Switching       Total    Row%
-------------------------------------------------------------------------------
    memory    0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  register    8.97938e-09  2.24302e-05  9.82830e-07  2.34220e-05  46.67%
     latch    0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     logic    1.09640e-08  6.26282e-06  1.94709e-05  2.57447e-05  51.30%
      bbox    0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     clock    0.00000e+00  0.00000e+00  1.02060e-06  1.02060e-06   2.03%
       pad    0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        pm    0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
-------------------------------------------------------------------------------
  Subtotal    1.99434e-08  2.86930e-05  2.14743e-05  5.01872e-05 100.00%
Percentage         0.04%       57.17%       42.79%      100.00% 100.00%
-------------------------------------------------------------------------------
@genus:root: 9>
```

Fig. 8. Power report.

Fig. 8. shows the post-layout power analysis report of the proposed PAG-STG multiplier. The total power consumption of the design is 5.01872 × 10^{-5} W (50.19 μW) with leakage, internal and switching power components. The reduced switching power indicates that the predictive activity-gating mechanism is effective in reducing unnecessary signal transitions and improving power efficiency.

F. Post-Layout Area Analysis

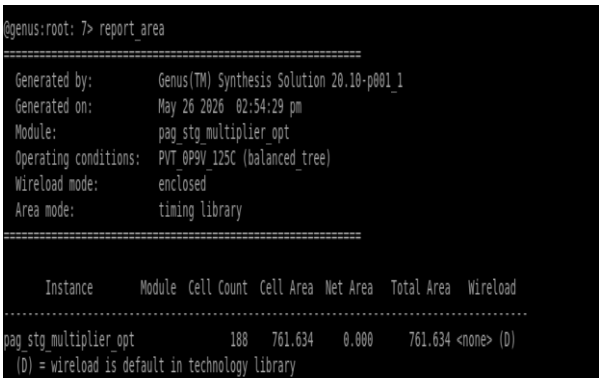

Fig. 9. Area report.

Post-layout area analysis of the proposed Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier is shown in Fig. 9. The synthesized design occupies total area of 761.634 um^2 using 188 standard cells. The results show that the suggested architecture provides efficient hardware usage and a compact design that can be used in low-power VLSI applications. The obtained area report shows that the proposed PAG-STG multiplier is area efficient without impacting its overall functionality.

G. Physical Verification

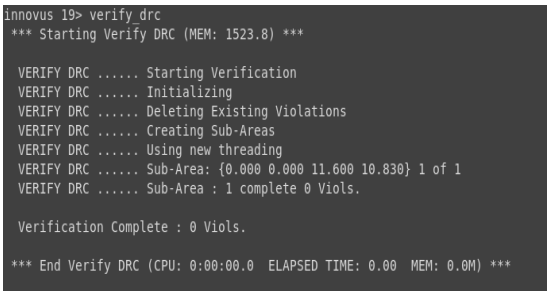

Fig. 10. DRC report.

The design rule check (DRC) report of the proposed PAG-STG multiplier is shown in Fig. 10. Final routed layout DRC verification was performed to verify that all fabrication rules concerning spacing, width, enclosure and routing constraints are met. The report shows zero violations confirming the correctness, manufacturability and physical validity of the implemented layout.

H. Performance Comparison.

Table Ⅲ shows the performance comparison between the conventional STG multiplier and the proposed PAG-STG multiplier in 45 nm CMOS technology. The proposed architecture achieves area reduction from 1026 um^2 to 761.634 um^2 and power reduction from 85.4 uW to 50.19 uW. These improvements are achieved by a predictive activity-aware gating that reduces the unnecessary switching activity in the datapath. The proposed architecture works correctly and gives significant improvements in power efficiency and hardware utilization, even though the multiplication is done sequentially.

Table III. Performance Evaluation of Existing STG and Proposed PAG-STG Multipliers.

Parameter	***STG Multiplier (45nm)***	***Proposed PAG-STG Multiplier(45nm)***
Area (μm^2)	1026	761.634
Power (μW)	85.4	50.19
Delay	0.035 ns	3.218 ps

6. Conclusion

This paper presents a Predictive Activity-Gated State Transition Graph (PAG-STG) multiplier for low-power VLSI applications using the RTL-to-GDSII design flow. The design we propose consists of an Activity Prediction Unit (APU) integrated with the datapath control logic to activate the arithmetic resources only when they are active in computation, thereby minimizing redundant signal transitions. Functional correctness of the design was validated using simulation. The design was synthesized using Cadence Genus and placed and routed using Cadence Innovus in a 45 nm CMOS process. The post layout analysis showed an area of 761.634 um^2 with power consumption of 50.19 uW, a data path delay of 3.218 ns, along with a positive timing slack of 6.460 ns, along with positive timing slack thereby achieving the timing closure. The proposed PAG-STG design achieves a better hardware efficiency in terms of area occupation and power consumption over the baseline STG multiplier without sacrificing the multiplication accuracy. The results obtained show that

predictive activity control is a feasible technique for the design of low power and energy conscious VLSI architectures.

References

[1] S. Padhy, L. Mounika, P. M. Varaprasad, and Y. Roshit, "Physical design (RTL-GDSII) implementation of state transition graph-based sequential multiplier," in Proc. 2nd Int. Conf. Signal Process., Commun., Power Embedded Syst. (SCOPES), 2024, pp. 1–6. https://ieeexplore.ieee.org/document/10782934

[2] S. Padhy, P. K. Patnaik, A. Rath, S. Pine, and R. K. Misra, "Design and synthesis of state transition graph based area efficient sequential multiplier," in Proc. 2nd Int. Conf. Networking, Embedded Wireless Syst. (ICNEWS), 2024, pp. 1–6. https://ieeexplore.ieee.org/document/10729853

[3] S. Pine, R. Sharma, R. Adhikary, S. Mishra, A. Sungheetha, and G. S. P. Ghantasala, "Design and synthesis of state transition graph-based sequential multiplier for fast computing operation," in Proc. 3rd Odisha Int. Conf. Electr. Power Eng., Commun. Comput. Technol. (ODICON), 2024, pp. 1–6. https://ieeexplore.ieee.org/document/10717336

[4] T. Gaurav, K. Patel, and R. Parekh, "RTL to GDSII implementation of RADIX-4 booth multiplier," in Proc. IEEE Int. Conf. Nanoelectron., 2022, pp. 1–5. https://ieeexplore.ieee.org/document/9992894

[5] G. Kanase and K. B. Sowmya, "Physical implementation of shift register with respect to timing and dynamic drop," in Proc. 5th Int. Conf. Commun. Electron. Syst. (ICCES), 2020, pp. 120–124. https://ieeexplore.ieee.org/document/9143360

[6] R. Jain and S. De, "RTL to GDSII implementation of RADIX-4 booth multiplier using open-source EDA tools," in Proc. 1st Int. Conf. Circuits, Power Intell. Syst. (CCPIS), 2023, pp. 1–4. https://ieeexplore.ieee.org/document/10169759

[7] S. Sabeetha, J. Ajayan, S. Shriram, K. Vivek, and V. Rajesh, "A study of performance comparison of digital multipliers using 22 nm strained silicon technology," in Proc. 2nd Int. Conf. Electron. Commun. Syst. (ICECS), 2015, pp. 180–184. https://ieeexplore.ieee.org/document/9904050

[8] S. Kumar, V. Jain, S. G. Sarode, and P. P. Thaker, "Modem ASIC design with minimal power consumption for the IDU of a SATCOM terminal: RTL to GDS II," in Proc. 3rd IEEE Int. Conf. Recent Trends Electron., Inf. Commun. Technol. (RTEICT), 2018, pp. 1668–1675. https://ieeexplore.ieee.org/document/8742655

[9] D. Umesh and S. G. Nayak, "High speed signed 8-bit multiplier using booth encoding and dadda tree architecture," in Proc. 5th Int. Conf. Intell. Technol. (CONIT), 2025, pp. 1–5. https://ieeexplore.ieee.org/document/10909258

[10] T. G. Chaithra, S. K. PradeepKumar, K. Nandini, S. V. Harshitha, and K. R. Ankitha, "ASIC realization and performance evaluation of 64×64 bit high speed multiplier in CMOS 45 nm using Wallace tree," in Proc. 2nd IEEE Int. Conf. Recent Trends Electron., Inf. Commun. Technol. (RTEICT), 2017, pp. 1115–1119. https://ieeexplore.ieee.org/document/8369305

[11] S. Maurya, "Design of RTL synthesizable 32-bit FIFO memory," Int. J. Eng. Res. Technol., vol. 5, no. 11, 2016. https://www.ijert.org/design-of-rtl-synthesizable-32-bit-fifo-memory

[12] H. V. R. Aradhya and G. Kanase, "RTL to GDSII of Harvard structure RISC processor," in Proc. IEEE Int. Conf. Electron., Comput. Commun. Technol. (CONECCT), 2021. https://ieeexplore.ieee.org/document/9677234

[13] S. Hesham, M. Shalan, M. W. El-Kharashi, and M. Dessouky, "Digital ASIC implementation of RISC-V: OpenLane and commercial approaches in comparison," in Proc. IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS), 2021, pp. 498–502. https://ieeexplore.ieee.org/document/9531738

[14] V. Padmakumar, T. M. Ignatius, T. B. Singha, and R. P. Palathinkal, "A serial-parallel-based 4-bit novel multiplier: Design, implementation, and performance analysis," in Proc. IEEE Silchar Subsection Conf. (SILCON), 2023, pp. 1–6. https://ieeexplore.ieee.org/document/10126039

[15] A. Chauhan, A. K. Meena, and A. Kumar, "Performance analysis of 4-bit multiplier using 90 nm technology," in Proc. 2nd Int. Conf. Intell. Technol. (CONIT), 2022, pp. 1–5. https://ieeexplore.ieee.org/document/9904114

[16] D. Jessintha, V. N. M, and S. S, "A two-speed, radix-4, parallel-parallel multiplier," in Proc. IEEE Int. Conf. Inf. Technol., Electron. Intell. Commun. Syst. (ICITEICS), Bangalore, India, 2024. https://ieeexplore.ieee.org/document/8658357