DESIGN OF GENERIC SQUARE ROOT COMPUTING ARCHITECTURES ON FPGA USING VHDL

ADE Mini Project

27/12/2024

DO Thu Giang - NGUYEN Phuong Linh

Contents

1 Behavior approach

The codes used in this section and the following section are available at the following GitHub Repository.

1.1 Sequential Design Using Newton's Method

This sequential architecture a1 is designed to implement Newton's method while minimizing the number of states required for computation. Each iteration is executed within a single clock cycle to enhance efficiency.

1.1.1 Description

The `square_root` entity computes the integer square root of a $2n$-bit input A based on the specified bit-width n. It operates synchronously with *clk* and includes control signals: *reset* for initialization and *start* to trigger the computation. The result is output as an n-bit vector *result*, with the *finished* signal indicating when the computation is complete.

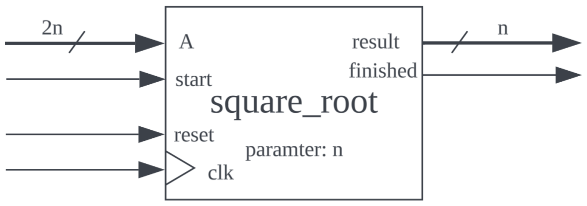


Figure 1: Top view level of the circuit

The proposed state machine, illustrated in Figure 2, consists of four states: **IDLE**, **INIT**, **COMP**, and **DONE**. When the *start* signal is set to '0', the state remains in **IDLE**, and the *finished* signal is reset to '0'. Once *start* is activated to '1', the state transitions to **INIT**, where the variable x (a register storing the updated result) is initialized. In the next clock cycle, the state moves to **COMP**, where the value of x is computed using Newton's method. Each iteration is completed within a single clock cycle. Once the final result is obtained, the state transitions to **DONE**, the *finished* signal is set from '0' to '1', and the output *result* is updated with the value of 'x'. If *start* is reset to '0', the state returns to **IDLE**, waiting for the next computation cycle.

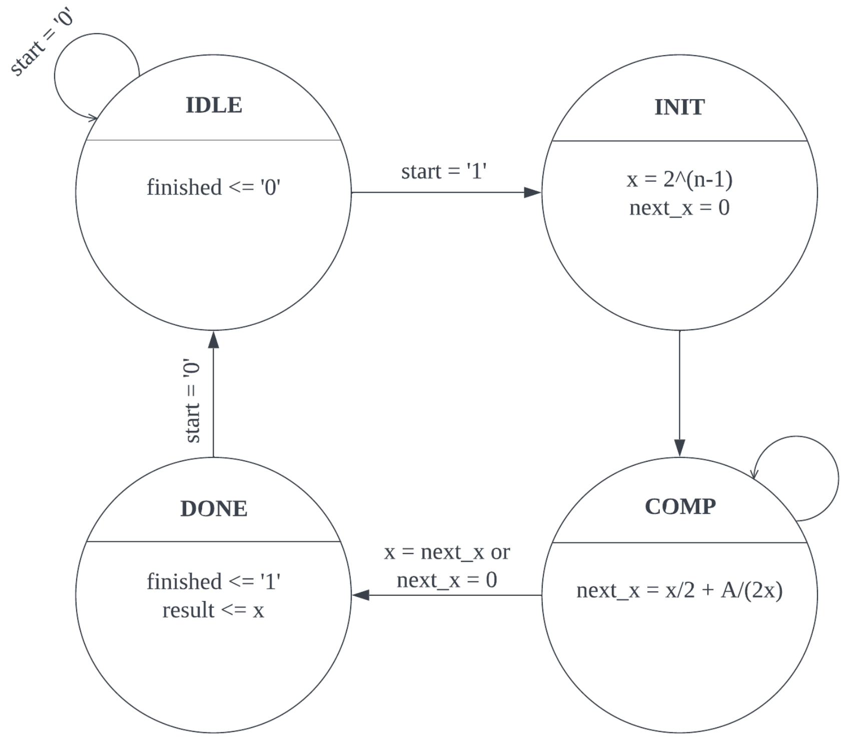


Figure 2: State machine for architectture 1

1.1.2 Simulation result

The testbench is designed to validate the `square_root` module by applying both of fixed and random test inputs. Initially, five specific inputs (0, 1, 512, 5499030, 1194877489) are tested sequentially. the *start* signal is set to initiate computation, while the system waits for the *finished* signal to confirm the result is ready before moving to the next input. Following these predefined tests, the testbench generates random inputs using a pseudo-random number generator, ensuring a wide variety of scenarios are evaluated until the total number of test cases reaches *NUM_TEST*.

The results indicate that the module correctly computes the square root for all test inputs. The number of clock cycles required to complete the computation depends on the the input.

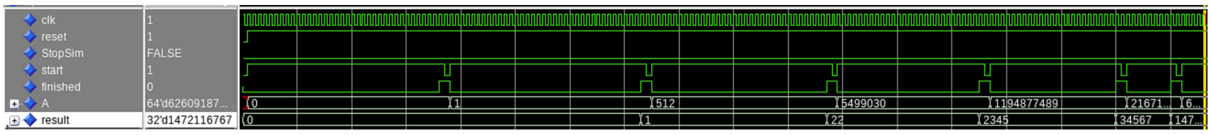
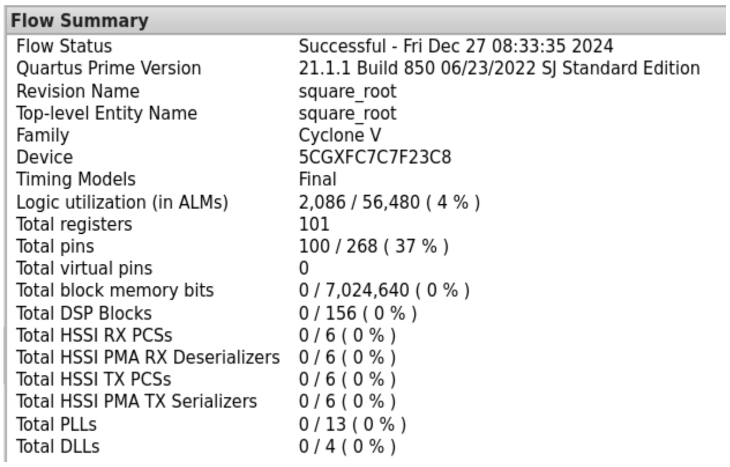


Figure 3: Simulation result for architecture 1

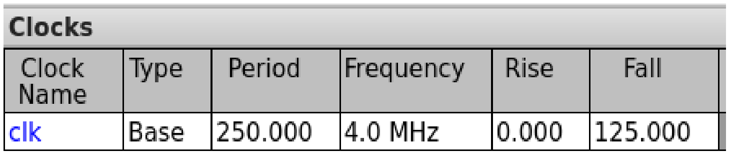
1.1.3 Syntheis result

The area and timing information for the architecture were obtained by performing synthesis in Quartus, targeting the Cyclone V 5CGXFC7C7F23C8 FPGA. The detailed settings are shown in the Figure 4.



Flow Summary	
Flow Status	Successful - Fri Dec 27 08:33:35 2024
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Standard Edition
Revision Name	square_root
Top-level Entity Name	square_root
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	2,086 / 56,480 (4 %)
Total registers	101
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

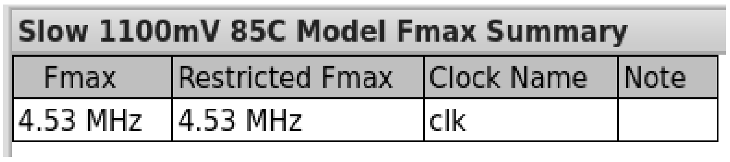
(a) Flow Summary of the `square_root` Design



Clocks

Clock Name	Type	Period	Frequency	Rise	Fall
clk	Base	250.000	4.0 MHz	0.000	125.000

(b) Clock Configuration for the Design



Slow 1100mV 85C Model Fmax Summary

Fmax	Restricted Fmax	Clock Name	Note
4.53 MHz	4.53 MHz	clk	

(c) Maximum Achievable Frequency (Fmax) Summary

Figure 4: Simulation result for architecture 1

The synthesis results indicate that the `square_root` design is highly resource-efficient, utilizing only 4% of ALMs (2,086/56,480) and 37% of pins (100/268), with minimal registers (101). It does not use block memory, DSP blocks, or advanced clocking resources. Timing analysis using the Slow 1100mV 85C Model is 4.53MHz for F_{max} the clock signal and 4.00 MHz for clock base.

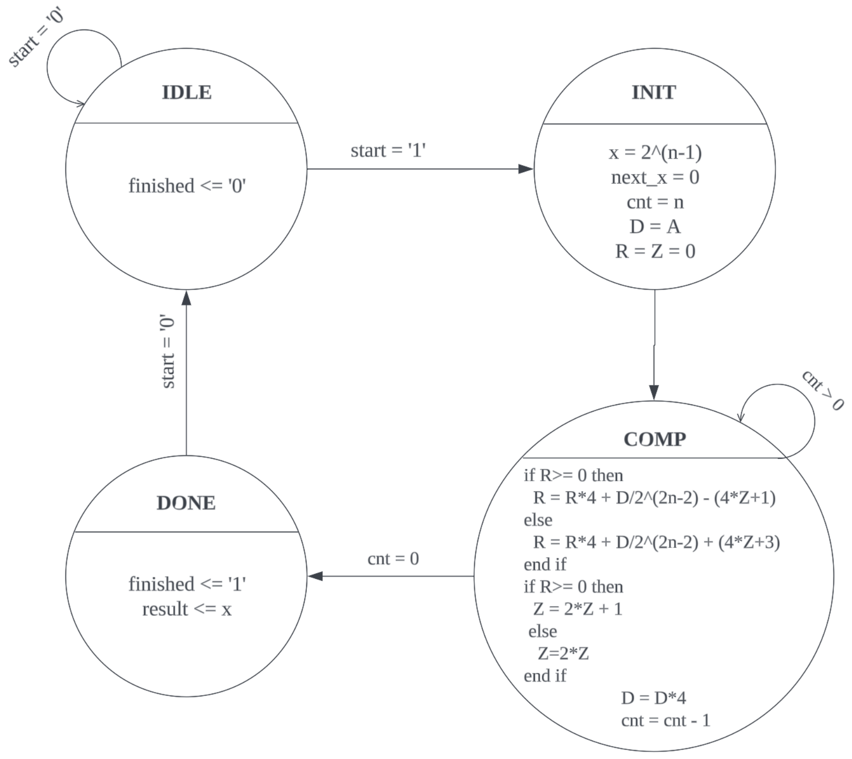


Figure 5: State machine for architecture 2

1.2 Sequential Design with Modified Non-Restoring Algorithm

1.2.1 Description

The ports of the entity for architecture a2 are the same as in architecture 1. For the state machine, the states and their functions remain unchanged. Additional signals introduced include D, Z, and the variables R and *cnt*. Here, Register Z holds the square root result, R represents the partial remainder, and D is a register that stores the input value A, which is shifted two bits to the left in each iteration.

1.2.2 Simulation result

The same testbench used for *architecture 1* is applied to *architecture 2*. For all input numbers, the square root results are successfully computed after n cycles. The computed square roots for the given input values in the problem statement, as well as for randomly generated inputs, are all correct.

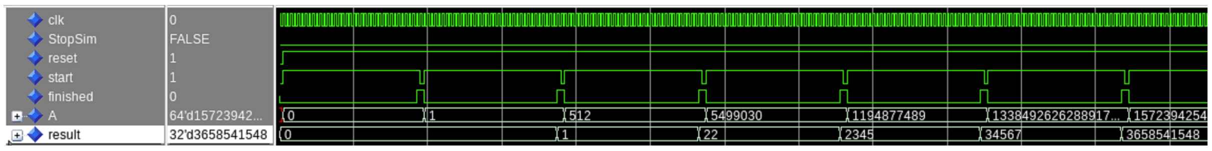


Figure 6: Simulation result for architecture 2

1.2.3 Synthesis result

The synthesis results in Figure 7 for `architecture 2` demonstrate the resource efficiency and performance of architecture 2. The design utilizes 160 Adaptive Logic Modules (ALMs), which accounts for less than 1% of the available 56,480 ALMs. A total of 248 registers are used, and 100 out of 268 pins (37%) are occupied. No block memory, DSP blocks, or advanced clocking resources such as PLLs or DLLs are required, highlighting the lightweight nature of the implementation.

Timing analysis using the *Slow 1100mV 0C Model* indicates a maximum clock frequency (*Fmax*) of 117.47 MHz and clock base of 111.11 MHz.

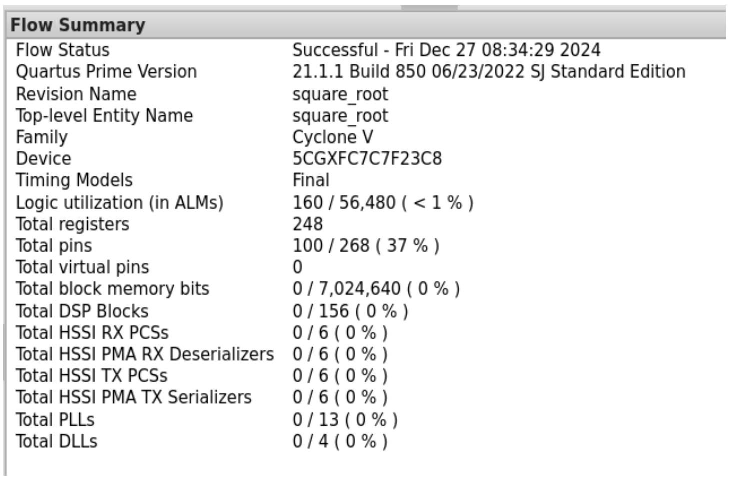
1.3 Combinational Design with Modified Non-Restoring Algorithm

1.3.1 Description

This architecture a3 executes all iterations of the for loop within a single clock cycle, independent of the *start*, *reset*, *clk*, and *finish* signals during functional operation. However, for synthesis purposes, the *clk, reset*, and *finish* signals are still taken into account. Additionally, two registers, *result_D* and *Q_A*, are added. Here, *result_D* stores the output variable, while *Q_A* holds the input variable, which is shifted two bits to the left in each iteration.

1.3.2 Simulation result

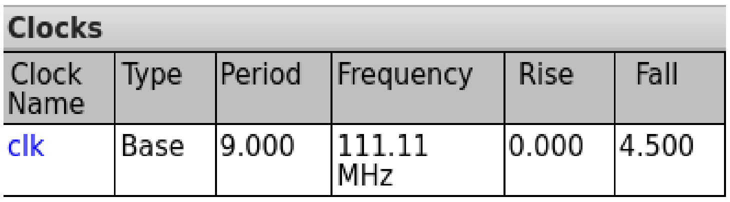
The simulation results show that the input signal *A* is correctly captured and stored in the input register *Q_A*. The intermediate result, stored in *result_D*, and the final output, *result*, match the expected square root values for all test cases. The computation successfully completes within the expected number of clock cycles for each input.



Flow Summary

Flow Status	Successful - Fri Dec 27 08:34:29 2024
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Standard Edition
Revision Name	square_root
Top-level Entity Name	square_root
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	160 / 56,480 (< 1 %)
Total registers	248
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

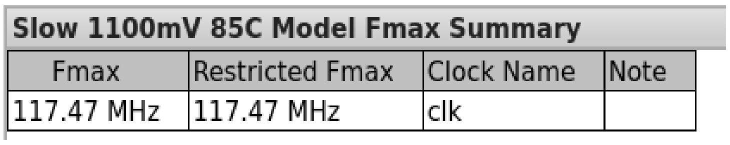
(a) Flow Summary of the `square_root` Design



Clocks

Clock Name	Type	Period	Frequency	Rise	Fall
clk	Base	9.000	111.11 MHz	0.000	4.500

(b) Clock Configuration for the Design

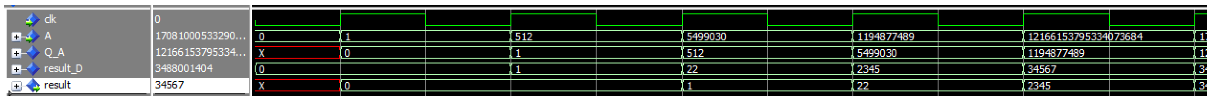


Slow 1100mV 85C Model Fmax Summary

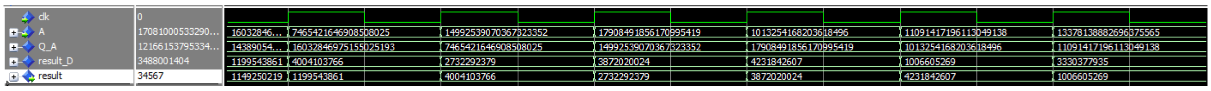
Fmax	Restricted Fmax	Clock Name	Note
117.47 MHz	117.47 MHz	clk	

(c) Maximum Achievable Frequency (Fmax) Summary

Figure 7: Simulation result for architecture 2



(a) Simulation result with fixed given inputs



(b) Simulation result with random inputs

Figure 8: Simulation result for architecture 3

1.3.3 Synthesis result

The diagram illustrates the top-level data flow of the square root computation module. The input signal *A[63:0]* is first stored in the input register *A_Q[63:0]* synchronized with the clock signal (*clk*). This value is then passed to the `square_root` module (labeled as *UUT*), which computes the square root of the 64-bit input and outputs the 32-bit result *result[31:0]*. The output is subsequently stored in the output register *result[31:0]* for further use, ensuring proper timing and synchronization across the design.

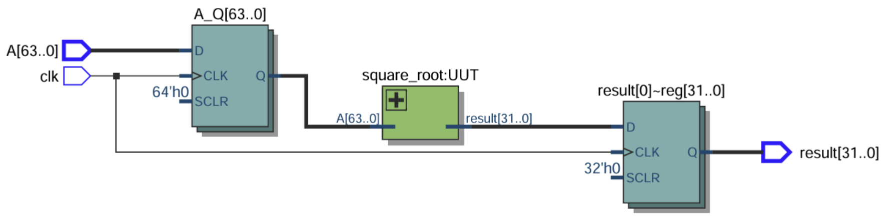


Figure 9: Data Flow of the Square Root Computation Module

The design uses 1,352 Adaptive Logic Modules (ALMs), accounting for 2% of the total available ALMs on the Cyclone V FPGA. A total of 111 registers are utilized, with 97 pins being used (36% of the available pins). No block memory, DSP blocks, or advanced high-speed serial interface (HSSI) resources are required.

Timing analysis using the *Slow 1100mV 85C Model* reveals a maximum clock frequency (*Fmax*) of 10.07 MHz and the clock base of 9.52 MHz.

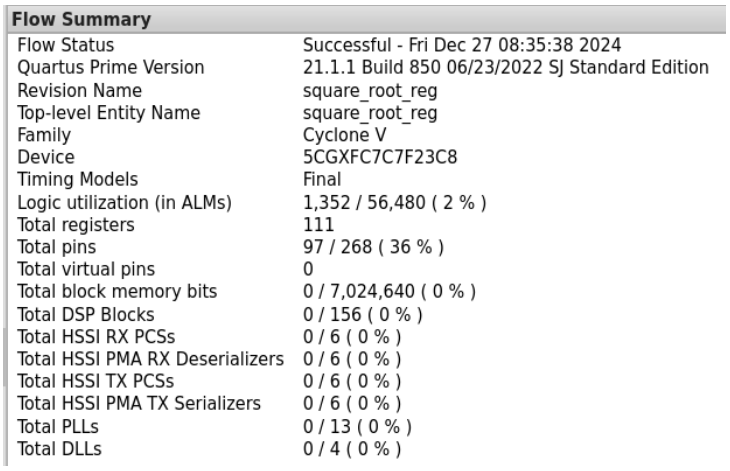
1.4 Pipelined Design with Modified Non-Restoring Algorithm

1.4.1 Description

This architecture a4 implements a pipeline square root calculation using an iterative algorithm. This fully pipelined design allows initiating a new square root computation on every clock cycle. Signals *stage_R*, *stage_Z*, and *stage_D* hold intermediate results for partial remainder, the result register, and shifted input register respectively. The two first signals are unsigned array to store all the results of the iterations. After n stages, the final value of *stage_Z(n)* represents the computed square root. The finished signal is asserted when the calculation is complete *(shift_reg_start(n) = '1')*.

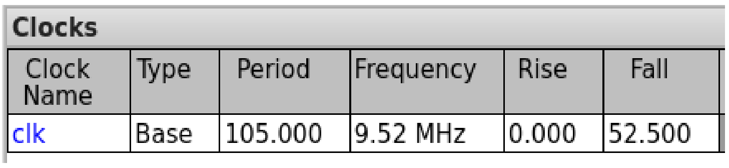
1.4.2 Simulation result

The testbench used to validate the pipeline architecture includes both direct tests and random tests, similar to the sequential architecture. However, in the pipeline testbench, the value of A is modified at each clock cycle. Figure 11 shows the simulation waveform.



Flow Summary	
Flow Status	Successful - Fri Dec 27 08:35:38 2024
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Standard Edition
Revision Name	square_root_reg
Top-level Entity Name	square_root_reg
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	1,352 / 56,480 (2 %)
Total registers	111
Total pins	97 / 268 (36 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

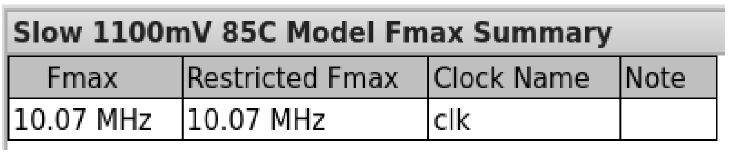
(a) Flow Summary of the `square_root` Design



Clocks

Clock Name	Type	Period	Frequency	Rise	Fall
clk	Base	105.000	9.52 MHz	0.000	52.500

(b) Clock Configuration for the Design



Slow 1100mV 85C Model Fmax Summary

Fmax	Restricted Fmax	Clock Name	Note
10.07 MHz	10.07 MHz	clk	

(c) Maximum Achievable Frequency (Fmax) Summary

Figure 10: Simulation result for architecture 3

The results indicate that the module correctly computes the square root for all test inputs. The pipeline architecture are also highlighted: after 32 clock cycles from the first input, the first output is computed, and subsequently, a new result is produced every cycle (Latency = 34, Throughput = 1).

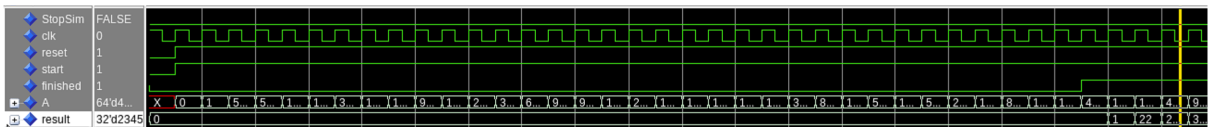


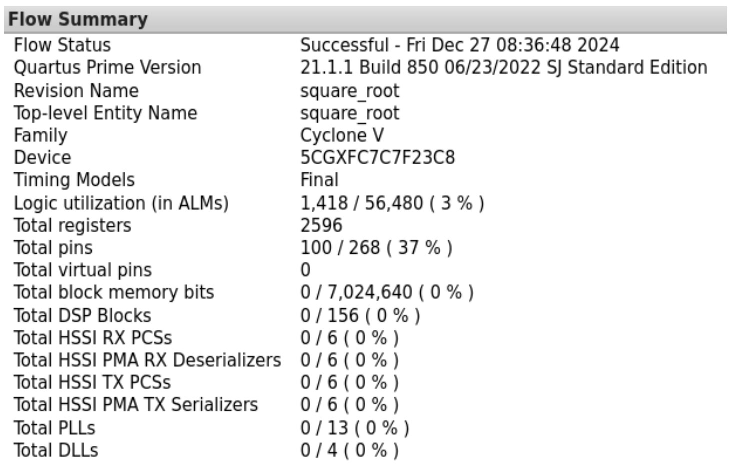
Figure 11: Simulation result for architecture 4

1.4.3 Synthesis result

The synthesis results for the `square_root` design show successful implementation on the Cyclone V FPGA. The design uses 1,418 Adaptive Logic Modules (ALMs), which is 3%

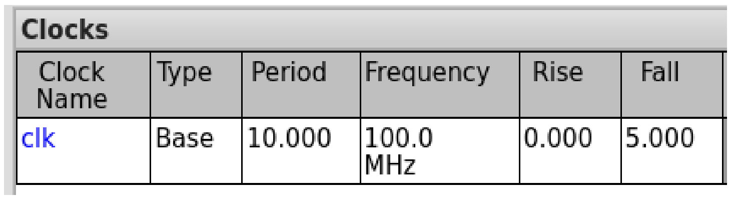
of the total available ALMs, along with 2,596 registers and 100 pins (37% of the available pins). No block memory bits, DSP blocks, or advanced high-speed serial interface (HSSI) resources are used.

Timing analysis with the Slow 1100mV 85C Model shows a maximum clock frequency (Fmax) of 116.54 MHz, while the base clock operates at 100 MHz.



Flow Summary	
Flow Status	Successful - Fri Dec 27 08:36:48 2024
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Standard Edition
Revision Name	square_root
Top-level Entity Name	square_root
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	1,418 / 56,480 (3 %)
Total registers	2596
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

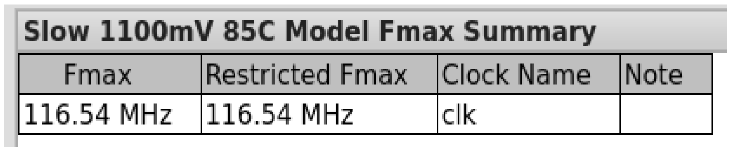
(a) Flow Summary of the `square_root` Design



Clocks

Clock Name	Type	Period	Frequency	Rise	Fall
clk	Base	10.000	100.0 MHz	0.000	5.000

(b) Clock Configuration for the Design



Slow 1100mV 85C Model Fmax Summary

Fmax	Restricted Fmax	Clock Name	Note
116.54 MHz	116.54 MHz	clk	

(c) Maximum Achievable Frequency (Fmax) Summary

Figure 12: Simulation result for architecture 4

2 Structural Approach

2.1 Description

This architecture a5 implements a square root calculator using a structural approach, separating the control path and datapath. The control path, implemented as a finite state machine (FSM), manages the computation process with four states: **IDLE, INIT, COMP**, and **DONE**. It controls signals like *cnt* and *finish_flag* to synchronize operations. The datapath consists of registers (D, Z, and R) and arithmetic logic to iteratively compute the square root. The FSM directs the datapath, which updates the partial remainder (R), result (Z), and shifted input (D) over multiple cycles. The final result is output when the FSM reaches the DONE state and asserts the finished signal. The datapath is illustrated in the Figure 13.

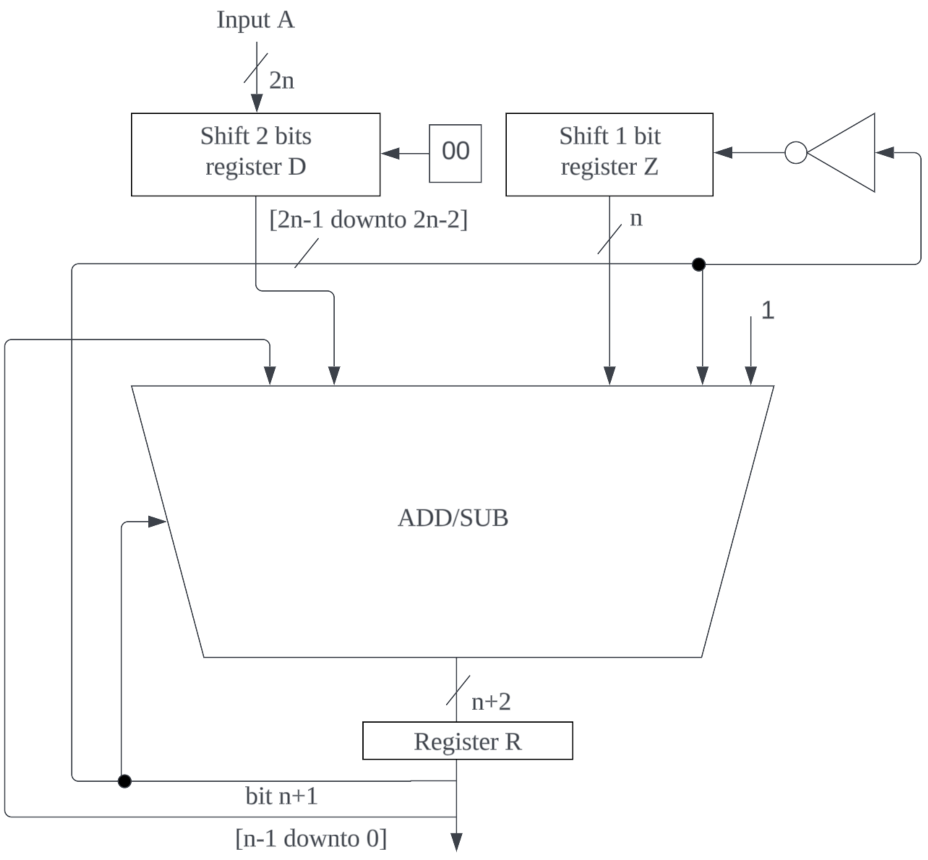


Figure 13: Datapath flow for architecture 5

2.2 Simulation result

The same testbench used for *architecture 1* is applied to *architecture 2*. For all input numbers, the square root results are successfully computed after n cycles. The computed square roots for the given input values in the problem statement, as well as for randomly generated inputs, are all correct.

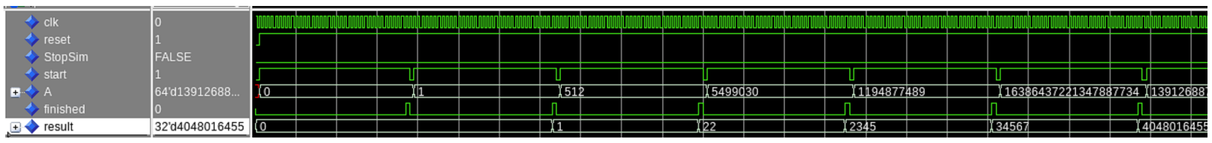
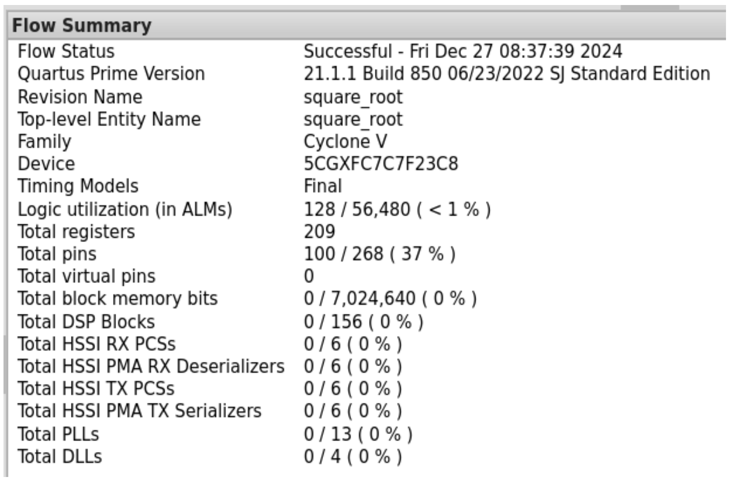


Figure 14: Simulation result for architecture a5

2.3 Synthesis result

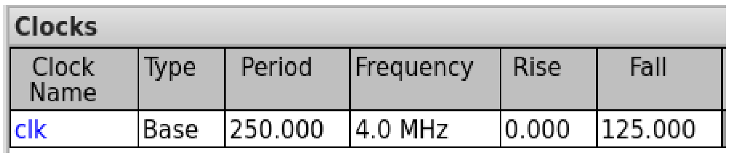
The square_root design uses 128 Adaptive Logic Modules (ALMs), accounting for less than 1% of the total available ALMs on the Cyclone V FPGA. A total of 209 registers are utilized, with 100 pins being used (37% of the available pins). No block memory, DSP blocks, or advanced high-speed serial interface (HSSI) resources are required.

Timing analysis using the *Slow 1100mV 85C Model* reveals a maximum clock frequency (*Fmax*) of 77.68 MHz, while the clock base operates at 4.0 MHz.



Flow Summary	
Flow Status	Successful - Fri Dec 27 08:37:39 2024
Quartus Prime Version	21.1.1 Build 850 06/23/2022 SJ Standard Edition
Revision Name	square_root
Top-level Entity Name	square_root
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	128 / 56,480 (< 1 %)
Total registers	209
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

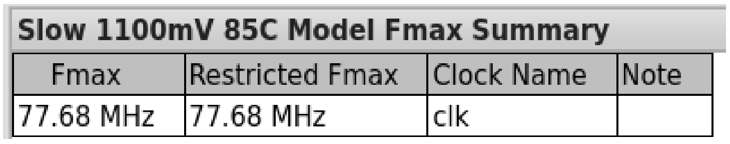
(a) Flow Summary of the `square_root` Design



Clocks

Clock Name	Type	Period	Frequency	Rise	Fall
clk	Base	250.000	4.0 MHz	0.000	125.000

(b) Clock Configuration for the Design



Slow 1100mV 85C Model Fmax Summary

Fmax	Restricted Fmax	Clock Name	Note
77.68 MHz	77.68 MHz	clk	

(c) Maximum Achievable Frequency (Fmax) Summary

Figure 15: Simulation result for architecture 5

3 Performance Analysis and Design Comparison

Table 1 summarizes the synthesis results of all architectures, including the number of clock cycles required to compute the result in the worst case (#cycles).

The iterative algorithm outperforms the Newton method across all metrics. While the number of clock cycles required to compute the result using the Newton method is dynamic (depending on the input and the initial value of the result), the number of clock cycles for the iterative algorithm is deterministic. The worst-case scenario for the Newton architecture is 35 clock cycles.

The following section compares architectures based on iterative algorithms.

Overall, the sequential architecture requires fewer ALMs than other architectures. As predicted, the combinational architecture (*a3*) consumes the highest number of ALMs. It can be observed that the structural architecture is more hardware resource-efficient because it provides a more detailed description at the hardware layer rather than focusing solely on the behavior of operations. In terms of registers, the combinational architecture requires only the registers used for its inputs and outputs. In contrast, the pipeline architecture (*a4*) demands a significantly higher number of registers due to the need for registers between each pipeline stage.

Regarding computation duration, the sequential architecture achieves the highest maximum frequency but requires more clock cycles to compute the result. On the other hand, the pipeline architecture achieves high throughput by producing a new output every clock cycle. To compare different architectures, we calculate the T_{comp}(us) value based on the highest achievable frequency and the number of clock cycles required to compute the results. It shows that architecture 4 (*a4)* is the best in terms of computation times but it requires a latency between the first input and the first output.

Arch	**Fclk (MHz)**	**Fclk$_{max}$ (MHz)**	**ALMs**	**Regs**	**#cycles**	**T$_{comp}$**
a1	4	4.53	2086	101	35	7.72
a2	111.11	111.47	160	248	34	0.305
a3	9.52	10.07	1352	111	1	0.099
a4	100	116.54	1418	2596	1	0.009
a5	4	77.68	128	209	35	0.451

Table 1: Performance comparison of different architectures.

We define a simple Figure of Merit to fairly evaluate their resource utilization and computational efficiency:

$$FoM = T_{comp}(us)(\#\text{ALMs} + \#\text{Regs})$$

A small FoM corresponds to an architecture with a good speed-size tradeoff. Considering this, architecture 4 demonstrates the best resource-latency tradeoff.