1 Simulation results

1.1 Clock cycle count

Answer:

The computation requires $N - 1$ clock cycles for an input N (for $N \geq 2$). For example, calculating Fibonacci(5) takes 4 compute cycles plus setup/done states. The simulation waveform confirms this behavior, showing the 'done' signal asserting after the 'state' transitions through the COMPUTE loop. The waveform is shown in Figure 1

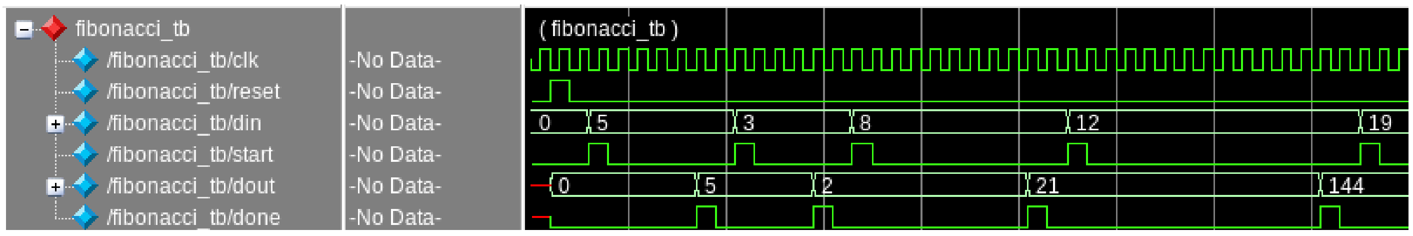
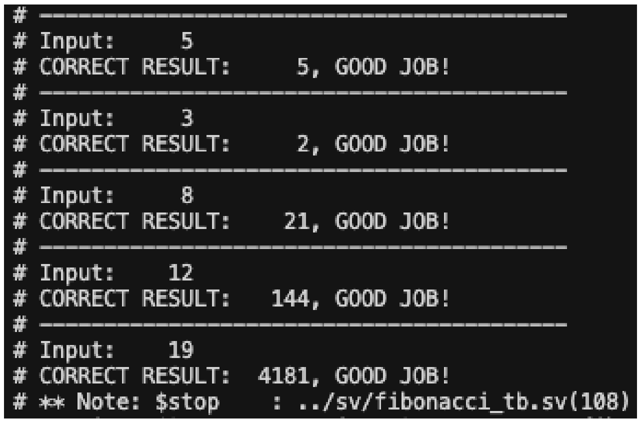


Figure 1: ModelSim waveform showing correct Fibonacci calculation for inputs 5, 3, 8, 12, and 19. The "dout" signal updates to the correct values (5, 2, 21, 144) as the "done" signal asserts.

1.2 Errors reported

Answer:

No errors were reported. The testbench successfully verified all 5 test cases against expected values as shown in Figure 2



```
# ----------------------------------------
# Input:     5
# CORRECT RESULT:     5, GOOD JOB!
# ----------------------------------------
# Input:     3
# CORRECT RESULT:     2, GOOD JOB!
# ----------------------------------------
# Input:     8
# CORRECT RESULT:    21, GOOD JOB!
# ----------------------------------------
# Input:    12
# CORRECT RESULT:   144, GOOD JOB!
# ----------------------------------------
# Input:    19
# CORRECT RESULT:  4181, GOOD JOB!
# ** Note: $stop    : ../sv/fibonacci_tb.sv(108)
```

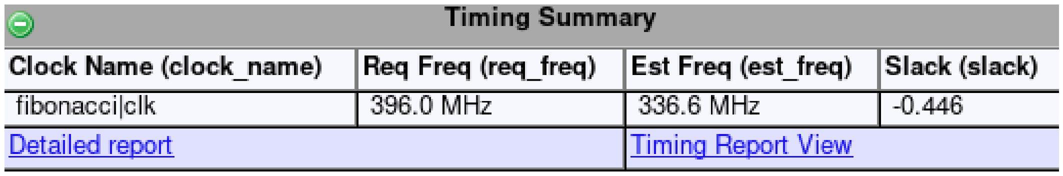
Figure 2: Simulation transcript confirming that all test cases (Input=5, 3, 8, 12, 19) passed with "CORRECT RESULT".

2 Synthesis results

2.1 Maximum frequency

Answer:

Based on the synthesis timing report, the maximum estimated operating frequency of the design is **336.6 MHz**. This estimation is derived from the critical path delay analysis after technology mapping. As shown in Figure 3, the synthesis tool targeted a requested frequency of 396.0 MHz, achieving an estimated frequency of 336.6 MHz with a slack of -0.446 ns.



Timing Summary

Clock Name (clock_name)	Req Freq (req_freq)	Est Freq (est_freq)	Slack (slack)
fibonacci|clk	396.0 MHz	336.6 MHz	-0.446
Detailed report		Timing Report View	

Figure 3: Timing summary report showing the estimated frequency of 336.6 MHz and the critical path slack.

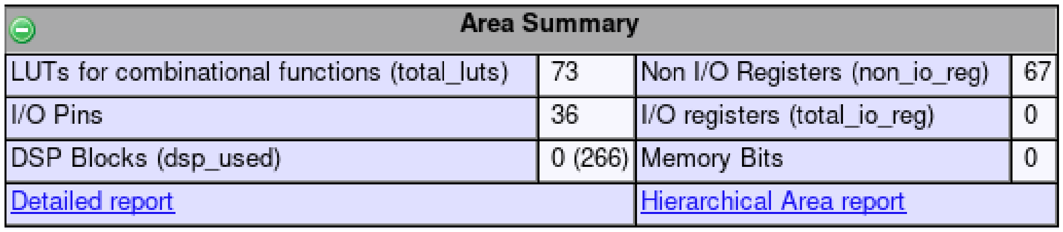
2.2 Registers/LUTs/Logic Elements

Answer:

The resource utilization for the Fibonacci generator on the Cyclone IV-E FPGA is summarized below, based on the synthesis area report shown in Figure 4:

1. Total Registers: 67
2. Total Combinational Functions (LUTs): 73
3. Logic Elements: 73 (Based on total combinational functions utilization)

These resources are used to implement the 16-bit datapath (counters, accumulators) and the finite state machine control logic.



Area Summary

LUTs for combinational functions (total_luts)	73	Non I/O Registers (non_io_reg)	67
I/O Pins	36	I/O registers (total_io_reg)	0
DSP Blocks (dsp_used)	0 (266)	Memory Bits	0
Detailed report		Hierarchical Area report	

Figure 4: Area summary report detailing the resource usage, including 73 LUTs and 67 registers.

2.3 Memory utilization

Answer:

The design utilizes **0 bits** of memory , as indicated in the "Memory Bits" field of Figure 4. The storage requirements for the Fibonacci sequence calculation are minimal and are handled by the Flip-Flops rather than dedicated memory blocks.

2.4 Multipliers(DSPs)

Answer:

The DSP block utilization is **0 (0.00%)**, as shown in Figure 4. The Fibonacci algorithm relies solely on addition operations, which are implemented using the carry chains and LUTs within the Logic Elements, requiring no dedicated hardware multipliers or DSP resources.

2.5 Worst path(timing analysis)

Answer:

1. **Critical Path Slack:** -0.446 ns (at 396 MHz target)
2. **Path Delay:** 2.971 ns
3. **Start Point:** `state[1]` (FSM Register)
4. **End Point:** `count[0]` (Counter Register enable signal)
5. **Logic Levels:** 2

The critical path analysis reveals the timing bottleneck in the design. Figure 5 illustrates the hierarchical schematic of this critical path. The path originates from the FSM state register (labeled `S_2`), propagates through combinational logic blocks (`C_3_24` and `C_4_11`) which implement the state decoding and control logic, and terminates at the enable pin of the counter registers (`count[15:0]`, labeled `S_0`). This indicates that the maximum operating frequency is limited by the time required for the FSM to decode the current state and assert the enable signal for the counter.

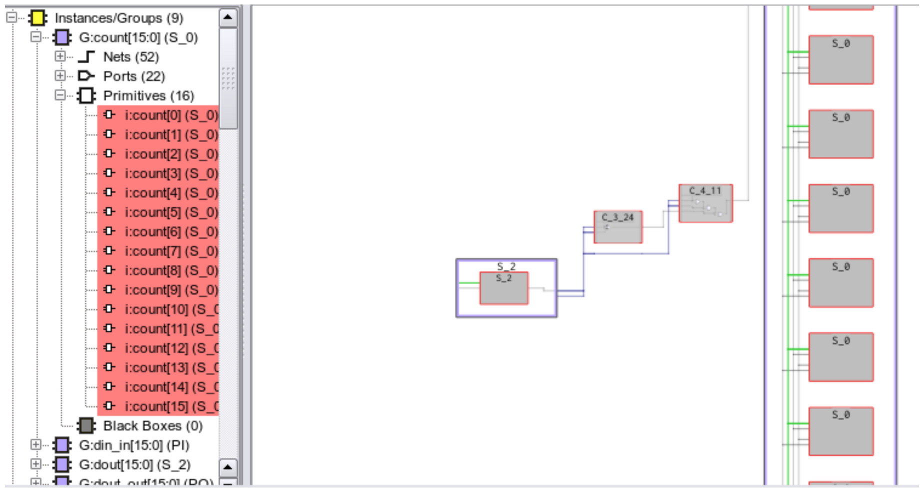


Figure 5: Hierarchical view of the critical path. The highlighted path shows the signal propagation from the FSM state register (`S_2`), through the control logic (`C_3_24`, `C_4_11`), to the enable input of the counter registers (`count[15:0]`).

2.6 Schematic architecture(RTL)

Answer:

The RTL schematic mainly consists of:

a. A 3-state Finite State Machine (IDLE, COMPUTE, DONE) encoded with 3 bits (One-Hot).

b. Three 16-bit Registers: count, f1, f2.

c. One 16-bit Adder: For computing f1 + f2.

d. One 16-bit Decrementer: For count $\leq$ count - 1.

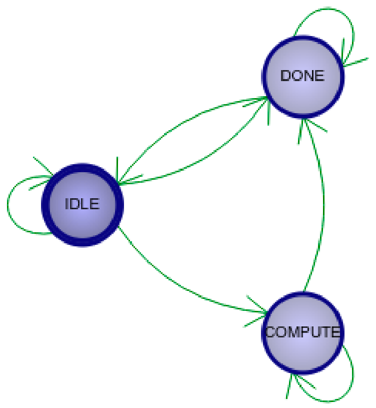
e. A multiplexers for selecting initial values (0, 1, or input din) vs. calculated values.

f. The control logic which generates enable signals for registers based on the current state and start signal.

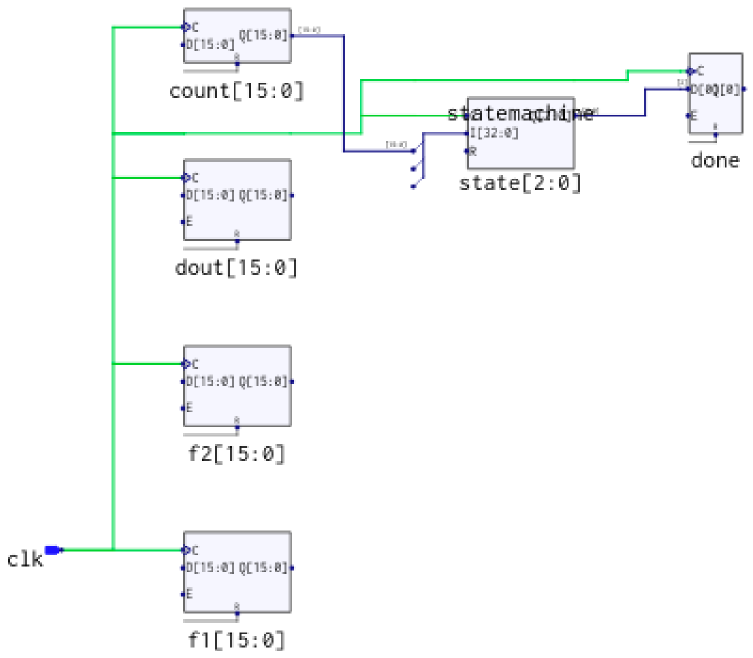
The synthesized design architecture is illustrated through the RTL views in Figure 6. The design is logically partitioned into a control path and a data path. The control path, shown in Figure 6a, consists of a 3-state Finite State Machine (IDLE, COMPUTE, DONE) implemented with a 3-bit one-hot encoding. This FSM orchestrates the overall operation sequence.

The hierarchical RTL view (Figure 6b) explicitly shows the major components: the `statemachine` block alongside the 16-bit data registers `count`, `f1`, `f2`, and `dout`.

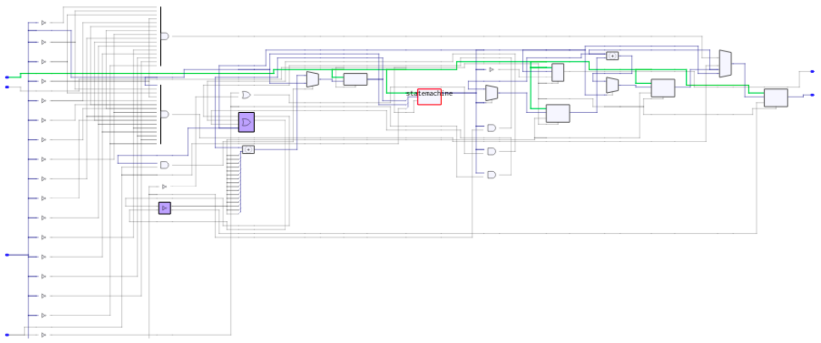
The flattened schematic (Figure 6c) reveals the underlying implementation details, including the adder for $f1 + f2$, the decrementer for the counter, and the multiplexers used for selecting between initialization values (0, 1, or input `din`) and calculated results. The control logic interconnects these components, generating the necessary enable signals for the registers based on the current state.



(a) State Transition Diagram



(b) Hierarchical RTL View



(c) Flattened RTL Schematic

Figure 6: RTL Architecture of the Fibonacci Design. (a) The FSM state diagram showing IDLE, COMPUTE, and DONE states. (b) Hierarchical view highlighting the separation of the FSM control block and the datapath registers. (c) Flattened view showing the detailed interconnections, logic gates, and arithmetic operators.