# Advanced VLSI

## Bitcoin

## Stage 1

Ahmad foqara  322389982

Mohammad foqara  212193916

Neama hebi 212100705

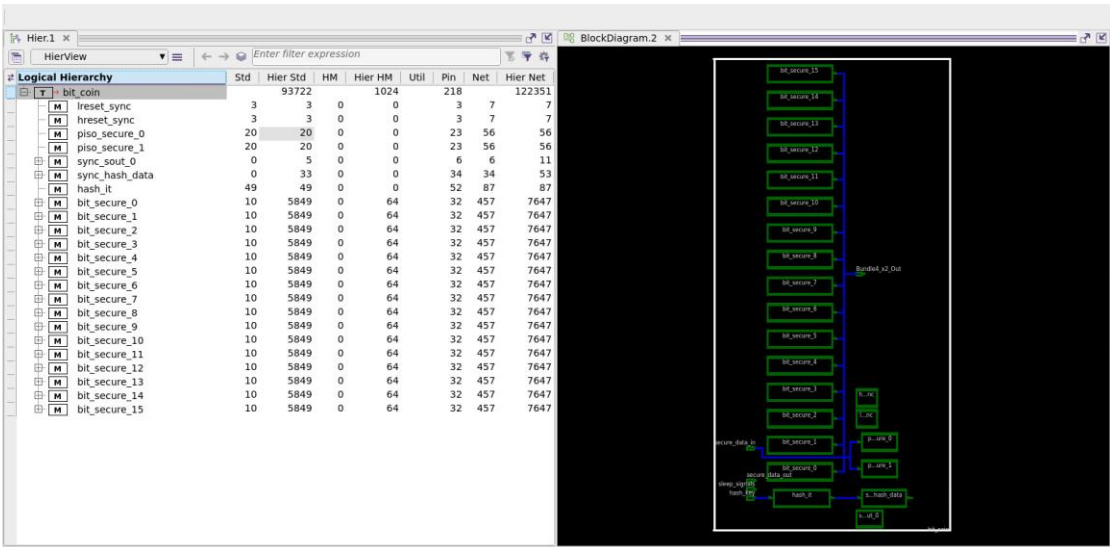
Ward aboyonis 212021869

Work path: /project/advvlsi/users/ foqara /ws/bitcoin

**[#P3.1_Q1] Locate and attach a few print screens of the major items from the architecture image (page 2) including:**

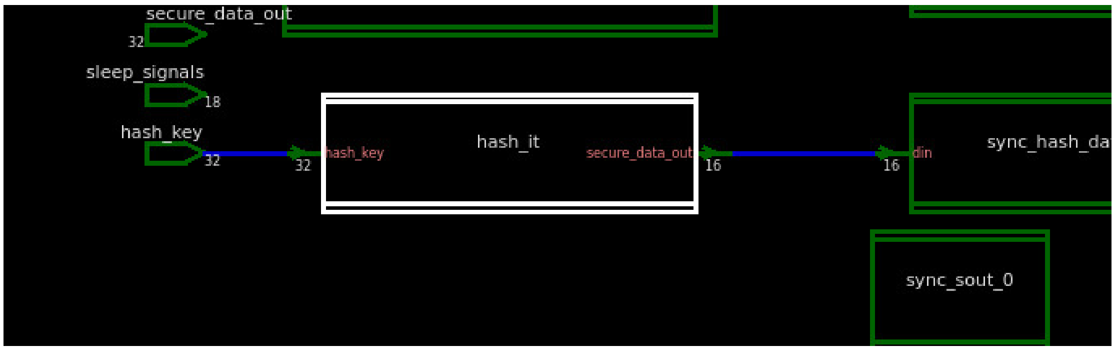
**Bit_coin:**

**16 bit_top (named also bit_secure):**

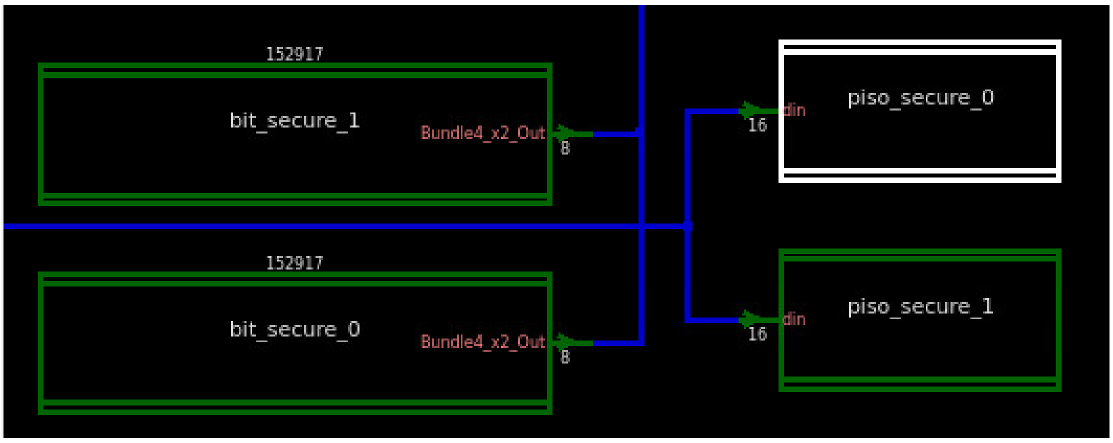


| Logical Hierarchy | Std | Hier Std | HM | Hier HM | Util | Pin | Net | Hier Net |
|---|---|---|---|---|---|---|---|---|
| bit_coin | | 93722 | | 1024 | | 218 | | 122351 |
| lreset_sync | 3 | 3 | 0 | 0 | | 3 | 7 | 7 |
| hreset_sync | 3 | 3 | 0 | 0 | | 3 | 7 | 7 |
| piso_secure_0 | 20 | 20 | 0 | 0 | | 23 | 56 | 56 |
| piso_secure_1 | 20 | 20 | 0 | 0 | | 23 | 56 | 56 |
| sync_sout_0 | 0 | 5 | 0 | 0 | | 6 | 6 | 11 |
| sync_hash_data | 0 | 33 | 0 | 0 | | 34 | 34 | 53 |
| hash_it | 49 | 49 | 0 | 0 | | 52 | 87 | 87 |
| bit_secure_0 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_1 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_2 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_3 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_4 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_5 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_6 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_7 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_8 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_9 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_10 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_11 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_12 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_13 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_14 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |
| bit_secure_15 | 10 | 5849 | 0 | 64 | | 32 | 457 | 7647 |

**Hash:**

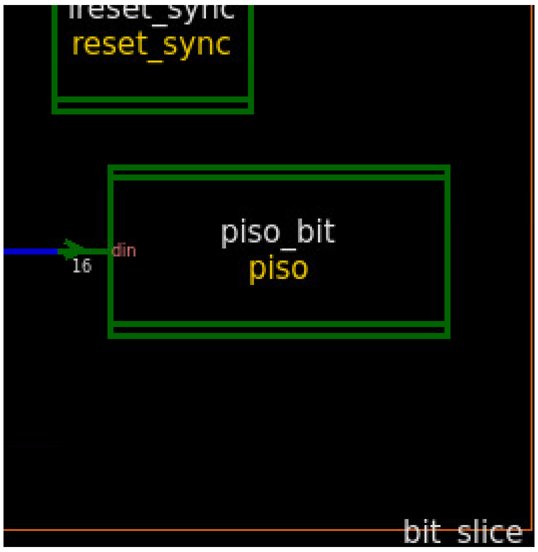


**Piso:**

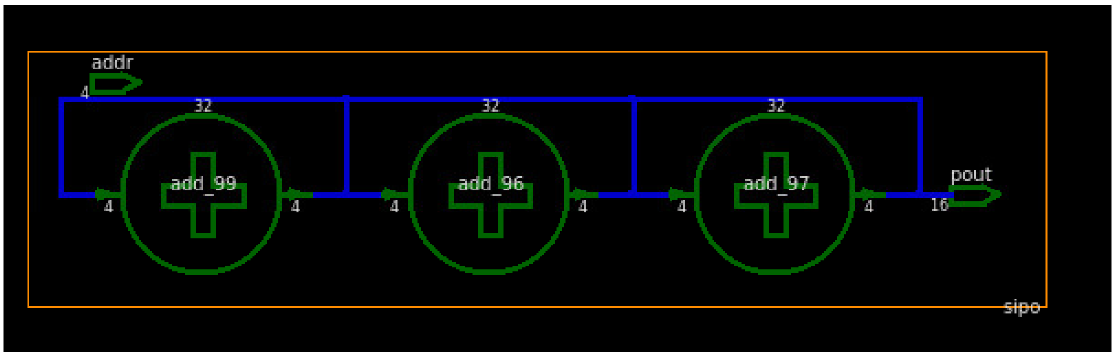


Bit_Slice :

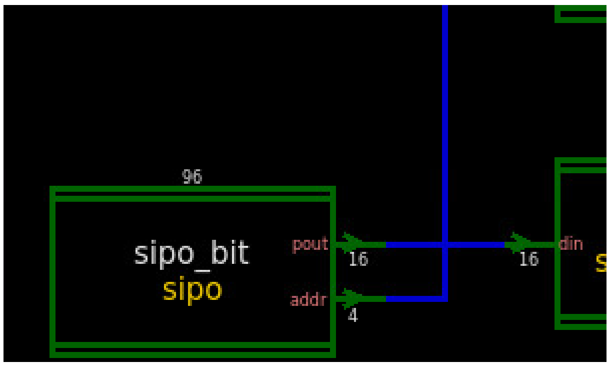
Piso:



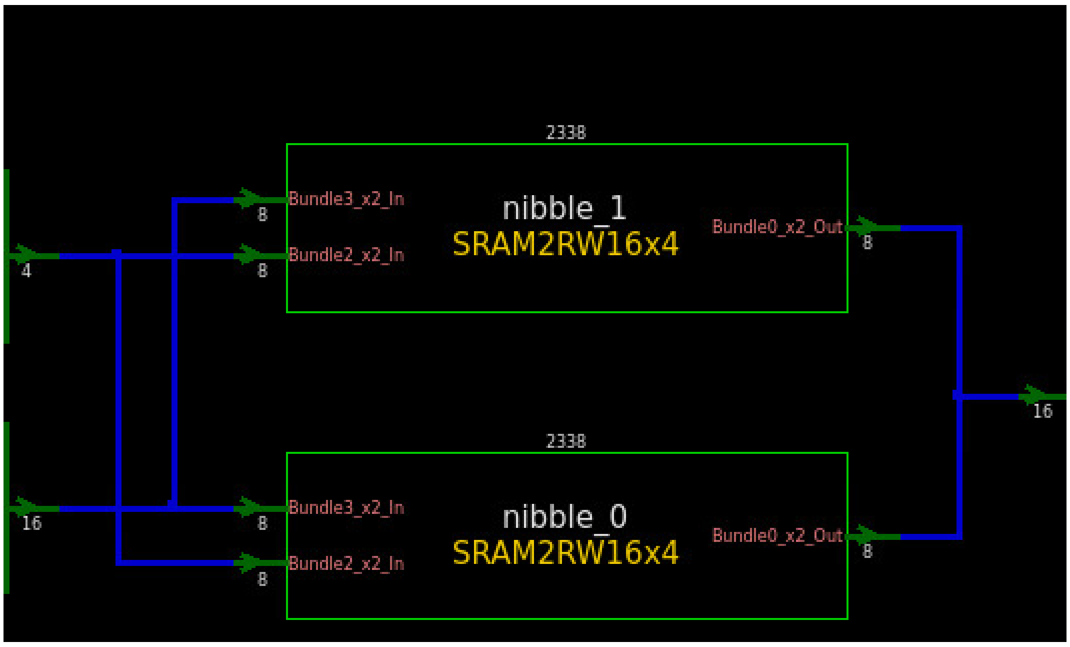
Sipo:



Sipo_bit:



2 Nibbles:



**[#P3.2_Q1] Constraints:**

**Write at least 5 different constraint examples, write who supplies them and their domain.**
**For example, we have the clocks' periods constraint from the timing domain. And the architecture team is responsible for providing them.**

1. **Clocks' Periods Constraint**

**Supplier: Architecture Team**

**Domain: Timing**

**Description:** A crucial timing parameter in VLSI design, the Clocks' Periods Constraint establishes the time interval between consecutive clock pulses and establishes the frequency at which a circuit functions. This restriction is essential for guaranteeing that all sequential design elements are in sync and that data is processed and delivered reliably within the allotted time. Setting clock periods correctly guarantees effective performance, helps prevent timing violations, and satisfies the design's overall timing requirements.

2. **Power Consumption Limits**

**Supplier: Power Management Team**

**Domain: Power**

**Description:** establishes the design's maximum permitted power consumption in order to guarantee thermal control and energy efficiency.

.

3. **Area Constraints**

**Supplier: Physical Design Team**

**Domain: Physical**

**Description:** shows the largest silicon area that the design is capable of taking up. In order to optimize the layout and ensure the chip fits inside the specified die size, certain limitations are necessary. Standard cells, macros, and other components are arranged according to area limits in order to optimize density and save wasted space. Because larger chips are more costly to create and may result in lower yield, it is imperative that these limits be followed for cost-effective manufacturing. Achieving design objectives including performance, power consumption, and reliability is also aided by effective area constraint management.

4. **Voltage Levels**

**Supplier: Electrical Engineering Team**

**Domain: Electrical**

**Description**: specifies the appropriate voltage levels for various circuit components for ensuring dependability and correct operation.

5. **Setup and Hold Time Constraints**

**Supplier: Timing Analysis Team**

**Domain: Timing**

**Description:** Sequential components like flip-flops and latches are ensured to record data accurately by setup and hold time limitations. The minimum amount of time before the clock edge that data must stay stable is known as setup time, and the minimum amount of time after the clock edge that data must remain stable is known as hold time.
These limitations are essential for avoiding timing errors, ensuring dependable data transfer, and preserving the circuit's operational integrity. Metastability, which causes unstable behavior and mistakes in the digital logic, can arise from breaking setup or hold times. The overall timing accuracy and endurance of the VLSI design depend on these constraints being properly followed**.**

**Name and explain 2 techniques that we (and maybe by extension the tool) can implement during the synthesis stage to meet low power constraints?**

**During the synthesis stage of integrated circuit (IC) design, several techniques can be employed to meet low power constraints. Here are three effective methods:**

1. **Clock Gating:**

Explanation A power-saving method called clock gating selectively cuts off clock signals to unused or inactive circuit components.
Lower power usage results from minimizing unnecessary switching activity in these areas by stopping the clock.
Application: Clock gating cells, which regulate clock signals according to specific conditions or enable signals, are used in this technology. These cells lower dynamic power usage by detecting when specific circuit components are not in use and turning off the clock signal.

2. **Voltage Scaling:**

Explanation: One technique for reducing power consumption is voltage scaling, which involves running the integrated circuit at lower supply voltages. Reduced voltage can result in significant power savings because dynamic power is related to the square of the supply voltage. This method is especially crucial for applications that require a lot of power, like mobile phones.

Application: The IC's various components can be made to function at different voltage levels that are best suited to their individual performance requirements. In order to control voltage variations across these domains, level shifters are frequently used to create voltage islands within the chip, each with its own power domain.

3. **Power Gating:**

Power gating is a technique that shuts off the power supply to an IC's idle or inactive blocks entirely.  In order to reduce leakage power, which happens even when the circuit is not in use, these blocks are isolated from the power source. This technique efficiently lowers power usage, both dynamic and static.

Application: To cut off power to particular circuit blocks, specialized power gating cells function as switches. Power management logic controls these switches, deciding when to cut off power to specific regions of the chip to save energy.

Designers can obtain significant power reductions in integrated circuits while preserving the necessary performance by putting these ideas into practice during the synthesis stage.

**What are the tradeoffs with the solutions that you offered?**

While clock gating, voltage scaling, and power gating are effective techniques for reducing power consumption, they each come with tradeoffs that must be carefully considered:

1. **Clock Gating Tradeoffs:**

Added Design Complexity: Clock gating necessitates the use of extra clock gating cell logic.
This may affect time closure and overall design productivity by making the design process more difficult.
Impact on the Clock Network: Adding clock gating cells may have an impact on the clock distribution network, increasing routing congestion or skew.
Power and size Overhead: Clock gating lowers dynamic power usage, but it also increases the total chip size and introduces some static power consumption due to the additional circuitry required for gating.

2. **Voltage Scaling Tradeoffs:**

Impact on Performance: While lowering the supply voltage lowers power consumption, performance may suffer as a result. Lower operating frequencies or more stringent design constraints to satisfy timing requirements may arise from decreased circuit speeds and greater propagation delays.
Sensitivity to Process Variations: Circuits may become more vulnerable to process changes as a result of voltage scaling, which could result in greater performance variances between manufactured chips.
Complexity of Design and Difficulties with Verification: The circuit's time, noise margins, and voltage levels are all impacted by voltage scaling. This adds complexity and verification overhead because it calls for meticulous design and validation.

3. **Power Gating Tradeoffs:**

Added Design Complexity: Power gating increases the complexity of the design process by requiring the insertion of control logic and power gating cells. It can be difficult to regulate control signals and guarantee correct operation during power changes.
Effect on Signal Integrity: Increased IR drop, crosstalk, and other delay variations are just a few of the signal integrity problems that power gating may cause by introducing more power domains and isolation techniques.
Design Restrictions and Trade-offs: Wake-up latency, power-up sequence, and control signal generation are just a few of the limitations that must be carefully taken into account when using power gating. It may be necessary to make trade-offs in order to balance these limitations with the intended power savings.
In conclusion, power gating, voltage scaling, and clock gating all reduce power consumption, but they come with trade-offs in terms of performance impact, design complexity, sensitivity to variations, and verification challenges.

## [#P3.2_Q2] False paths: What are False paths? When should we use them?

In digital design, false paths are those that, although they seem like potential paths for signal propagation, are never really traveled when the circuit is operating. This may be because of the circuit's particular design or how it is used. Essentially, under the design's specified operating parameters, these paths are illogical.

Parts of the logic having varying delays, where certain logic sections are faster or slower than others, may be included in false paths. For example, paths that might ordinarily be possible under particular simultaneous conditions become false paths if a design guarantees that specific control signals or situations never occur simultaneously.

### When Should We Use False Paths?

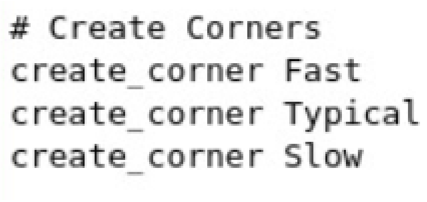
False paths should be specified and used during the design and verification process for several reasons:

1. **Synthesis Optimization**: Tools optimize the design during logic synthesis in order to satisfy time limitations. The synthesis tool may waste resources attempting to optimize erroneous pathways if they are not detected, which would make the design needlessly complex and inefficient.
2. **Timing Analysis** False path information is used by timing analysis tools to disregard these paths during setup and hold time checks. By doing this, incorrect timing violations are avoided, which could otherwise result in inaccurate assessments of the timing performance of the design.
3. **Resource Allocation**Design tools can more effectively direct resources and effort toward optimizing real, critical paths by detecting and eliminating false paths. This ensures that the timing-sensitive portions of the design are appropriately optimized.

4. **Simulation Efficiency**: By identifying which paths are false during simulation, the simulator can save time and computational resources by not having to check these paths, which speeds up and improves the efficiency of the verification process.
5. **Avoiding Design Bugs**Early detection of possible errors in design at the RTL simulation stage, as opposed to later detection during the gate-level simulation stage, can be achieved by accurately recognizing erroneous paths. This minimizes the time and effort required to troubleshoot and resolve problems.

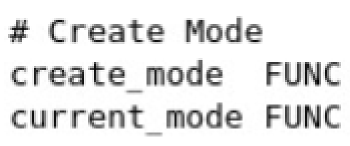
**[#P3.2_Q3] Corners, modes, and scenarios: - Look at bitcoin_stage_1.tcl and locate the part related to the corners, modes, and scenarios (MCMM). Which scenario \ scenarios are active?**

**Corners:**



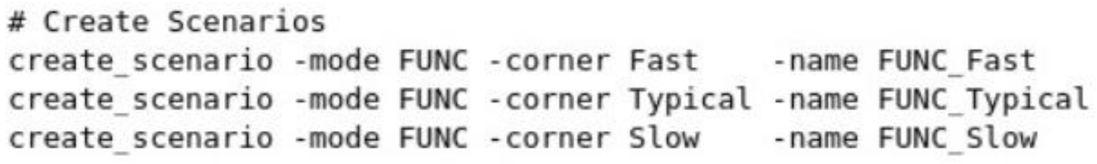
```
# Create Corners
create_corner Fast
create_corner Typical
create_corner Slow
```

**Modes:**



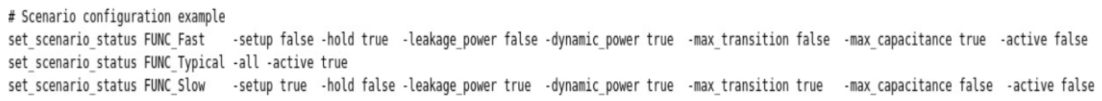
```
# Create Mode
create_mode  FUNC
current_mode FUNC
```

**Scenarios:**



```
# Create Scenarios
create_scenario -mode FUNC -corner Fast    -name FUNC_Fast
create_scenario -mode FUNC -corner Typical -name FUNC_Typical
create_scenario -mode FUNC -corner Slow    -name FUNC_Slow
```

**Which scenario \ scenarios are active?**



```
# Scenario configuration example
set_scenario_status FUNC_Fast    -setup false -hold true  -leakage_power false -dynamic_power true  -max_transition false  -max_capacitance true  -active false
set_scenario_status FUNC_Typical -all -active true
set_scenario_status FUNC_Slow    -setup true  -hold false -leakage_power true  -dynamic_power true  -max_transition true   -max_capacitance false  -active false
```

The FUNC_Typical is active .

**What kind of corners, modes, and scenarios would we test in a typical design and why?**

In a standard design, the particular corners, modes, and scenarios tested vary based on the design requirements and intended application. Nonetheless, here are some frequently tested factors and their importance:

Corners

Process, Voltage, and Temperature (PVT) Corners:

- Process Corners: Variations in the manufacturing process can lead to differences in the electrical characteristics of transistors. Typical process corners include:

  Typical-Typical (TT): Represents the nominal manufacturing process.

  Slow-Slow (SS): Represents the slowest process, where both NMOS and PMOS transistors have worse-than-nominal performance.

  Fast-Fast (FF): Represents the fastest process, where both NMOS and PMOS transistors have better-than-nominal performance.

  Slow-Fast (SF) and Fast-Slow (FS): Mixed process corners where one type of transistor is fast, and the other is slow.

- Voltage Corners: Variations in the supply voltage can affect the performance and power consumption. Common voltage corners are:

  Nominal Voltage: The standard operating voltage.

  Low Voltage: Lower than nominal, which can reduce power consumption but might affect performance.

  High Voltage: Higher than nominal, which can improve performance but increase power consumption and heat.

- Temperature Corners: The temperature range in which the device operates. Typical temperature corners include:

  Cold (e.g., -40°C): Ensures the device operates correctly at low temperatures.

  Room Temperature (e.g., 25°C): Standard operating condition.

  Hot (e.g., 125°C): Ensures the device operates correctly at high temperatures.

Testing across PVT corners helps to ensure the design performs reliably under all manufacturing, voltage, and temperature variations.

Modes

Operating Modes:

- Functional Mode: Ensures the design performs its intended functionality.
- Test Mode: Used during manufacturing testing to check for defects in the silicon.
- Low Power Mode: Tests the design’s behavior in power-saving states.
- Performance Mode: Tests the design at maximum performance conditions.

Verifying different operating modes ensures the design meets the intended functionality, performance, and power consumption targets under various usage scenarios.

Scenarios

Use Case Scenarios:

- Typical Use Case: Tests the design under normal operating conditions and expected usage patterns.
- Worst-Case Scenario: Tests the design under extreme conditions that could stress the system, such as maximum data rates, highest clock frequencies, and peak power consumption.
- Best-Case Scenario: Ensures the design performs optimally under ideal conditions.

Testing across different scenarios helps to validate the design’s robustness and reliability, ensuring it can handle both typical and extreme conditions encountered in real-world applications.

**[#P3.2_Q4] What are Multibit Registers?**

Multibit registers are sequential elements in digital circuits that store and synchronize multiple bits of data. Unlike single-bit registers that handle individual bits, multibit registers manage multiple bits simultaneously as a single unit. Here are some key features, characteristics:

Key Features and Characteristics:

1. Data Storage:

Multibit registers store multiple bits of data as a group. They contain multiple flip-flops or latches within a single register, each capable of storing a single bit.

2. Synchronization:

All bits within a multibit register are synchronized to a common clock signal, ensuring that the stored data is sampled and updated simultaneously, which maintains data integrity and avoids timing issues.

3. Width and Word Size:

The width of a multibit register refers to the number of bits it can store, such as a 4-bit register storing four bits of data. The word size represents the number of bits in a single data word that can be stored or processed as a unit.

4. Parallel Data Handling:

Multibit registers efficiently capture and store multiple bits of data simultaneously, enabling parallel processing and operations in digital systems.

Advantages:

1. Efficiency:

By grouping multiple single-bit flip-flops into a single register, multibit registers reduce the need for multiple clock drivers and associated circuitry, which simplifies the design and improves efficiency.

2. Power Savings:

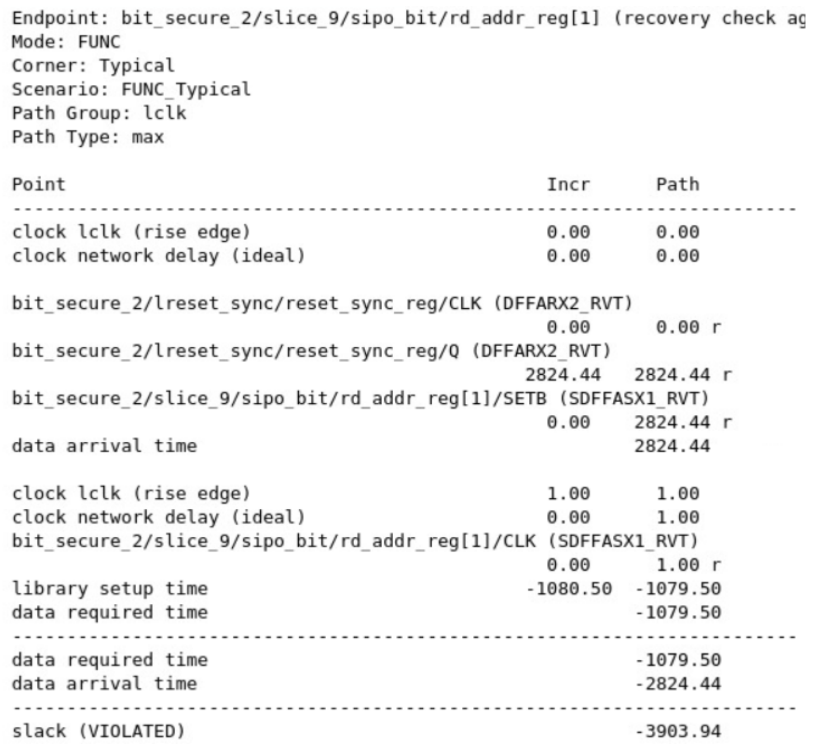
Sharing common clock and control signals like reset and enable across multiple bits reduces dynamic power consumption.

3. Compact Design:

Using multibit registers reduces silicon area and routing complexity, leading to more compact and manageable designs.

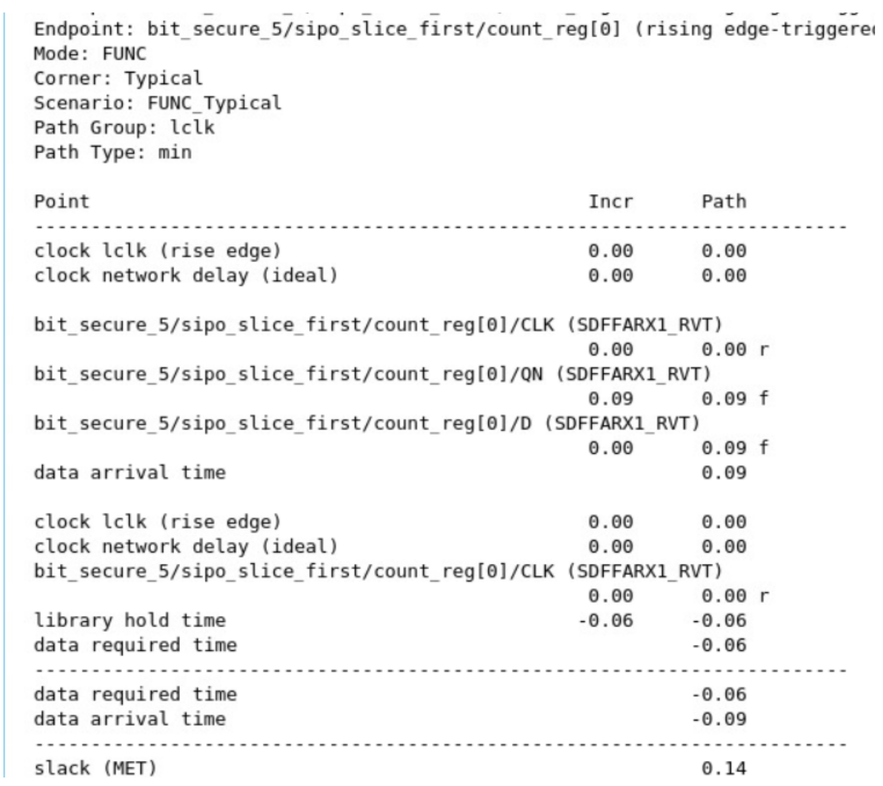
**[#P3.3_Q1] What is the slack for the SETUP and HOLD constraint? Attach a print screen for both.**

**Setup:**



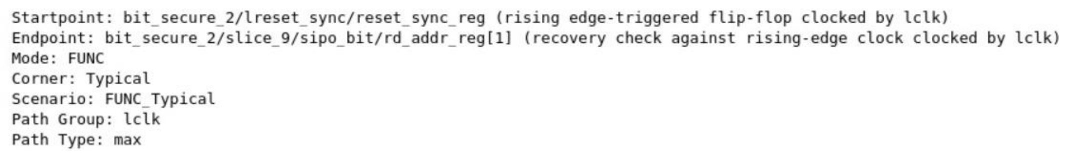
```
Endpoint: bit_secure_2/slice_9/sipo_bit/rd_addr_reg[1] (recovery check ag
Mode: FUNC
Corner: Typical
Scenario: FUNC_Typical
Path Group: lclk
Path Type: max

Point                                               Incr      Path
---------------------------------------------------------------------
clock lclk (rise edge)                              0.00      0.00
clock network delay (ideal)                         0.00      0.00

bit_secure_2/lreset_sync/reset_sync_reg/CLK (DFFARX2_RVT)
                                                    0.00      0.00 r
bit_secure_2/lreset_sync/reset_sync_reg/Q (DFFARX2_RVT)
                                                  2824.44   2824.44 r
bit_secure_2/slice_9/sipo_bit/rd_addr_reg[1]/SETB (SDFFASX1_RVT)
                                                    0.00    2824.44 r
data arrival time                                           2824.44

clock lclk (rise edge)                              1.00      1.00
clock network delay (ideal)                         0.00      1.00
bit_secure_2/slice_9/sipo_bit/rd_addr_reg[1]/CLK (SDFFASX1_RVT)
                                                    0.00      1.00 r
library setup time                                -1080.50  -1079.50
data required time                                          -1079.50
---------------------------------------------------------------------
data required time                                          -1079.50
data arrival time                                           -2824.44
---------------------------------------------------------------------
slack (VIOLATED)                                            -3903.94
```

**Hold:**



```
Endpoint: bit_secure_5/sipo_slice_first/count_reg[0] (rising edge-triggere
Mode: FUNC
Corner: Typical
Scenario: FUNC_Typical
Path Group: lclk
Path Type: min

Point                                              Incr      Path
-------------------------------------------------------------------
clock lclk (rise edge)                             0.00      0.00
clock network delay (ideal)                        0.00      0.00

bit_secure_5/sipo_slice_first/count_reg[0]/CLK (SDFFARX1_RVT)
                                                   0.00      0.00 r
bit_secure_5/sipo_slice_first/count_reg[0]/QN (SDFFARX1_RVT)
                                                   0.09      0.09 f
bit_secure_5/sipo_slice_first/count_reg[0]/D (SDFFARX1_RVT)
                                                   0.00      0.09 f
data arrival time                                            0.09

clock lclk (rise edge)                             0.00      0.00
clock network delay (ideal)                        0.00      0.00
bit_secure_5/sipo_slice_first/count_reg[0]/CLK (SDFFARX1_RVT)
                                                   0.00      0.00 r
library hold time                                 -0.06     -0.06
data required time                                          -0.06
-------------------------------------------------------------------
data required time                                          -0.06
data arrival time                                           -0.09
-------------------------------------------------------------------
slack (MET)                                                  0.14
```

**[#P3.3_Q2] Read the reports and find the starting and ending point of the critical path. If needed use the "-through" option to specify the specific path between those cells. Fill the code for the command: (the through is optional, you can delete the flag if you don't use it).**

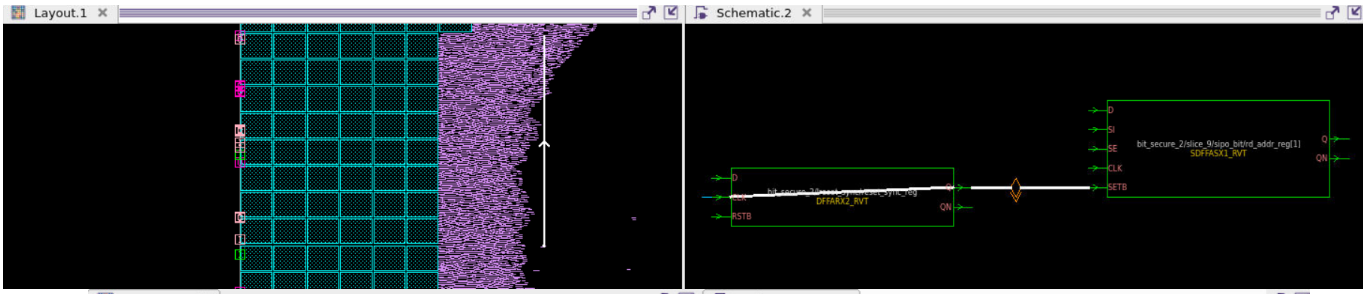


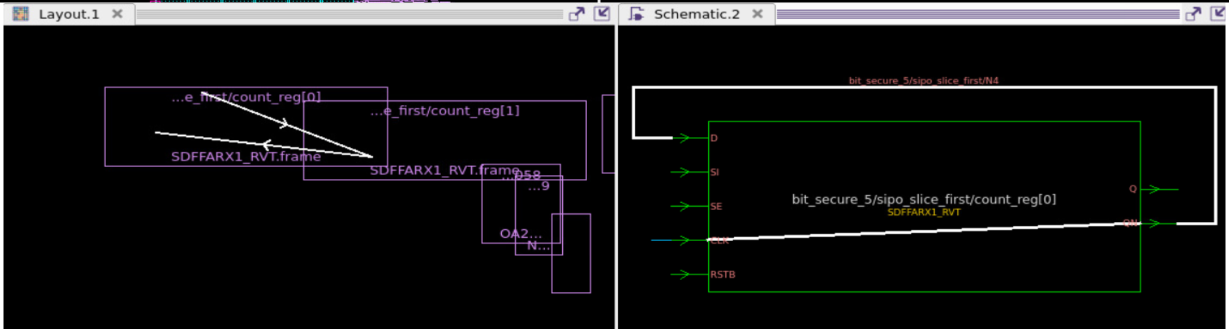
```
Startpoint: bit_secure_2/lreset_sync/reset_sync_reg (rising edge-triggered flip-flop clocked by lclk)
Endpoint: bit_secure_2/slice_9/sipo_bit/rd_addr_reg[1] (recovery check against rising-edge clock clocked by lclk)
Mode: FUNC
Corner: Typical
Scenario: FUNC_Typical
Path Group: lclk
Path Type: max
```

**[#P3.3_Q3] While we still in the context of the path (done by section d) select the "Schematic View" and take a snapshot of the critical paths for SETUP and HOLD:**

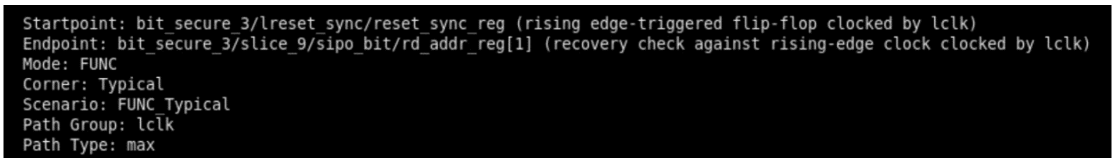
**Setup:**

**Hold:**



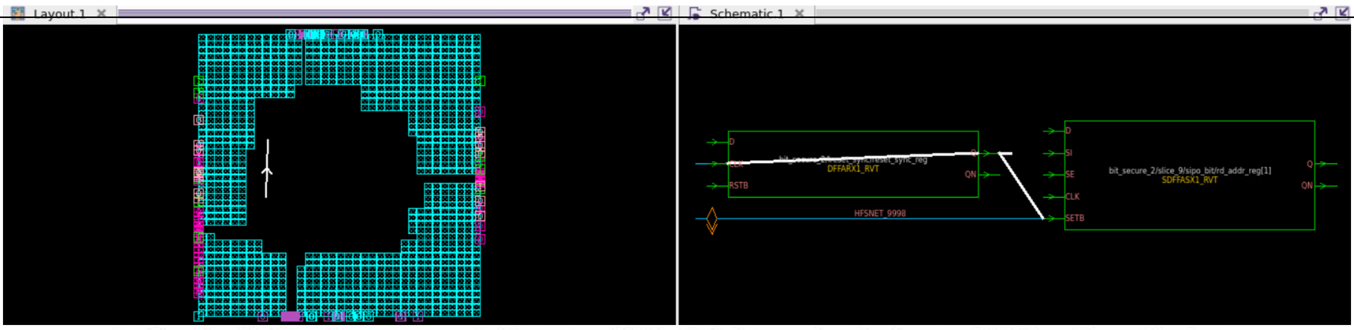


**[#P3.3_Q4] Find an additional timing path with a major negative slack. To do that, use the report_timing command with the following flag: "-max_path". Try to find a pattern to those paths. Hint: Maybe you can find a common factor that can cause the problematic slack. Use this pattern to fill the code for the "set_false_path" command.**



```
Startpoint: bit_secure_3/lreset_sync/reset_sync_reg (rising edge-triggered flip-flop clocked by lclk)
Endpoint: bit_secure_3/slice_9/sipo_bit/rd_addr_reg[1] (recovery check against rising-edge clock clocked by lclk)
Mode: FUNC
Corner: Typical
Scenario: FUNC_Typical
Path Group: lclk
Path Type: max
```

**[#P3.3_Q5] After you ran the "final_opto" compilation, change your selection again to the same path you chose for the SETUP path in [#P3.3_Q3].**



```
Startpoint: piso_secure_1/temp_reg[14] (rising edge-triggered flip-flop clocke
Endpoint: piso_secure_1/temp_reg[15] (rising edge-triggered flip-flop clocked
Mode: FUNC
Corner: Typical
Scenario: FUNC_Typical
Path Group: lclk
Path Type: max

Point                                               Incr      Path
------------------------------------------------------------------------
clock lclk (rise edge)                              0.00      0.00
clock network delay (ideal)                         0.00      0.00

piso_secure_1/temp_reg[14]/CLK (SDFFARX1_RVT)       0.00      0.00 r
piso_secure_1/temp_reg[14]/Q (SDFFARX1_RVT)         0.14      0.14 r
ctmi_26693/Y (AO22X2_RVT)                           0.12      0.26 r
HFSINV_291_19184/Y (INVX16_RVT)                     0.04      0.30 f
HFSINV_255_19183/Y (INVX16_RVT)                     0.08      0.38 r
piso_secure_1/temp_reg[15]/D (SDFFARX1_RVT)         0.05      0.43 r
data arrival time                                             0.43

clock lclk (rise edge)                              1.00      1.00
clock network delay (ideal)                         0.00      1.00
piso_secure_1/temp_reg[15]/CLK (SDFFARX1_RVT)       0.00      1.00 r
library setup time                                 -0.13      0.87
data required time                                            0.87
------------------------------------------------------------------------
data required time                                            0.87
data arrival time                                            -0.43
------------------------------------------------------------------------
slack (MET)                                                   0.44
```

**[#P3.3_Q6] Analyze those reports and answer the following questions: - Setup timing: What is the first stage in the compile fusion flow that fixes the SETUP violations?**

In the compile fusion process, the initial_opto stage, also referred to as the "Fusion Analysis" stage, is responsible for addressing setup violations. During this phase, the tool examines the design to identify any violations of setup time requirements. Multiple rounds of optimization are conducted to enhance metrics such as timing, power, and logical DRC, including clock concurrent optimization (CCD). Additionally, optimizing high fanout nets (HFN) can effectively resolve many setup issues.

**- Hold timing: From which stage in the general chip implementation (Synthesis, STA, Floorplan, Placement…) should we check the hold violations and why?**

In the general chip implementation flow, it is important to check hold violations after the placement stage and before the static timing analysis (STA) stage. Placement affects timing paths and interconnect lengths, so ensuring proper hold times before STA is crucial. During the STA stage, hold violations can be detected and addressed, allowing designers to identify timing issues before moving on to the routing stage.

Fixing hold violations early in the flow can help avoid complex and time-consuming routing optimizations and iterations, ensuring that the design meets required hold time constraints for reliable operation. After synthesis and placement, clock tree synthesis (CTS) is performed using the clock_opt command, creating a non-ideal clock. This helps identify and check hold time violations, allowing potential issues to be resolved early and ensuring hold time requirements are met during STA.

**Area: What can we derive out of the following information from the area report?**

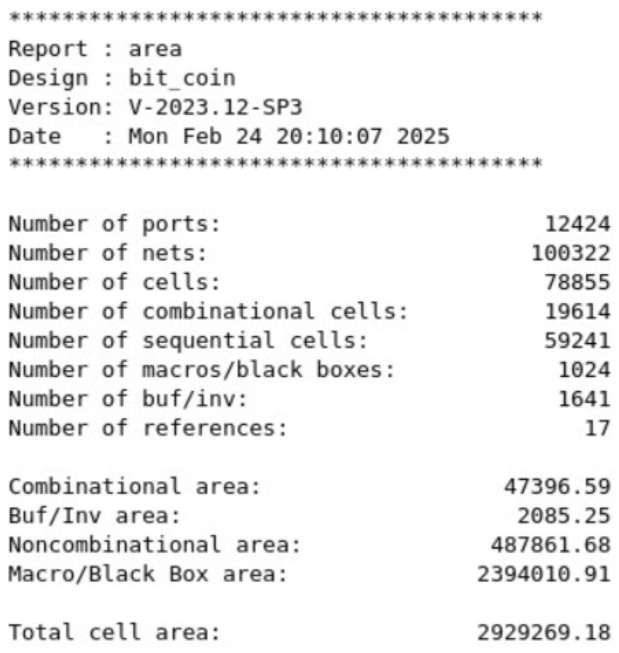
Number of ports = 12242

The ratio between number of the combinational cells and the sequential cells = 19614/59241=0.331

Number of macros = 1024

Number of buffers and inverters = 1641

Area of cells + area of macros = 2929269.18+23941010.91=5323280.09



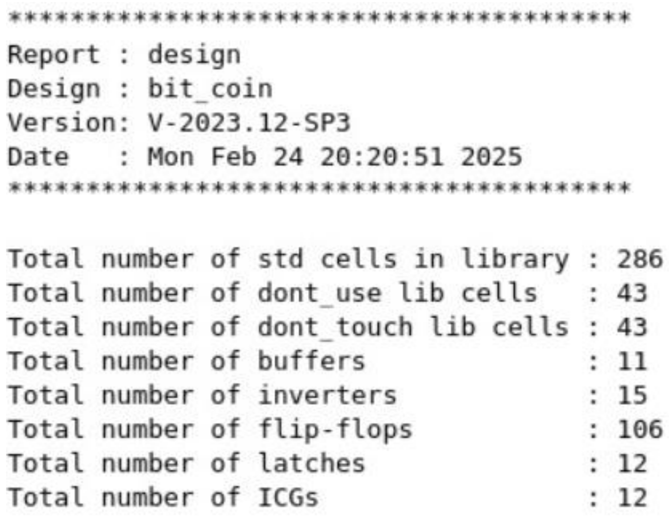
```
****************************************
Report : area
Design : bit_coin
Version: V-2023.12-SP3
Date   : Mon Feb 24 20:10:07 2025
****************************************

Number of ports:                        12424
Number of nets:                        100322
Number of cells:                        78855
Number of combinational cells:          19614
Number of sequential cells:             59241
Number of macros/black boxes:            1024
Number of buf/inv:                       1641
Number of references:                      17

Combinational area:                  47396.59
Buf/Inv area:                         2085.25
Noncombinational area:              487861.68
Macro/Black Box area:              2394010.91

Total cell area:                   2929269.18
```

**- Design: What can we derive out of the following information from the design report?**

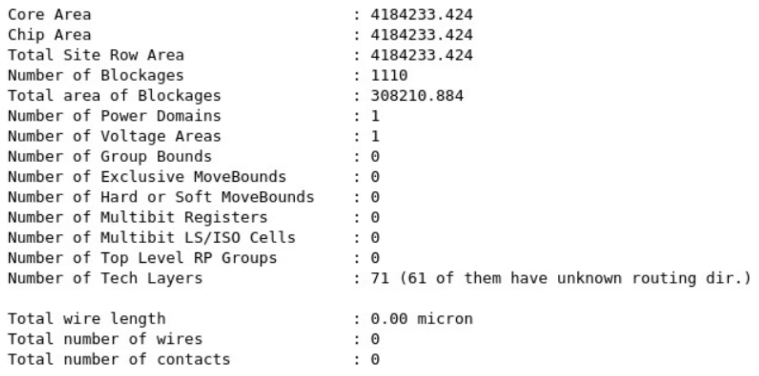
ICG - integrated clock gates = 12

Latches = 12

Number of Power Domains = 1



```
****************************************
Report : design
Design : bit_coin
Version: V-2023.12-SP3
Date   : Mon Feb 24 20:20:51 2025
****************************************

Total number of std cells in library : 286
Total number of dont_use lib cells   : 43
Total number of dont_touch lib cells : 43
Total number of buffers              : 11
Total number of inverters            : 15
Total number of flip-flops           : 106
Total number of latches              : 12
Total number of ICGs                 : 12
```



```
Core Area                           : 4184233.424
Chip Area                           : 4184233.424
Total Site Row Area                 : 4184233.424
Number of Blockages                 : 1110
Total area of Blockages             : 308210.884
Number of Power Domains             : 1
Number of Voltage Areas             : 1
Number of Group Bounds              : 0
Number of Exclusive MoveBounds      : 0
Number of Hard or Soft MoveBounds   : 0
Number of Multibit Registers        : 0
Number of Multibit LS/ISO Cells     : 0
Number of Top Level RP Groups       : 0
Number of Tech Layers               : 71 (61 of them have unknown routing dir.)

Total wire length                   : 0.00 micron
Total number of wires               : 0
Total number of contacts            : 0
```

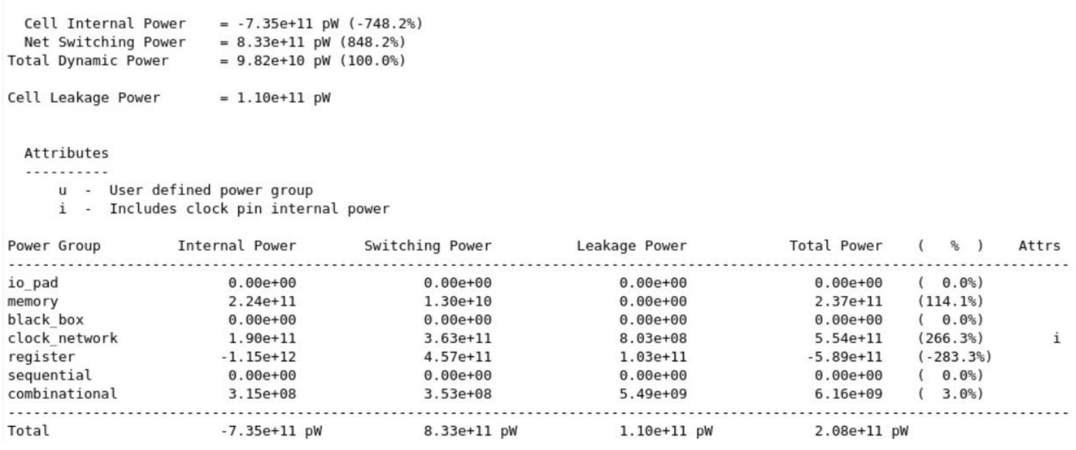
**Power: What can we derive out of the following information from the power report?**

**o What is the static, dynamic and total power after all compile_fusion command?**

Static/Leakage power: 1.43e+11 pw

Dynamic power: 7.97e+11 pw

Total power: 9.64e+11 pw



```
  Cell Internal Power    = -7.35e+11 pW (-748.2%)
  Net Switching Power    = 8.33e+11 pW (848.2%)
Total Dynamic Power      = 9.82e+10 pW (100.0%)

Cell Leakage Power       = 1.10e+11 pW


  Attributes
  ----------
      u  -  User defined power group
      i  -  Includes clock pin internal power

Power Group         Internal Power       Switching Power         Leakage Power           Total Power    (   %  )    Attrs
--------------------------------------------------------------------------------------------------------------------------
io_pad                    0.00e+00              0.00e+00              0.00e+00              0.00e+00    (  0.0%)
memory                    2.24e+11              1.30e+10              0.00e+00              2.37e+11    (114.1%)
black_box                 0.00e+00              0.00e+00              0.00e+00              0.00e+00    (  0.0%)
clock_network             1.90e+11              3.63e+11              8.03e+08              5.54e+11    (266.3%)        i
register                 -1.15e+12              4.57e+11              1.03e+11             -5.89e+11    (-283.3%)
sequential                0.00e+00              0.00e+00              0.00e+00              0.00e+00    (  0.0%)
combinational             3.15e+08              3.53e+08              5.49e+09              6.16e+09    (  3.0%)
--------------------------------------------------------------------------------------------------------------------------
Total                    -7.35e+11 pW           8.33e+11 pW           1.10e+11 pW           2.08e+11 pW
```

**What does each of the power types mean?**

**Dynamic Power**

Dynamic power, also known as switching power, is the energy consumed by a circuit during its active operations when transistors switch between different states. This type of power consumption occurs due to the charging and discharging of capacitive loads and the brief period during transitions when both NMOS and PMOS transistors are simultaneously on, leading to short-circuit currents. Factors such as the frequency of switching, activity levels, and capacitance influence dynamic power consumption.

**Static Power**

Static power, also called leakage power, is the energy consumed by a circuit when it is not actively switching states. Even when transistors are turned off, there is still a leakage current that flows through them, contributing to static power consumption.

This leakage can occur through subthreshold currents, gate oxide layers, and reverse-biased PN junctions. As transistors become smaller and supply voltages decrease, static power becomes increasingly significant due to higher leakage currents.

**Total Power**

The total power of an integrated circuit (IC) is the sum of dynamic and static power. It encompasses the energy consumed during both active switching and idle or standby periods, providing a comprehensive measure of the circuit's overall power consumption.

**Implications for IC Design**

Optimizing power consumption in IC design involves addressing both dynamic and static power components. For dynamic power, techniques such as reducing supply voltage, lowering clock frequency, minimizing load capacitance, and optimizing switching activity are effective. On the other hand, static power can be reduced by using high-threshold voltage transistors, implementing power gating to deactivate unused circuit blocks, and adopting low-leakage process technologies.

Overall power optimization requires balancing trade-offs between dynamic and static power. Advanced power management techniques, such as dynamic voltage and frequency scaling (DVFS), can be employed to achieve this balance. Additionally, designing efficient clock gating helps reduce unnecessary switching, further optimizing power consumption and ensuring the efficient and reliable performance of electronic devices.