CENTER OF NANOELECTRONICS & DEVICES

CND211:Advanced Digital Design

Complete ASIC Flow of I2C communication protocol

Section #: ----15-----

Submitted by:

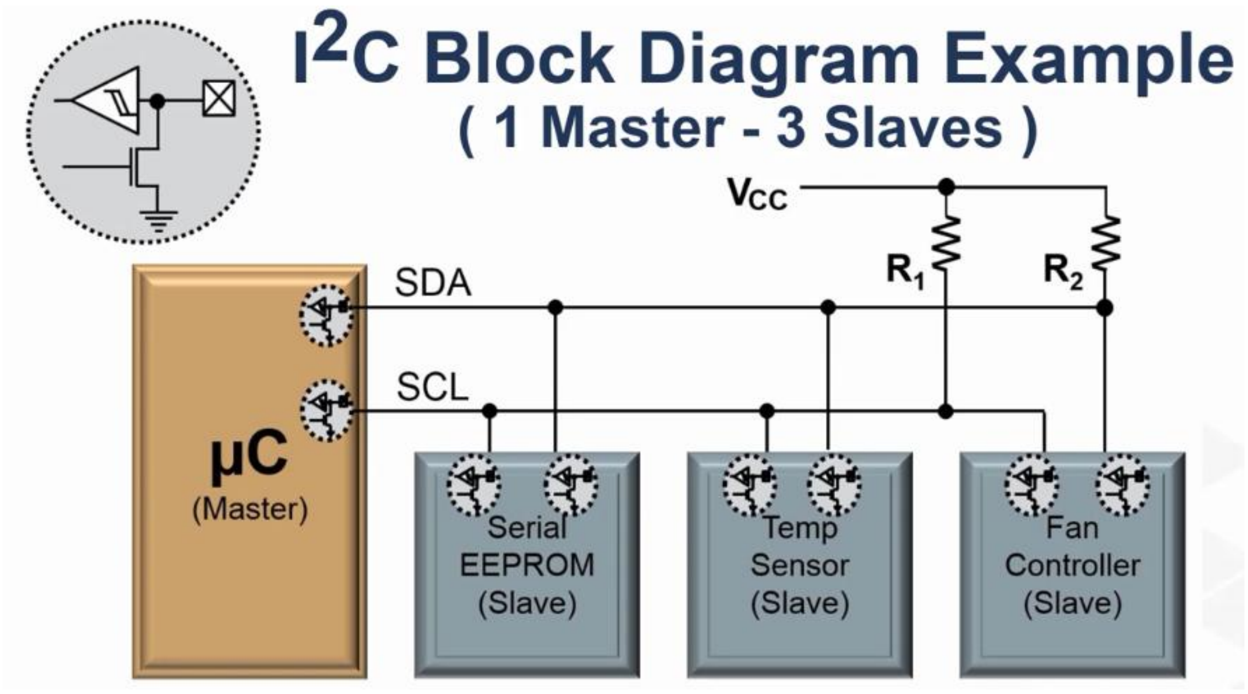
Student Name	ID
Asmaa Tharwat	V23010254
Alaa Ibrahim Elshahawy	V23010211
Esraa Ahmed Hemdan	V23010292
Habiba-Allah Tarek Elgindy	V23010444
Jaida Adel Aly Sakr	V23010446

Submitted to TA: Eng. Amira

INTRODUCTION

I2C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. It is most suitable for applications requiring occasional communication over a short distance between many devices,Specially we use this protocol between sensors. The I2C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

Figure shows a typical I2C bus for an embedded system, where multiple slave devices are used. The microcontroller represents the I2C master, and controls the IO expanders, various sensors, EEPROM, Fan controller. All of which are controlled with only 2 pins from the master. **SDL** and **SCL**



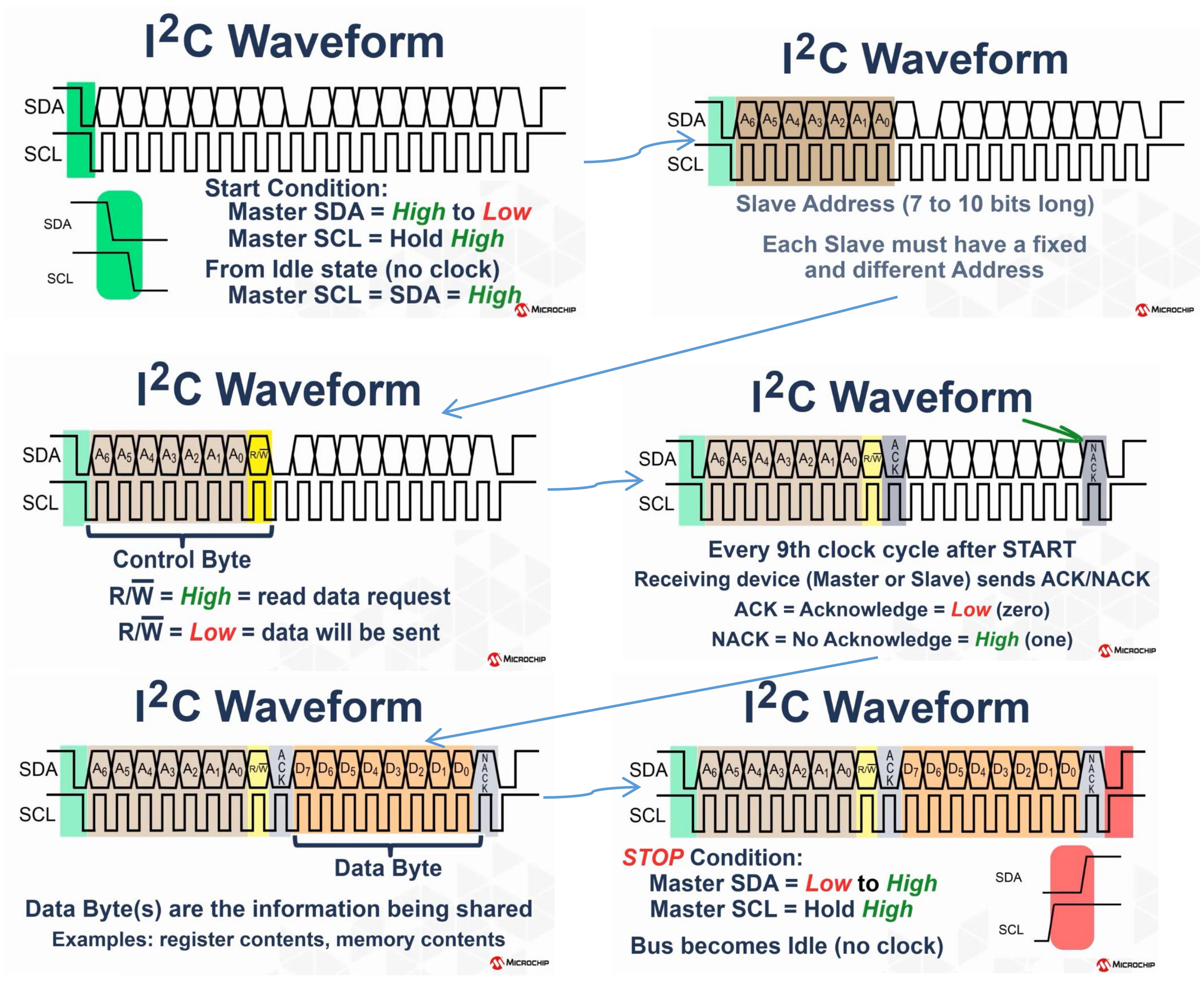
From Figure we notice that SCL & SDL need to connect with pull up resistor.

Functionality

I2C (Inter-Integrated Circuit) is a serial communication protocol designed for low-speed communication between integrated circuits (ICs) on a circuit board. Here's a summary of its key features:

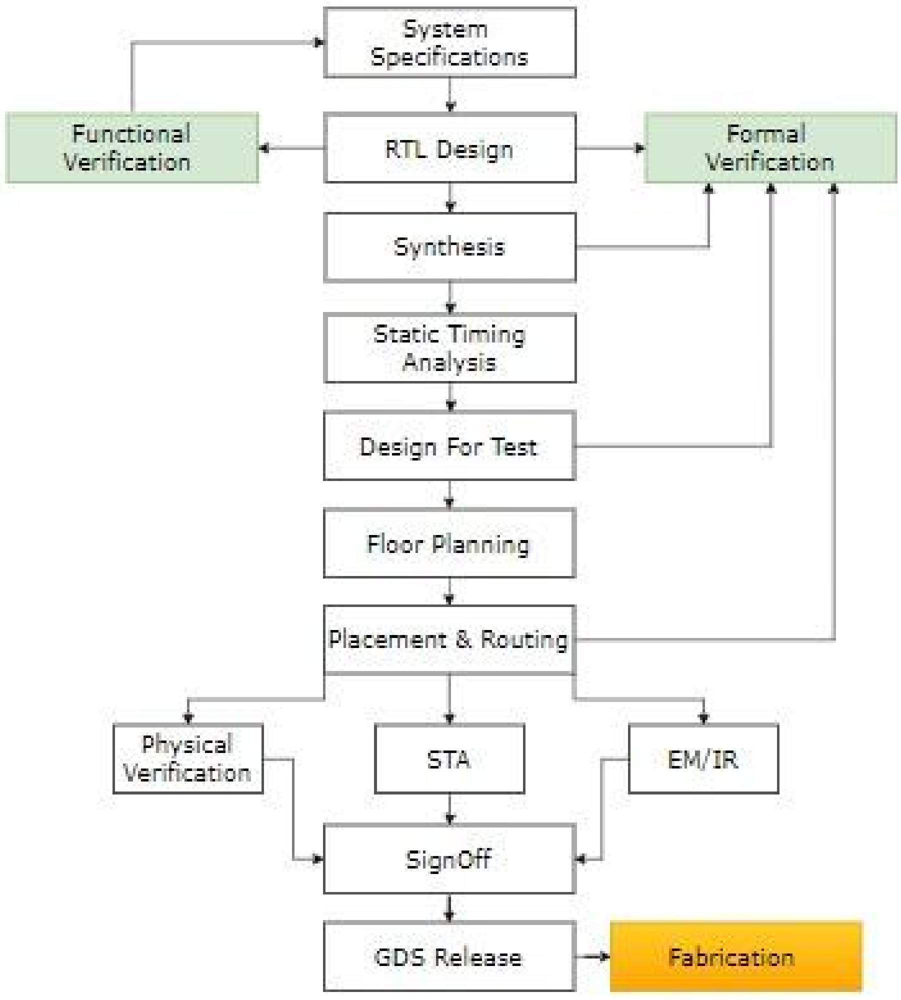
- **Serial communication:** Data is transmitted one bit at a time over two wires, reducing pin count compared to parallel communication.
- **Bi-directional:** Allows data transfer in both directions between devices.
- **Multi-master:** Multiple devices can initiate communication as a "master" on the bus, with arbitration to prevent collisions.
- **Simple and efficient:** I2C uses a minimal set of signals (SCL - clock, SDA - data) and a straightforward protocol, making it suitable for low-complexity applications.

I2C devices are identified by unique addresses, enabling a master to target specific slaves for data exchange. The protocol includes mechanisms for error detection and acknowledgement, ensuring data integrity.


I2C Waveform
SDA
SCL
Start Condition:
Master SDA = High to Low
Master SCL = Hold High
From Idle state (no clock)
Master SCL = SDA = High
Slave Address (7 to 10 bits long)
Each Slave must have a fixed and different Address
Control Byte
R/W = High = read data request
R/W = Low = data will be sent
Every 9th clock cycle after START
Receiving device (Master or Slave) sends ACK/NACK
ACK = Acknowledge = Low (zero)
NACK = No Acknowledge = High (one)
Data Byte
Data Byte(s) are the information being shared
Examples: register contents, memory contents
STOP Condition:
Master SDA = Low to High
Master SCL = Hold High
Bus becomes Idle (no clock)
MICROCHIP

ASIC FLOW

After we understand the I2C Protocol and its functionality , we need to start understanding the flow of ASIC operation and each step .



1.1 RTL Design

We have a 4 main modules for 12C (Synopsys tool: VCS):

- I2C_master_top
- I2C_master_byte_ctrl
- I2C_master_bit_ctrl
- I2C_master_difiens

1.2 Synthesis:

Synthesis is achieving an optimal gate level netlist from HDL code

(Synopsys tool: Design compiler).

The logic synthesis process consists of 3 steps:

- **Translation**:
  Translation involves transforming a HDL (RTL) description to gates.

- **Optimization**:
  Synthesis optimizes the design for various metrics, such as area, power consumption, and timing. By leveraging advanced algorithms, it improves the performance and efficiency of the resulting netlist.
- **Mapping**:

Technology mapping is the phase of logic synthesis when gates are selected from a technology library to implement the circuit through read target library and link it which The target library is used during compile to create a technology-specific gate-level netlist.

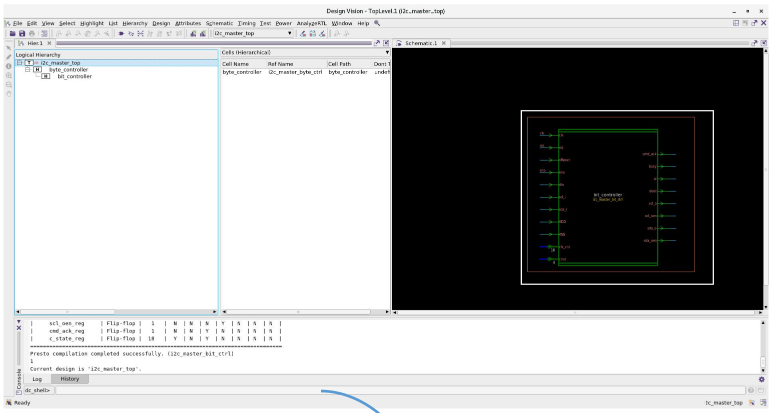
Start The synthesis step:

After get the desired tecLib which we use this command to set the top module for i2c:

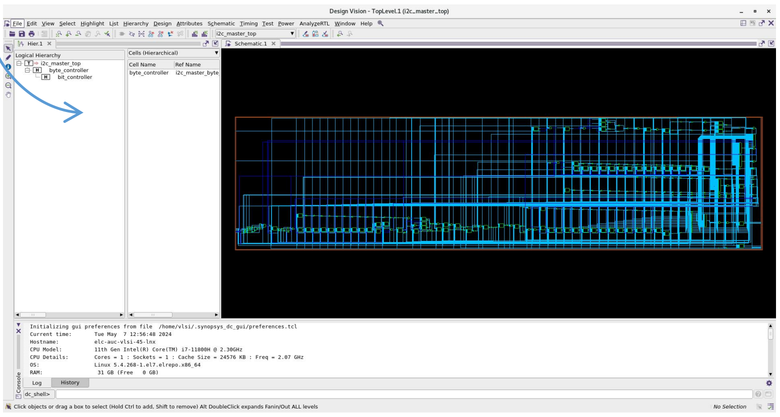
```
set design i2c_master_top.
```

After writing this commend we will start elaborate our design and it shows its schematic view for I2C

```
elaborate $design -lib work
```



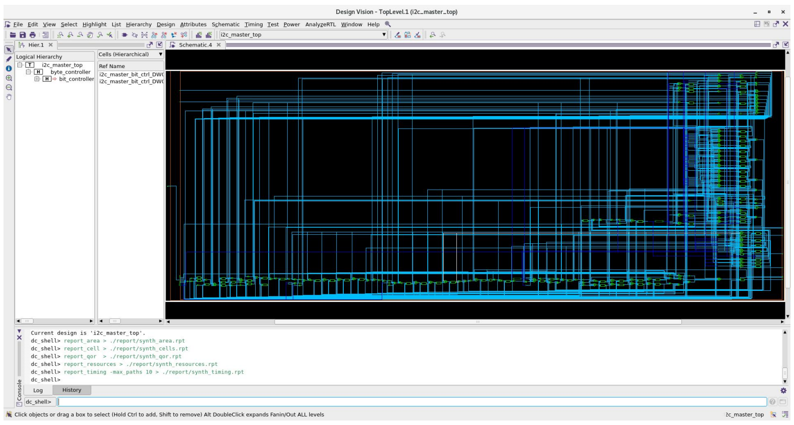
```
|     scl_oen_reg     | Flip-flop |   1   |  N  |  N  |  N  |  Y  |  N  |  N  |  N  |
|     cmd_ack_reg     | Flip-flop |   1   |  N  |  N  |  Y  |  N  |  N  |  N  |  N  |
|     c_state_reg     | Flip-flop |  18   |  Y  |  N  |  Y  |  N  |  N  |  N  |  N  |
===============================================================================
Presto compilation completed successfully. (i2c_master_bit_ctrl)
1
Current design is 'i2c_master_top'.
```



```
Initializing gui preferences from file  /home/vlsi/.synopsys_dc_gui/preferences.tcl
Current time:       Tue May  7 12:56:48 2024
Hostname:           elc-auc-vlsi-45-lnx
CPU Model:          11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz
CPU Details:        Cores = 1 : Sockets = 1 : Cache Size = 24576 KB : Freq = 2.07 GHz
OS:                 Linux 5.4.268-1.el7.elrepo.x86_64
RAM:                 31 GB (Free   0 GB)
```

When we write this command it will show the current desigh which it i2c_master_top

```
current_design
```



```
Current design is 'i2c_master_top'.
dc_shell> report_area > ./report/synth_area.rpt
dc_shell> report_cell > ./report/synth_cells.rpt
dc_shell> report_qor  > ./report/synth_qor.rpt
dc_shell> report_resources > ./report/synth_resources.rpt
dc_shell> report_timing -max_paths 10 > ./report/synth_timing.rpt
dc_shell>
```

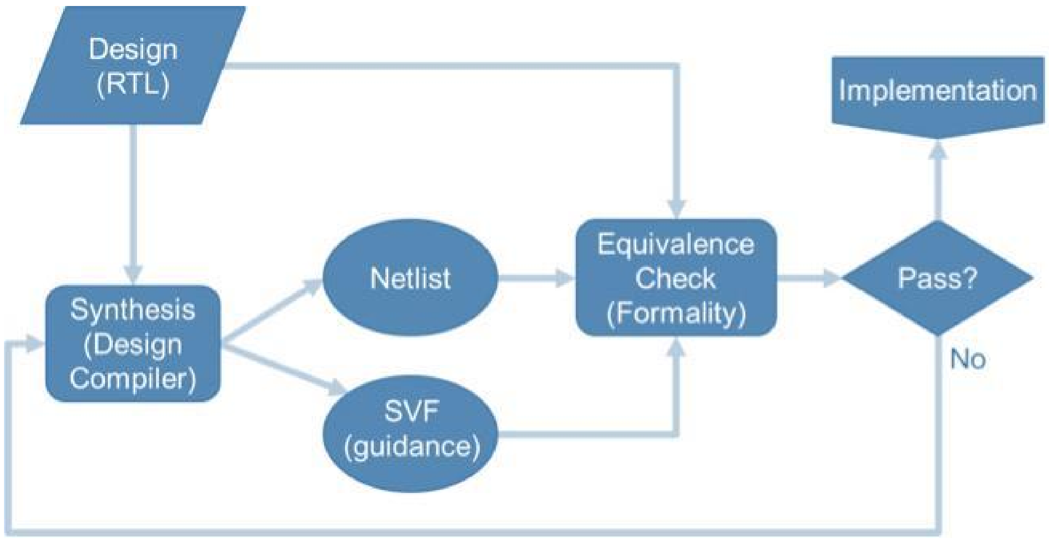
Note: this design will setup timing constraints which **(clock_period=5,Clock_Uncertainty =0.3,**

Transition = 0.2,Clock_latency = 0.25,Input_delay 10 % Clk_period=.5 ,Output_delay 10 %Clk_period =.5)

1.3 STA & Formality

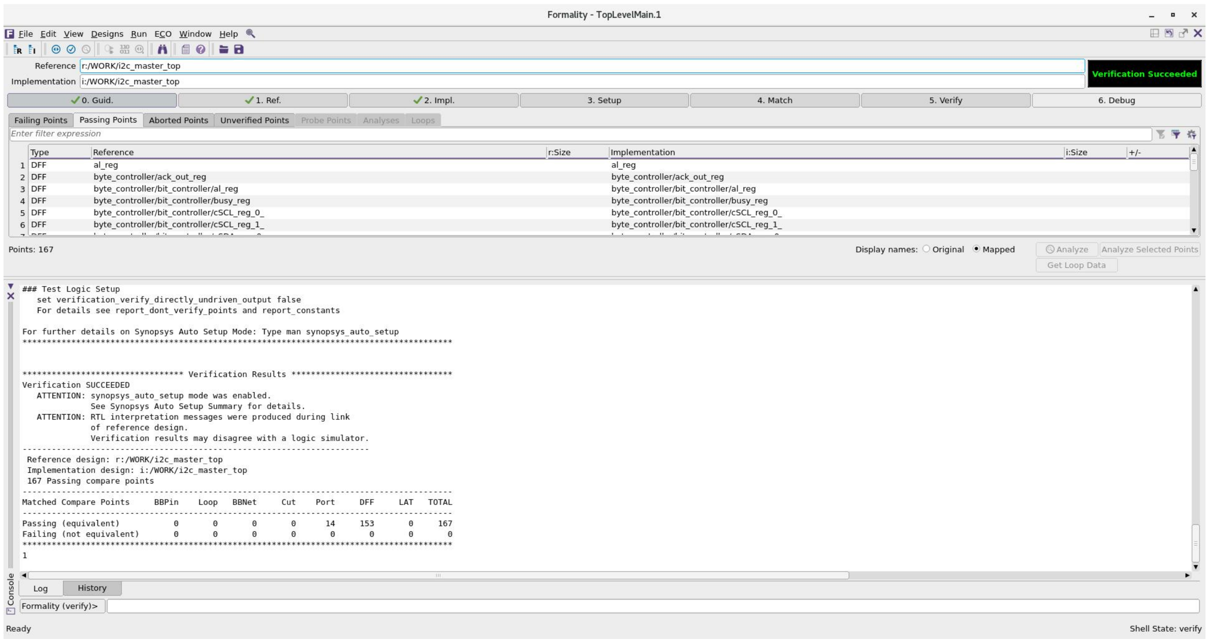
According to Formal Verification:

Formal Verification is an alternative to verification through simulation. As design become larger and more complex and require more simulation vectors, regression testing with traditional **simulation tools becomes a bottleneck in the design flow.**



Steps for doing formal verification using tool:

- ✓ **Guide Ref implement**



Reference r:/WORK/i2c_master_top
Implementation i:/WORK/i2c_master_top

	Type	Reference	r:Size	Implementation	i:Size	+/-
1	DFF	al_reg		al_reg		
2	DFF	byte_controller/ack_out_reg		byte_controller/ack_out_reg		
3	DFF	byte_controller/bit_controller/al_reg		byte_controller/bit_controller/al_reg		
4	DFF	byte_controller/bit_controller/busy_reg		byte_controller/bit_controller/busy_reg		
5	DFF	byte_controller/bit_controller/cSCL_reg_0_		byte_controller/bit_controller/cSCL_reg_0_		
6	DFF	byte_controller/bit_controller/cSCL_reg_1_		byte_controller/bit_controller/cSCL_reg_1_		

Points: 167

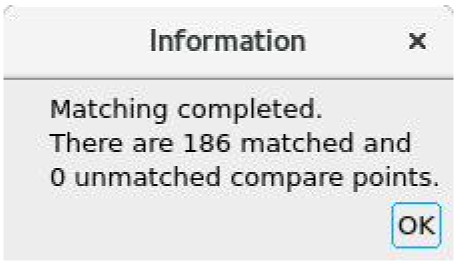
```
### Test Logic Setup
   set verification_verify_directly_undriven_output false
   For details see report_dont_verify_points and report_constants

For further details on Synopsys Auto Setup Mode: Type man synopsys_auto_setup
*************************************************************************************


********************************* Verification Results *********************************
Verification SUCCEEDED
   ATTENTION: synopsys_auto_setup mode was enabled.
              See Synopsys Auto Setup Summary for details.
   ATTENTION: RTL interpretation messages were produced during link
              of reference design.
              Verification results may disagree with a logic simulator.
-----------------------------------------------------------------------
 Reference design: r:/WORK/i2c_master_top
 Implementation design: i:/WORK/i2c_master_top
 167 Passing compare points
-------------------------------------------------------------------------------------
Matched Compare Points     BBPin    Loop   BBNet     Cut    Port     DFF     LAT   TOTAL
-------------------------------------------------------------------------------------
Passing (equivalent)           0       0       0       0      14     153       0     167
Failing (not equivalent)       0       0       0       0       0       0       0       0
*************************************************************************************
1
```

- ✓

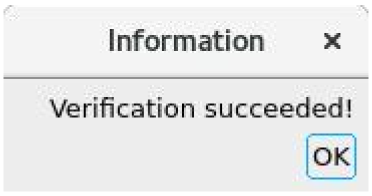
- ✓ Setup
- ✓ Match



Information

Matching completed.
There are 186 matched and
0 unmatched compare points.

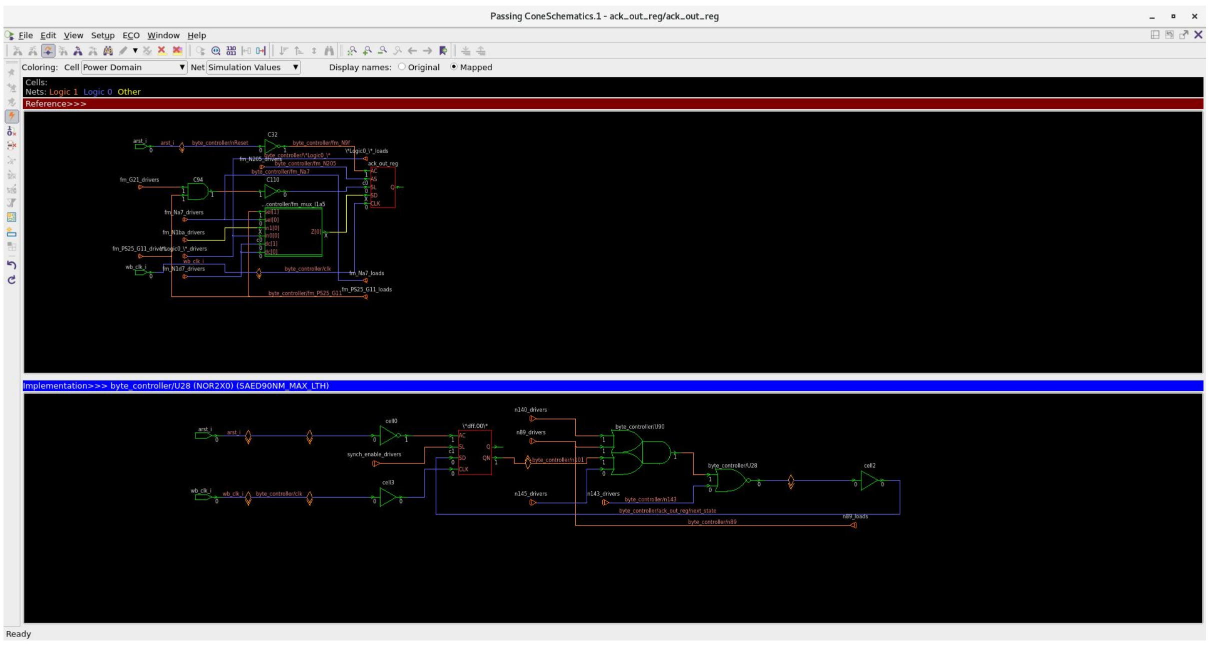
- ✓ Verify



Information

Verification succeeded!

- ✓ Debug: if we have errors to recover it, but we don't have :)



Passing ConeSchematics.1 - ack_out_reg/ack_out_reg

Coloring: Cell Power Domain Net Simulation Values Display names: Original Mapped

Cells:
Nets: Logic 1 Logic 0 Other

Reference>>>

Implementation>>> byte_controller/U28 (NOR2X0) (SAED90NM_MAX_LTH)

Ready

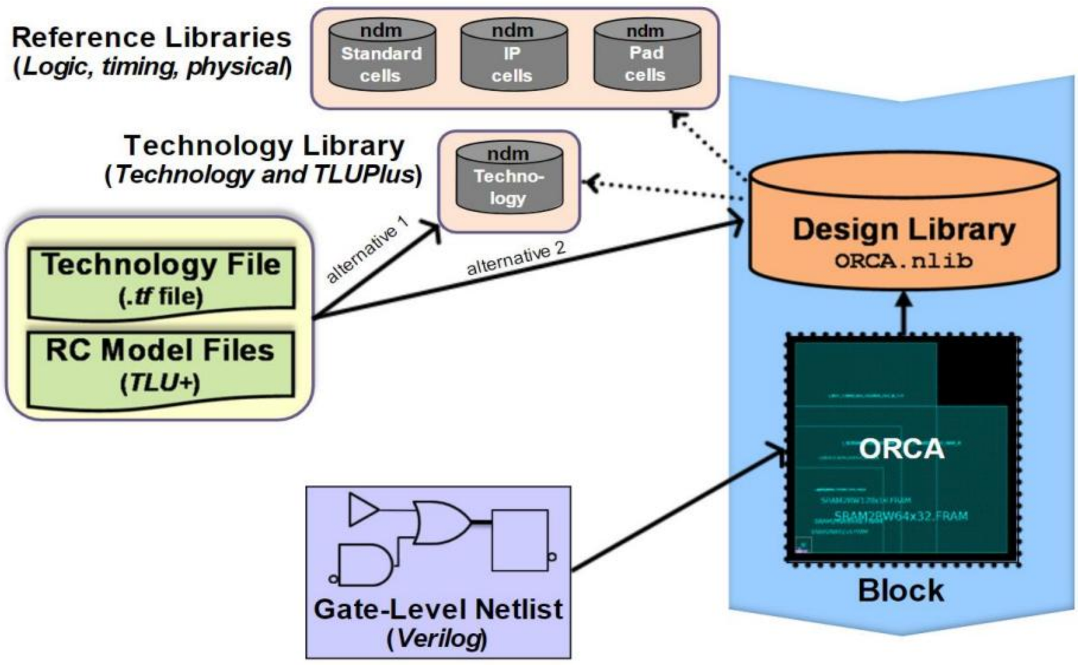
1.4 Setup and Floorplanning

Floorplanning is a critical stage that determines the physical layout of the chip.

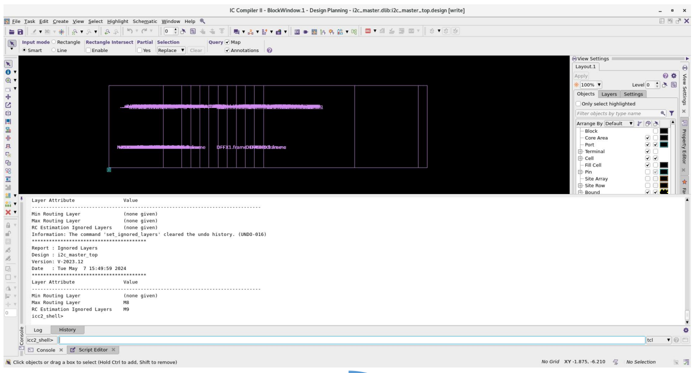
- **Block Placement:** Deciding where functional blocks (IP cores, memory, etc.) will reside on the chip.
- **Pin Placement:** Positioning input/output pins for external connections.
- **Power Optimization:** Ensuring efficient power distribution across the chip.

- The floorplan specifies the chip's size, gate placements, and wiring connections.

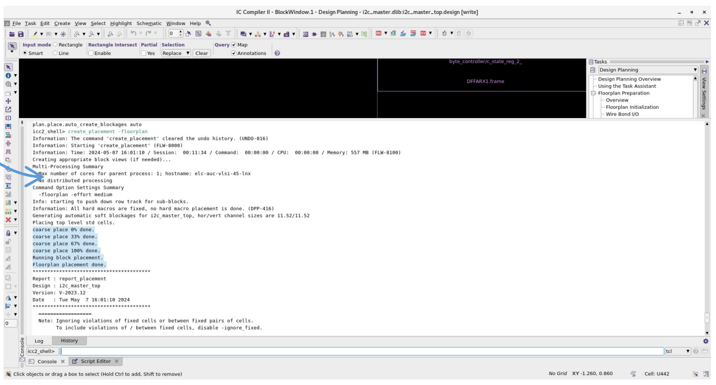
  According to this concept we conclude that we need **The main inputs required for the PnR flow in order to generate our GDS:**



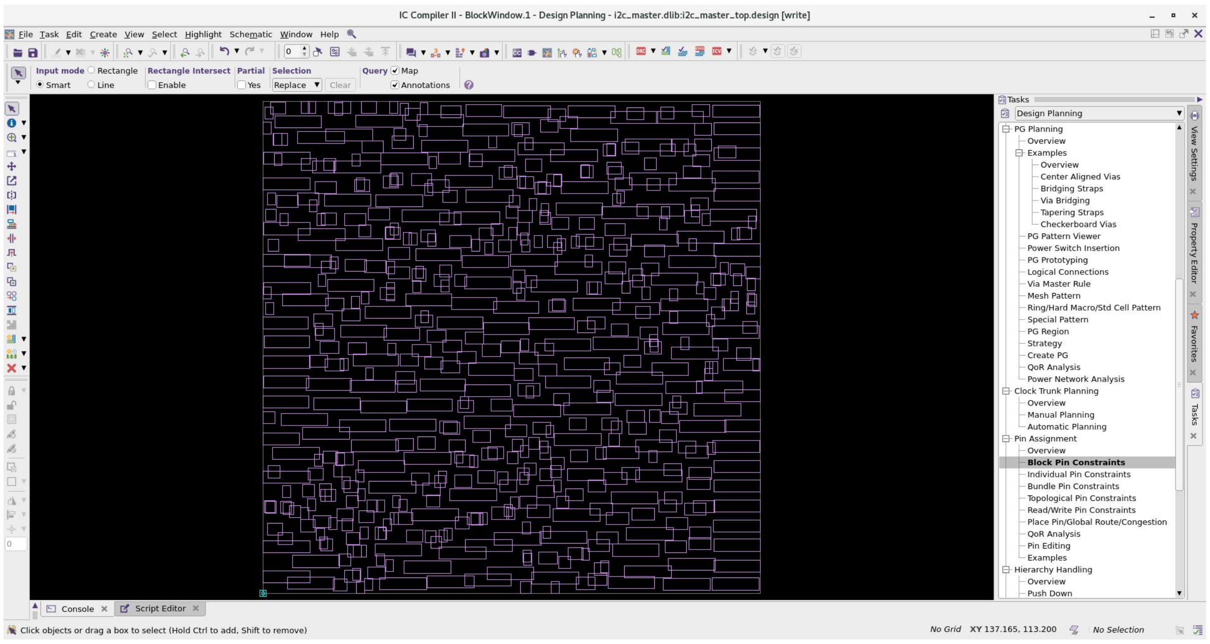
After floorplanning our design and setup:



```
Layer Attribute               Value
--------------------------------------------------------------------------------
Min Routing Layer             (none given)
Max Routing Layer             (none given)
RC Estimation Ignored Layers  (none given)
Information: The command 'set_ignored_layers' cleared the undo history. (UNDO-016)
****************************************
Report : Ignored Layers
Design : i2c_master_top
Version: V-2023.12
Date   : Tue May  7 15:49:59 2024
****************************************
Layer Attribute               Value
--------------------------------------------------------------------------------
Min Routing Layer             (none given)
Max Routing Layer             M8
RC Estimation Ignored Layers  M9
icc2_shell>
```



```
plan.place.auto_create_blockages auto
icc2_shell> create_placement -floorplan
Information: The command 'create_placement' cleared the undo history. (UNDO-016)
Information: Starting 'create_placement' (FLW-8000)
Information: Time: 2024-05-07 16:01:10 / Session:  00:11:34 / Command:  00:00:00 / CPU:  00:00:00 / Memory: 557 MB (FLW-8100)
Creating appropriate block views (if needed)...
Multi-Processing Summary
  Max number of cores for parent process: 1; hostname: elc-auc-vlsi-45-lnx
  No distributed processing
Command Option Settings Summary
  -floorplan -effort medium
Info: starting to push down row track for sub-blocks.
Information: All hard macros are fixed, no hard macro placement is done. (DPP-416)
Generating automatic soft blockages for i2c_master_top, hor/vert channel sizes are 11.52/11.52
Placing top level std cells.
coarse place 0% done.
coarse place 33% done.
coarse place 67% done.
coarse place 100% done.
Running block placement.
Floorplan placement done.
****************************************
Report : report_placement
Design : i2c_master_top
Version: V-2023.12
Date   : Tue May  7 16:01:10 2024
****************************************
  ------------------
  Note: Ignoring violations of fixed cells or between fixed pairs of cells.
        To include violations of / between fixed cells, disable -ignore_fixed.
```



1.4 powerplanning & placement

All analysis and optimization done in logic synthesis and PnR is based on the fact that cells are supplied with **ideal voltage**. However, If the actual circuit supply is significantly different, cell operation will be **different** from the **behavior characterized in the .libs**, which will cause all types of timing Violations.

So what is power planning?

Power planning is deciding how we will deliver power to the design's standard cells and macros ,through connect our pins VSS,VDD to power and ground ,after that we will create power ring and mesh to can take direct supply from the nearest point around design

Why create mesh kind of structure?

- To distribute the Power from power pads/pins to all elements of the chip.
- Provides multiple paths from PG sources to destinations (less series resistance)
- Uniformly distribute power with **less voltage drop**.
- To meet IR/EM targets
- For meeting timing requirements

Power Grid Planning

Steps in Power Grid Planning:

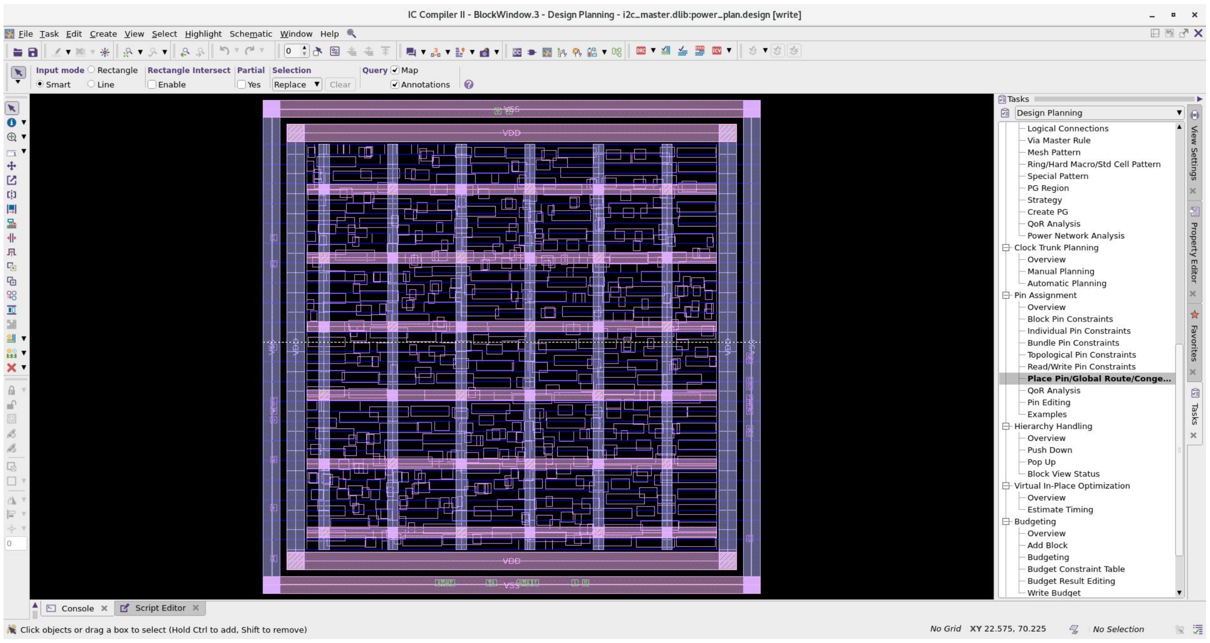
- Determine the number of power pads
- Determine which metal layers will be used for power routing

- Define the width of the top-level power bus
- Determine the structure for block and macro-level power routing
- Finally, Power Network Analysis (PNA)

Note: It prefers to create a copy of file.dlib and complete steps for Pnr and CTA to avoid corruption of source file

Challenge: we we need to create a VDD/VSS pins to get source from it , it still tells that it's created before because it's exist as ports for top module ,so it was the solution

```
create_net -power VDD1
create_net -ground VSS1
connect_pg_net -net VDD1 [get_pins -hierarchical "*/VDD"]
connect_pg_net -net VSS1 [get_pins -hierarchical "*/VSS"]
```


IC Compiler II - BlockWindow.3 - Design Planning - i2c_master.dlib:power_plan.design [write]

Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)

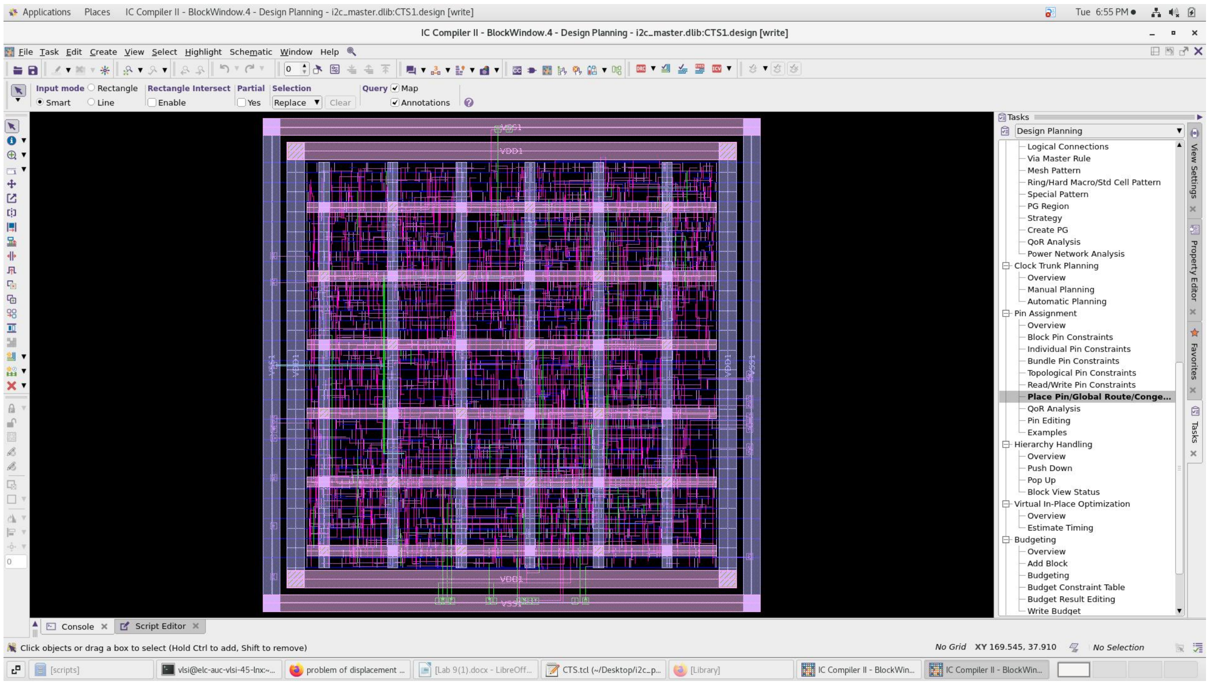
No Grid XY 22.575, 70.225 No Selection

1.5 Placement & CTA

It Builds an initial load-balanced, **DRC-clean clock tree,through determine the best path for clock which we can apply .**

It's goal is to meet the clock tree Design Rule Constraints (DRC):

- Maximum transition delay
- Maximum load capacitance
- Maximum fanout (2000, by default)
- Maximum buffer levels (50, by default)



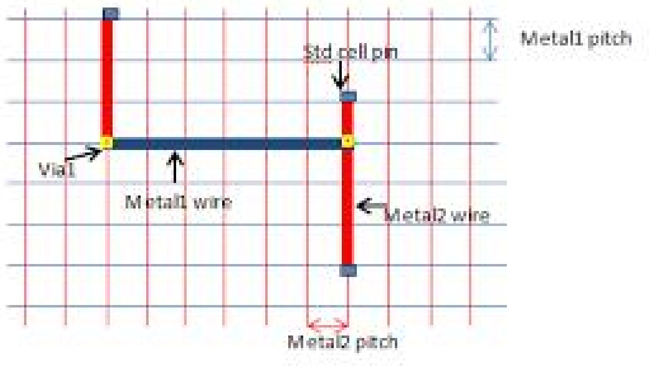
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)

1.6 Routing & signoff

Routing is the stage after Clock Tree Synthesis and optimization where-

- Exact paths for the interconnection of standard cells and macros and I/O pins are determined.
- Electrical connections using metals and views are created in the layout, defined by the logical connections present in the netlist.

The routing of nets that require special attention, clock and power nets, for example, is normally done before detailed routing of signal nets. The architecture and structure of these nets are performed as part of floorplanning, but the sizing and topology of these nets are finalized as part of the routing step.



Goals of Routing

- Minimize the total wire length.
- Meeting the timing DRC's.
- Minimize the number via's.
- No Design Rules Check (DRC) violations.
- No LVS errors.
- Minimum routing area.
- Minimizing the congestion hotspots.

Inputs for routing

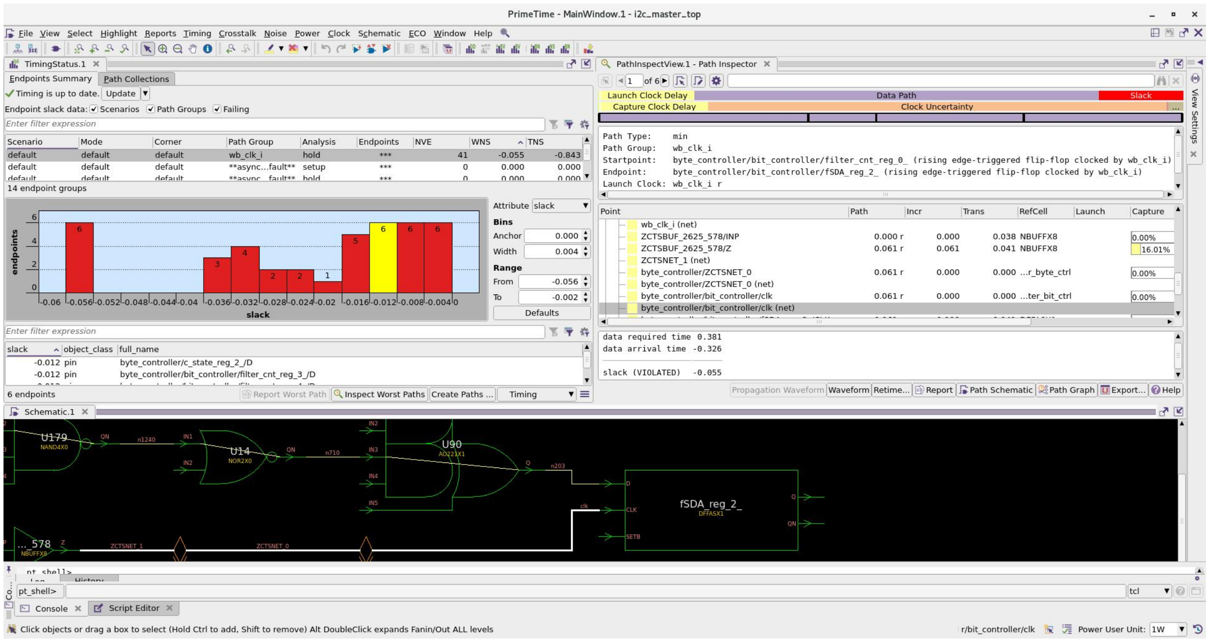
- Technology file.
- TLU+ file.
- SDC file.
- Design with complete CTS.

Output of routing

Design with completed interconnection and geometric layout of the nets.

After we finished the routing , w start to do prime Tim analyze to check hold/setup violation after add parasitics.

Challenges which we face it was error when we try to run primeTimeSetup.tcl , because the i2c_top_master is not viewed , so it solved by moving it manually in its file



Scenario	Mode	Corner	Path Group	Analysis	Endpoints	NVE	WNS	TNS
default	default	default	wb_clk_i	hold	***	41	-0.055	-0.843
default	default	default	**async...fault**	setup	***	0	0.000	0.000
default	default	default	**async...fault**	hold	***	0	0.000	0.000

14 endpoint groups

slack	object_class	full_name
-0.012	pin	byte_controller/c_state_reg_2_/D
-0.012	pin	byte_controller/bit_controller/filter_cnt_reg_3_/D

6 endpoints

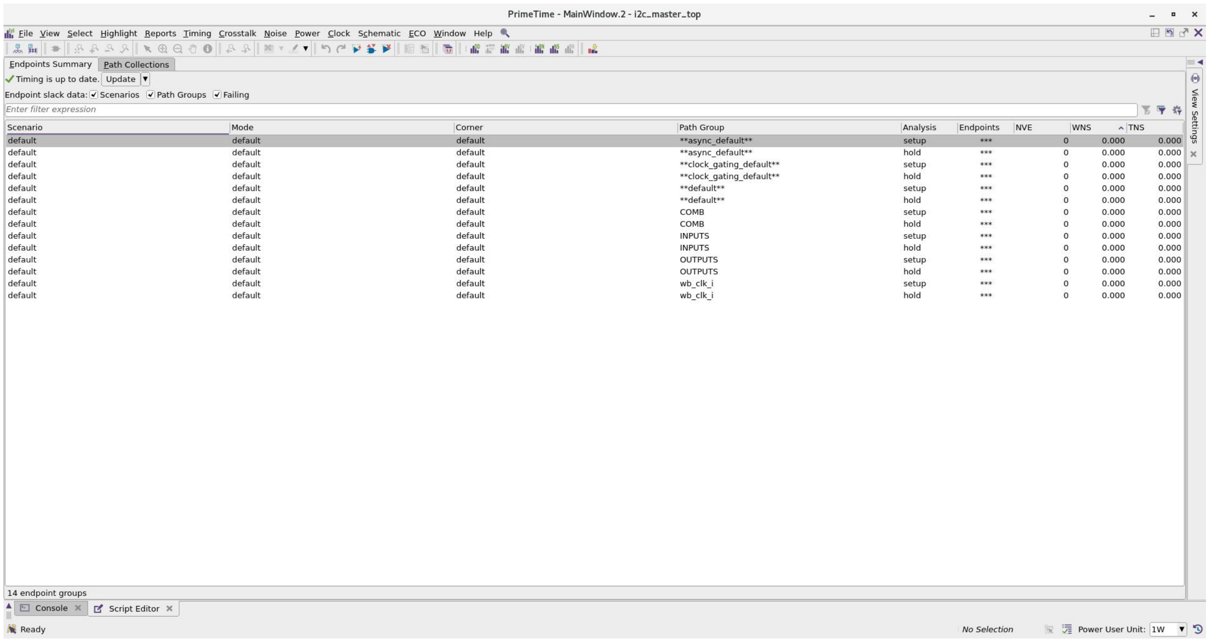
```
Path Type:     min
Path Group:    wb_clk_i
Startpoint:    byte_controller/bit_controller/filter_cnt_reg_0_ (rising edge-triggered flip-flop clocked by wb_clk_i)
Endpoint:      byte_controller/bit_controller/fSDA_reg_2_ (rising edge-triggered flip-flop clocked by wb_clk_i)
Launch Clock:  wb_clk_i r
```

Point	Path	Incr	Trans	RefCell	Launch	Capture
wb_clk_i (net)						
ZCTSBUF_2625_578/INP	0.000 r	0.000	0.038	NBUFFX8		0.00%
ZCTSBUF_2625_578/Z	0.061 r	0.061	0.041	NBUFFX8		16.01%
ZCTSNET_1 (net)						
byte_controller/ZCTSNET_0	0.061 r	0.000	0.000	...r_byte_ctrl		0.00%
byte_controller/ZCTSNET_0 (net)						
byte_controller/bit_controller/clk	0.061 r	0.000	0.000	...ter_bit_ctrl		0.00%
byte_controller/bit_controller/clk (net)						

```
data required time  0.381
data arrival time  -0.326
-------------------------
slack (VIOLATED)   -0.055
```

From image it shows w have 41 hold violation , so we can solve it by insert buffers or use this commend

```
fix_eco_timing -type hold -methods insert_buffer -buffer_list {NBUFFX1 NBUFFX2 NBUFFX3}
```


PrimeTime - MainWindow.2 - i2c_master_top

Endpoints Summary | Path Collections

Timing is up to date. Update

Endpoint slack data: ✓ Scenarios ✓ Path Groups ✓ Failing

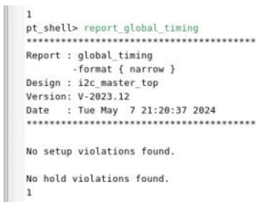
Scenario	Mode	Corner	Path Group	Analysis	Endpoints	NVE	WNS	TNS
default	default	default	**async_default**	setup	***	0	0.000	0.000
default	default	default	**async_default**	hold	***	0	0.000	0.000
default	default	default	**clock_gating_default**	setup	***	0	0.000	0.000
default	default	default	**clock_gating_default**	hold	***	0	0.000	0.000
default	default	default	**default**	setup	***	0	0.000	0.000
default	default	default	**default**	hold	***	0	0.000	0.000
default	default	default	COMB	setup	***	0	0.000	0.000
default	default	default	COMB	hold	***	0	0.000	0.000
default	default	default	INPUTS	setup	***	0	0.000	0.000
default	default	default	INPUTS	hold	***	0	0.000	0.000
default	default	default	OUTPUTS	setup	***	0	0.000	0.000
default	default	default	OUTPUTS	hold	***	0	0.000	0.000
default	default	default	wb_clk_i	setup	***	0	0.000	0.000
default	default	default	wb_clk_i	hold	***	0	0.000	0.000

14 endpoint groups

Ready No Selection Power User Unit: 1W

1.7 A look at reportsts :)

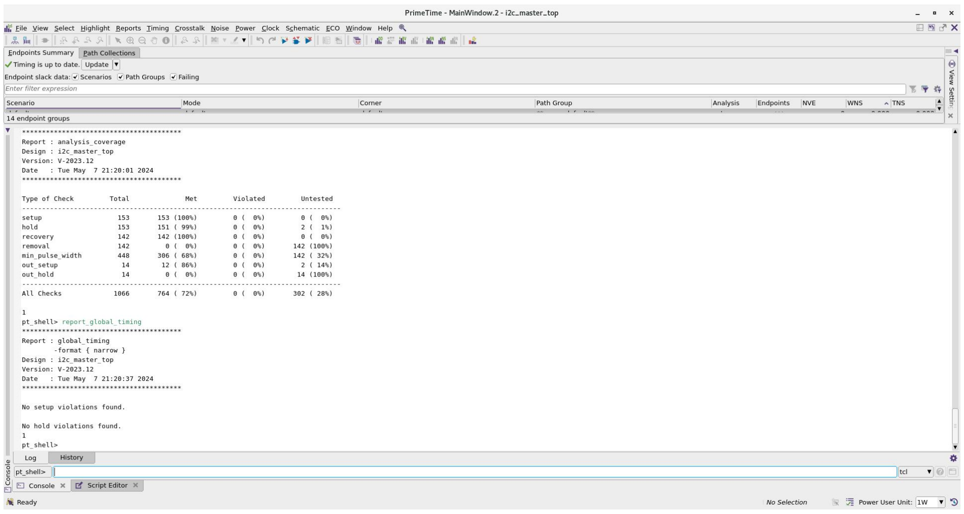
Report global timing and discover if any violation


```
1
pt_shell> report_global_timing
****************************************
Report : global_timing
        -format { narrow }
Design : i2c_master_top
Version: V-2023.12
Date   : Tue May  7 21:20:37 2024
****************************************

No setup violations found.

No hold violations found.
1
```

Report coverage analysis


PrimeTime - MainWindow.2 - i2c_master_top

Endpoints Summary | Path Collections

Timing is up to date. Update

Endpoint slack data: ✓ Scenarios ✓ Path Groups ✓ Failing

14 endpoint groups

```
****************************************
Report : analysis_coverage
Design : i2c_master_top
Version: V-2023.12
Date   : Tue May  7 21:20:01 2024
****************************************

Type of Check         Total             Met         Violated         Untested
--------------------------------------------------------------------------------
setup                   153       153 (100%)         0 (   0%)         0 (   0%)
hold                    153       151 ( 99%)         0 (   0%)         2 (   1%)
recovery                142       142 (100%)         0 (   0%)         0 (   0%)
removal                 142         0 (   0%)         0 (   0%)       142 (100%)
min_pulse_width         448       306 ( 68%)         0 (   0%)       142 ( 32%)
out_setup                14        12 ( 86%)         0 (   0%)         2 ( 14%)
out_hold                 14         0 (   0%)         0 (   0%)        14 (100%)
--------------------------------------------------------------------------------
All Checks             1066       764 ( 72%)         0 (   0%)       302 ( 28%)

1
pt_shell> report_global_timing
****************************************
Report : global_timing
        -format { narrow }
Design : i2c_master_top
Version: V-2023.12
Date   : Tue May  7 21:20:37 2024
****************************************

No setup violations found.

No hold violations found.
1
pt_shell>
```

Ready No Selection Power User Unit: 1W

1.8 Conclusion

Our journey through the ASIC flow illuminated the meticulous steps involved in transforming a high-level design into a physical chip. From RTL design to synthesis, and from floorplanning to routing and signoff, each phase was pivotal in achieving an optimal design that meets stringent power, area, and timing constraints.

In conclusion, the I2C protocol's design and its implementation through the ASIC flow exemplify the synergy between communication protocols and digital design methodologies. This harmonious integration is crucial for the development of efficient, high-performance integrated circuits in the modern era of electronics.