**Programmable Interval Timer (PIT)**

**ASIC Diploma**

**Najah Now**

**Introduction**

**About PIT**

The Programmable Interval Timer Module, PIT, is a simple timer to generate a periodic signal for a microcontroller system. This signal may be used for a variety of purposes such as triggering the start of an Analog to Digital or Digital to Analog conversion, as a periodic system interrupt, or to synchronize the start of various other hardware processes.

The PIT core has 4 primary blocks:

* WISHBONE Interface
* Control Registers
* Prescale Counter
* Main Counter

A diagram of a computer system

Description automatically generated

**PIT structure**

**Clocks**

**We have only a master clock for the whole system named wb\_clk\_i with max frequency = 200MHz so its period = 1/200\*10^6 = 5ns.**

A white and black chart

Description automatically generated

**Tools**

Synopsys tools were used to implement this project such as:

Design Compiler – ICC2 Compiler – Prime Time

**== == == == == == == == == == == == ==**

**Synthesis**

Synthesis was performed using Design Compiler tool.

We used saed32nm library to implement the project.

1. **Design libraries**

link and target libraries were set with min and max libraries to perform the correct timing analysis.

We used:

Min corner: ff1p16vn40c for Hold timing analysis

Max corner: ss0p75v125c for Setup timing analysis

1. **Design Constraints**

* Clock definition with clock period = 5ns
* Input delay = 0.2 \* clock\_period
* Output delay = 0.4 \* clock\_period
* wire load model “35000”
* operating conditions

“ff1p16vn40c” for min corner and “ss0p75v125c” for max corner

* input transition (using “AND2X1\_HVT” from max corner to derive the input)
* output load = 1 ff

1. **compile stage**

We compiled the design using compile\_ultra command and some app options to optimize the design.

1. **Generating design files**

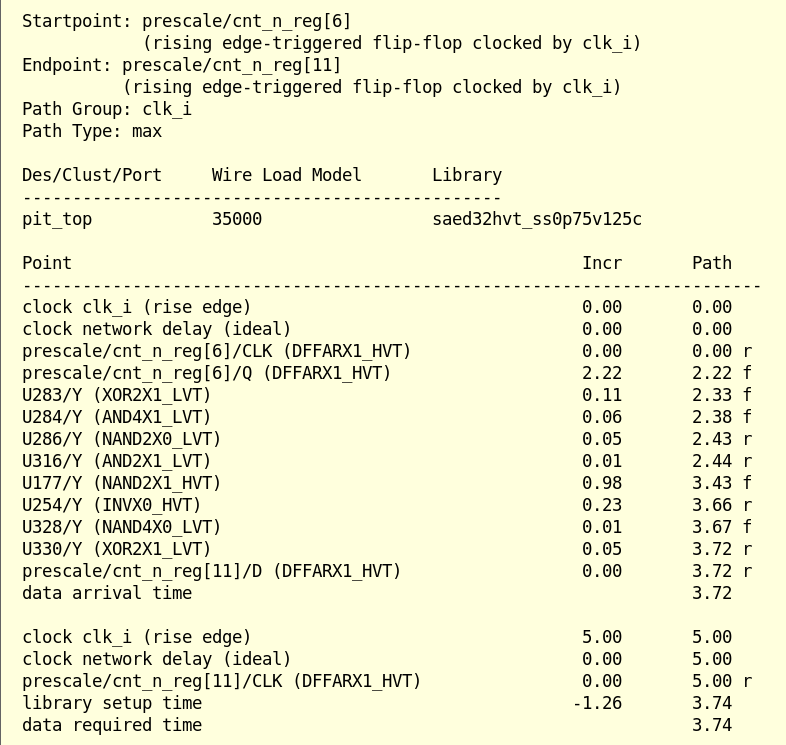
* SDC file
* Design netlist

*Synthesis results summery*

|  |  |
| --- | --- |
| **Report** | **Result** |
| **Setup Time** | **0.02 (MET)** |
| **Hold Time** | **0.05 (MET)** |
| **Area** | **1311.83** |
| **Power** | **6.8310e+03 uW** |

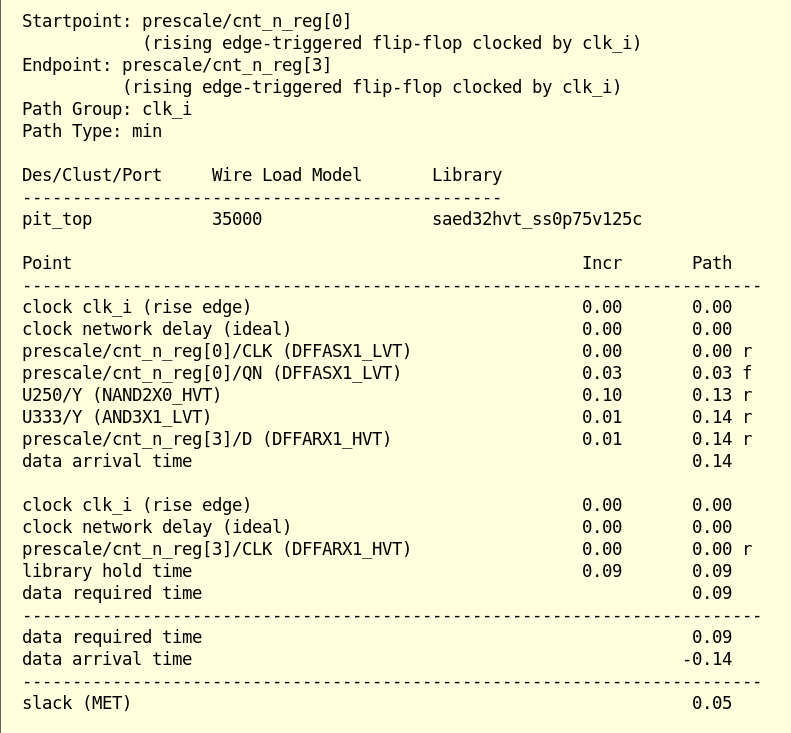
1. **reports**

**setup Report**

**A white rectangular object with black lines

Description automatically generated**

**Hold Report**

****

**Area Report**

**A screenshot of a computer

Description automatically generated**

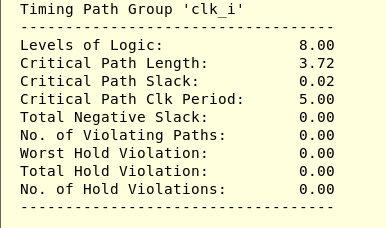
**Power Report**

**A screenshot of a computer program

Description automatically generatedA close-up of a receipt

Description automatically generated**

**QOR Report**

**A screenshot of a cell count

Description automatically generated**

**A close up of a number

Description automatically generatedA screenshot of a computer

Description automatically generated**

**PnR**

PnR stage was done using ICC2 tool from Synopsys.

In this stage we performed all PnR stages such as: floorplanning, Powerplanning, Placement, CTS, Routing.

First stage was data setup to set the libraries, ndm, technology file, read parasitics tluplus files, read design netlist and source design constraints.

* Floorplanning
* In this stage we defined routing tracks (M1 to M9) directions and placed the pins arround the rectangular core.
* Utilization was 40%

**A black square with pink squares

Description automatically generated**

* Powerplanning

We used M8 metal layer for vertical mesh and M9 for the horizontal mesh.

A grid with purple and pink squares

Description automatically generated

* Placement

MMMC file was sourced to set the needed files and corners for timing checking.

A grid with purple squares

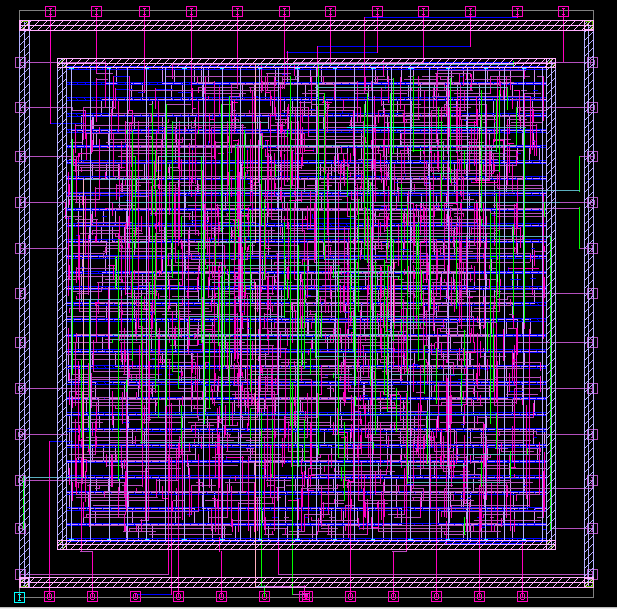
Description automatically generated

* CTS
* Clock network was built on metal layers M5, M6, M7.
* Trget skew was setted by 0.2.
* Clock latency was computed by the tool.

A grid with lines and dots

Description automatically generated with medium confidence

* Routing

****

*PnR results summery*

|  |  |
| --- | --- |
| **Report** | **Result** |
| **Setup Time** | **0.00 (MET)** |
| **Hold Time** | **-0.44 (VIOLATED)** |
| **Area** | **2264.17** |
| **Power** | **2.59e+08 pW** |

**QOR Report**

**A screenshot of a cell count

Description automatically generatedA screenshot of a computer screen

Description automatically generatedTiming Area**

**A white background with black lines

Description automatically generatedDRC**

**Congestion Report**

A close-up of a number

Description automatically generated

**Prime Time Session**

First, we set up the libraries and important files such as: design netlist, parasitics file(.spef), constraints file.

Then we updated the timing, power, and noise info to start checking timing, power, and noise.

After this we performed an ECO fixing Session to fix timing and DRC violations.

After the ECO fixing, we get these results on PrimeTime tool

*Results summery*

|  |  |
| --- | --- |
| **Report** | **Result** |
| **Setup Time** | **0.265 (MET)** |
| **Hold Time** | **0.002 (MET)** |

**A screenshot of a computer

Description automatically generated**

**Costraints**

**Setup A screenshot of a computer program

Description automatically generated**

**Hold**

**A screenshot of a computer program

Description automatically generated** **A black lines on a white background

Description automatically generated**