

|  |
| --- |
| **DPLL – Time to Digital Converter**  **Digital Phase Locked Loop in cooperation with Intel** |

|  |  |
| --- | --- |
| **24-1-1-3099** | **Project Number:** |

|  |
| --- |
| **Project Report** |

|  |  |  |  |
| --- | --- | --- | --- |
| Student: | Shay Dotan | ID: | 205705577 |
| Student: | Amin Yasin | ID: | 211825773 |

|  |  |
| --- | --- |
| Supervisor: | Rotem Banin / Ofir Degani |

|  |  |
| --- | --- |
| Project Carried Out at: | Tel-Aviv University |

Contents

[Abstract……….. 2](#_Toc135313526)

[1 Introduction 3](#_Toc135313527)

[2 Theoretical background 4](#_Toc135313528)

[3 Simulation.. 5](#_Toc135313529)

[4 Implementation 6](#_Toc135313530)

[4.1 Hardware Description 6](#_Toc135313531)

[4.2 Software Description 7](#_Toc135313532)

[5 Analysis of results 8](#_Toc135313533)

[6 Conclusions and further work 9](#_Toc135313534)

[7 Project Documentation 10](#_Toc135313535)

[8 References.. 11](#_Toc135313536)

[Appendix A. Subject 1 12](#_Toc135313537)

[Appendix B Subject 2 13](#_Toc135313538)

**List of figures**

[Figure 1: Block diagram 2](#_Toc135313863)

[Figure 2: The System 6](#_Toc135313864)

[Figure 3: Amplifier Input Circuit 6](#_Toc135313865)

**List of tables**

[Table 1: Results 8](#_Toc135313943)

**List of equations**

Abstract

This project focuses on the design and tapeout of a Digital Phase-Locked Loop (DPLL), a system used to align the phase of a generated clock with a stable reference clock. The DPLL achieves synchronization by detecting and correcting phase differences between the two signals.

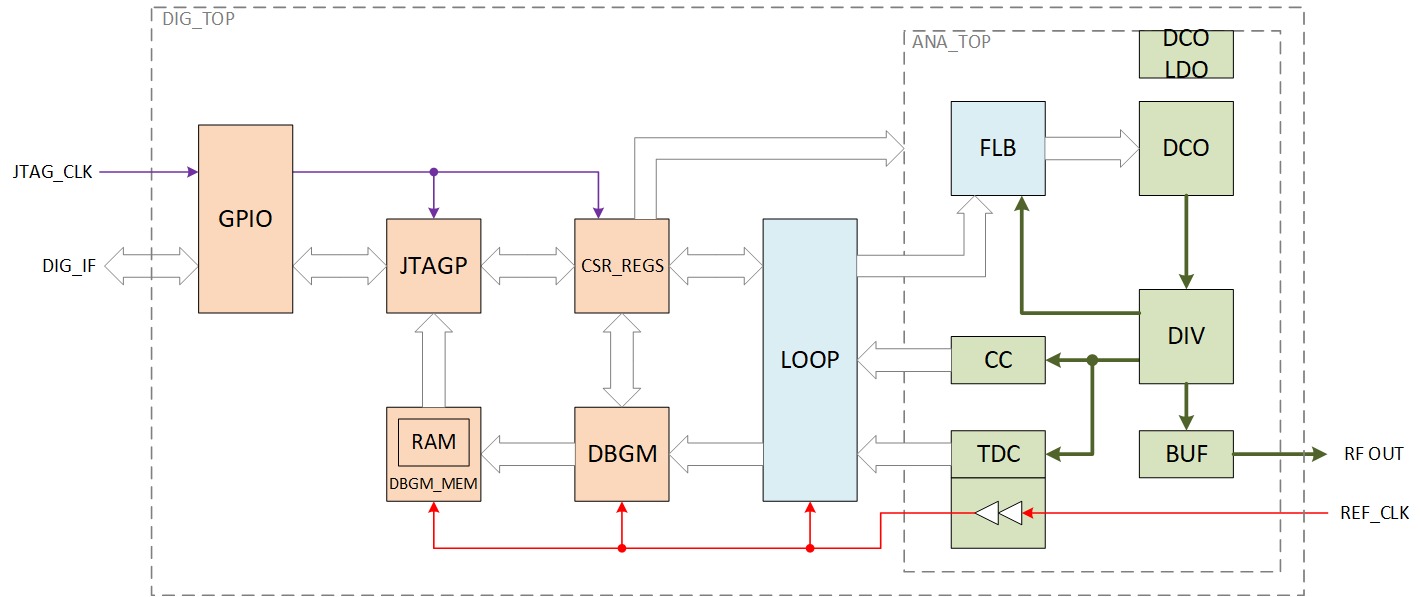


Figure 1 DPLL Block Diagram

The work was done in teams of two, with each team assigned a specific block in the system. Our team was responsible for the analog design of the Time-to-Digital Converter (TDC), a critical component that measures the phase difference between the reference clock and the generated clock. This digital value is later used by other components in the loop to correct the clock phase and maintain alignment.

The project involved designing the schematic and layout of the TDC according to strict specifications, resulting in a layout ready for tapeout and integration into the complete DPLL system.

Table 1 Specs definition

|  |  |
| --- | --- |
| Specification |  |
| Quantization noise | RMS 5ps |
| Output range | Min 200ps after PVT |
| Voltage Supply | 0.9V |
| Meta-Stability | Spectral density Jitter 6dB below Quantization noise |
| Bubbles | No bubbles or holes in the output (3 sigma) |
| Power | Show consideration in the design |

# How is the Design Process of Microchips: Analog IC Design Flow to Tapeout - Mis CircuitosIntroduction

Figure 2 Flowchart of project

The goal of this project is not only to **design a Time-to-Digital Converter** (TDC) for tapeout as part of a Digital Phase-Locked Loop (DPLL), but also to give students **practical experience** in the full cycle of a real-world engineering project. This includes defining specifications, planning, schematic design, simulations, layout creation, verification (DRC and LVS), and preparation for fabrication. The project acts as a capstone for our electrical engineering degree, integrating knowledge from a wide range of courses such as analog and digital circuit design, VLSI systems, signal processing, semiconductor physics, and more.

Beyond technical skills, the project emphasizes the importance of teamwork, communication, and problem-solving under realistic constraints. Working in teams of two, each group is assigned responsibility for a specific component within the DPLL system. This structure simulates industry workflows, where teams must coordinate with others to ensure compatibility and timing across modules.

Our team was assigned the analog design of the TDC. Through this process, we improved our ability to work collaboratively, share tasks effectively, and make joint decisions under pressure.

Another major objective is to expose students to tools and environments not fully covered in the classroom. For example, we used Cadence Virtuoso for design and layout—a tool that, prior to this project, many students had little or no experience with. This required us to learn independently, search for solutions in documentation, tutorials, or online forums, and apply new methods without direct instruction. The learning curve was steep but valuable, reinforcing our ability to adapt and pick up new technologies—an essential skill for any engineer.

The motivation behind the project lies in preparing us for real engineering work. In the industry, success depends not only on technical knowledge but also on the ability to deal with unknowns, communicate with peers, and complete projects from start to finish. This project simulates those demands, giving us a controlled yet realistic environment to develop those skills.

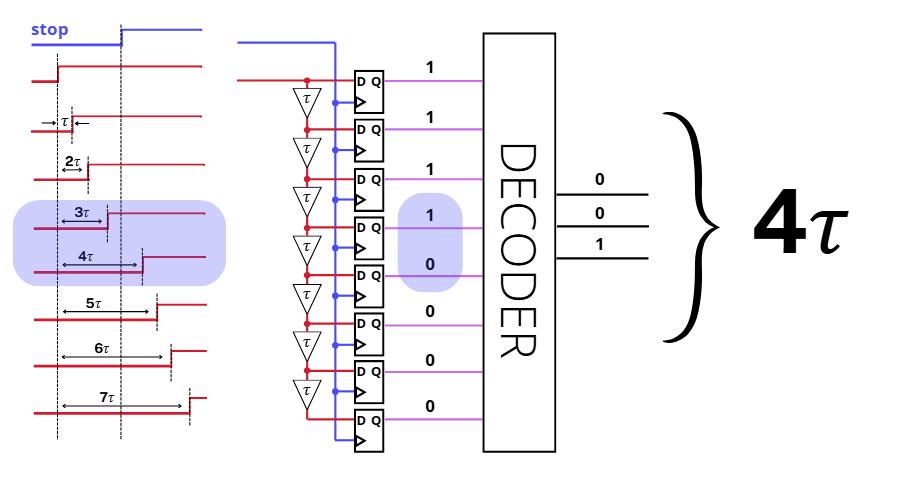
When problems arose—such as tool errors, unexpected circuit behavior, or design rule violations—we explored multiple ways to solve them. This included consulting other teams who had already faced similar issues, searching online for application notes and examples, and getting guidance from our academic mentor. These situations taught us how to troubleshoot efficiently and when to seek help versus when to keep experimenting.

Through this comprehensive process, we gained technical depth, project experience, and practical skills that cannot be fully taught in lectures. The result is not only a working circuit but a deeper understanding of what it means to be an engineer.

# Theoretical background

Most of the theoretical background is referenced from "Introduction to Design of Digital Radio Circuits for Communication," [1].

## What’s Flash TDC

A diagram of a decoder

AI-generated content may be incorrect.

Figure 3 Simple flash TDC (left), Simple Vernier TDC (right)

A flash TDC uses a single delay chain, parallel delay lines, or a vernier delay line to generate equally spaced delay times, denoted as . Each delayed start signal is input to a flip-flop and latched at the rising edge of the delayed stop signal to generate a zero-to-one transition. Its time resolution is determined by a single gate delay Τ, or a differential gate delay , which is limited by the capability of sizing the transistors in the vernier delay line.

## Noise

Noise in a Vernier Time-to-Digital Converter (TDC) primarily stems from the inherent jitter, and the quantization noise introduced by the discrete nature of the measurement.

### Quantization noise

The quantization noise in a digital system typically follows a uniform probability density function (PDF), especially when the quantization error is considered a random variable. This means the noise is equally likely to fall anywhere within the quantization interval.

0

Figure 4 PDF of quantization noise

So we can say that the RMS value of the quantization noise is :

Where is the mean value of the delay between the Clock and Data signals.

### Jitter

We use the Phase Noise, considering Ideal signal (a) in frequency domain, and the actual signal (b), we chose the noise to be phase noise, the time dependence is on the phase -

Figure 5 Phase noise example

From this mathematical represnetion we can define the Phase Noise

If we look at the signal’s spectrum and isolate a 1 Hz slice, we can ask how much energy it has compared to the carrier’s energy. This gives the phase noise density, measured in [dBc/Hz]. ω₀ represents the ideal frequency (e.g., 40 MHz). The integral of the offset (integrated phase noise) is the total noise summed relative to the signal’s energy. This is the SNR, the total energy of the noise in relation to the total energy of the carrier. Furthermore, the integral of the phase noise in the frequency domain equals the integral in the time domain due to Parseval’s law. **The phase noise in the frequency domain has a direct relationship to the Jitter in the time domain.**

To measure the Jitter, we use JEE – Jitter Edge to Edge, which is Phase Jitter. To convert the Phase noise to jitter we use the following formula, using the integral on the phase noise and the ideal frequency of our signal.

### Comparison

To compare jitter noise with quantization noise, we simply calculate their ratio using:

## Bubble

A bubble at the output of a TDC (Time-to-Digital Converter) is a wrong bit flip inside the output code, such as a ‘0’ appearing in the middle of ‘1’s or the opposite. This happens due to timing mismatches, metastability, clock skew, or setup and hold violations inside the flip-flops. Bubbles are problematic because they break the expected monotonic code sequence, causing incorrect time measurements. To fix this, designers often use bubble correction logic or improve the timing and layout of the TDC circuitry.

To check for bubbles we run transiet simulation with a verilog-A decoder and make sure that the output is non-decreasing. Furthermore we check the delays between the elements for 3 points (start,middle,end) with montecarlo and ensure that there isn’t negative value (clock/data delay is flipped)

11111110000000000011111111 – normal expected operation

11111010000000000011111111– rising edge bubble

00000101111111111000000000 – falling edge bubble

## Meta-stability

Meta-stability is measured inside the FF, considering Main-Secondary –

B

A

Figure 6 Main-Secondary FF

main

secondary

The A-B loop is closed, and it takes some time for nodes A and B to settle to stable digital values. This period is called the metastability phase. This A-B loop can be characterized as follows:

R

C

-A

VA

VB

R

C

-A

Figure 7 A-B Loop characterized

We can solve the differential current equations for both sides and get:

If is the voltage difference for which the circuit exists metastability,

We want to find – the time that takes the circuit to exit metastability.

Assuming is distributed uniformly in , we consider the time to exit metastability as random variable distributed exponentially with distribution parameter.

The average and STD of this distribution is and the probability that it will remain in this state at is . So the probability to exit metastability is . We’ll force metastability with transient simulation, measure the time that the circuit takes to stabilize, take ln() of the voltage difference, and find where is the slope is linear () by derive the entire solution. This will give us .

Using this parameter, we can calculate the success rate of the TDC failing metastability. Given the clock rate Hz, we can calculate the times of entering metastability state in one day:

We want the probability that the FF will remain in this state after , the probability is:

*And the probability that it will not fail during the whole day is*

# Simulation

## Output (With ideal Decoder)

Using a Verilog A decoder with the TDC output, we observe the behavior. We simulate with a phase ramp to detect bubbles and unexpected behaviors.





Print 1 TDC output with Decoder output

The transient signal shows the TDC output, while the decoder output is shown as bits translated into an unsigned integer.

## Monte Carlo

Using Monte Carlo sampling option, we can measure the delay between elements with mismatch. We chose to measure the delay along the start, the middle and the end.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| stage | Min [PS] | Max [PS] | Mid [PS] | std |
| 6 | 2.685 | 4.641 | 3.574 | 345f |
| 13 | 2.946 | 4.669 | 3.695 | 350.9f |
| 16 | 2.774 | 4.773 | 3.714 | 341.4f |
| 21 | 2.605 | 4.842 | 3.707 | 351.7f |
| 29 | 2.709 | 4.549 | 3.579 | 341.5f |

Table 2 Monte Carlo simulation summary for delay between stages

A graph with red dots

AI-generated content may be incorrect.

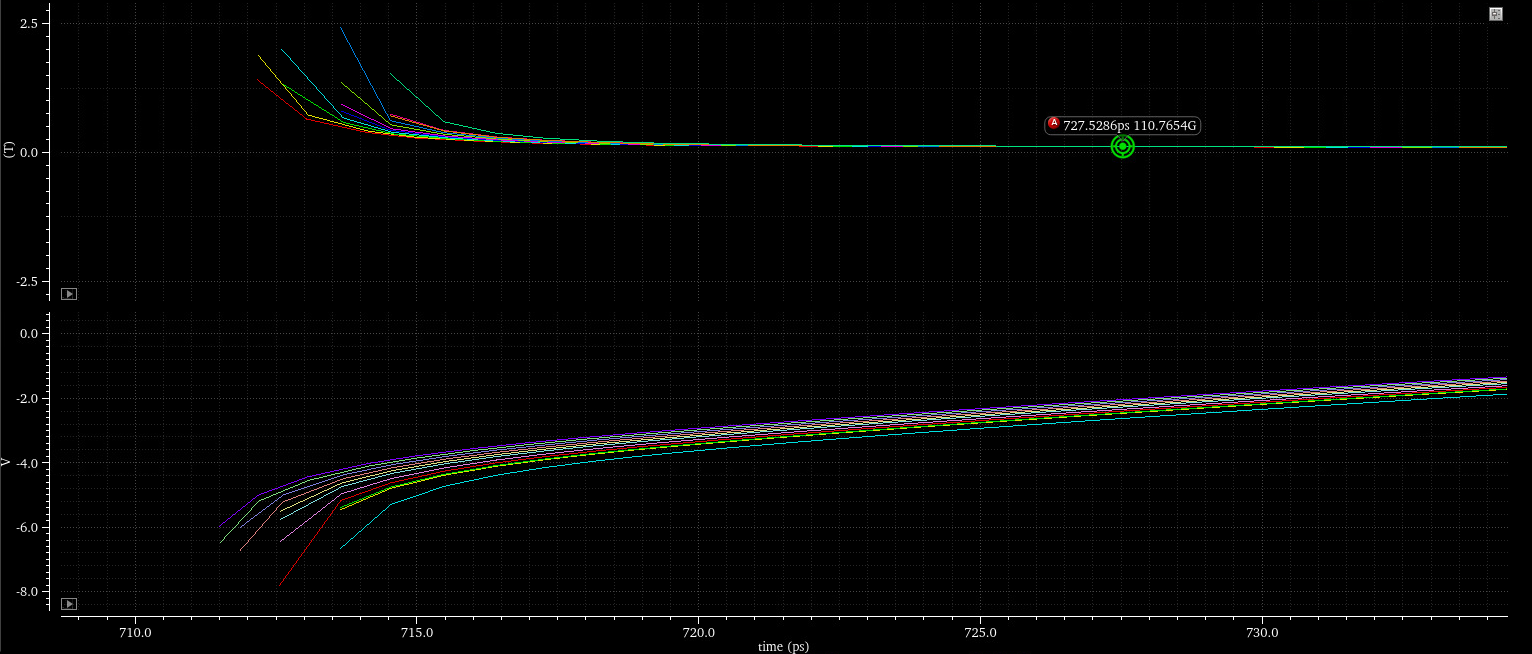
Print 2 Delay distribution for stage 14

## Metastability

A graph with green lines

AI-generated content may be incorrect.

Print 3 Meta-Stability inside FF on stage 14, top is positive and bottom is negative



Print 4 ln (top) and deriv of ln (bottom) of the diffrential signal inside A-B loop of diffrential FF

## Power

## Transient

A computer screen shot of a computer screen

AI-generated content may be incorrect.

Print 5 Example of Data delay of stage 14 inside TDC

A screenshot of a computer

AI-generated content may be incorrect.

Print 6 Example of Clock delay of stage 14 inside TDC

## Jitter

A screenshot of a computer

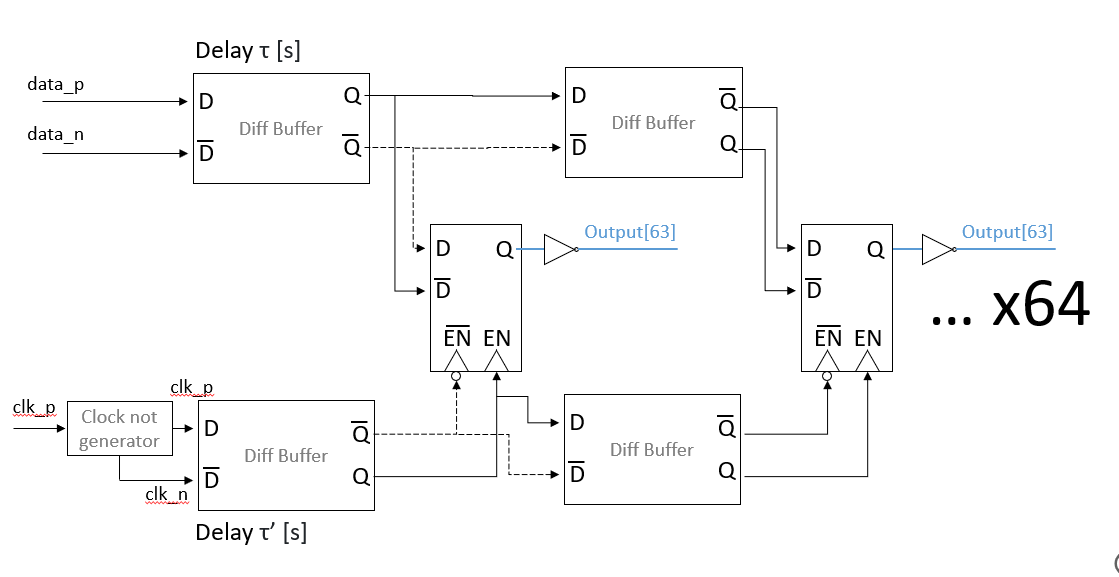
AI-generated content may be incorrect.

Print 7 Jitter simulation result

# Implementation

## TDC

Figure 8 TDC block diagram



Diff flip flop

Diff buffer

a

b

c

Using Vernier methodology, we can observe the following components:

1. Differential Buffer

Both the Clock and Data delay lines using differential buffers lines. They both are designed the same, with different sizes of transistors to make the differential delay as desired.

1. Differential Flip Flop

The Clock signal is fed to 64 differential flip flops via the EN input. The Data signal is fed to the data input, differentially. The output of the Flip Flop will be fed to an encoder designed by another team from Digital, so a buffer that separates between the Analog and Digital side is attached to the outputs.

1. Clock not generator

The clock signal is given to the TDC without differential phase, so a ‘not’ generator is created to be able to use the differential signal design.

## Integration and power consumption consideration

5

… x64

Output[1]

Diff flip flop

Diff buffer

In addition to TDC, we have components for chip integration and power consumption consideration.

Figure 9 Block digram of Add-ons to the TDC

64 stages TDC

Signal Enabler

Powe Saver

Buffer bus

1. Signal Enabler

The entire Analog section has en signal that will shut down the operation of the signals, for testing and power saving. Using AND gates with the en signal we can control the signals operation.

1. Power Saver

The TDC runs on a 5 GHz data signal and a 40 MHz clock, sampling only near the clock’s rising edge—at most one data cycle (200 ps). Meanwhile, the data keeps toggling, causing the flip-flops to switch and waste power. Using a Power saver component reduces power consumption during TDC operation.

1. Buffer bus

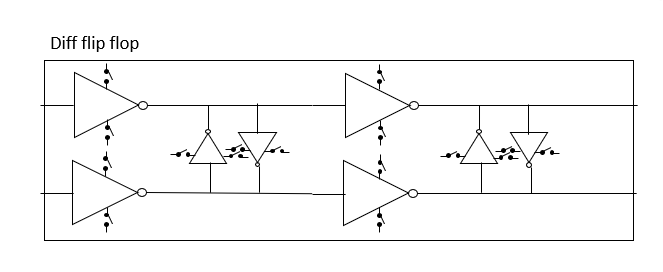
Using a buffer bus at the clock input ensures that the TDC clock is square-shaped, even if the system clock signal is sinusoidal.

## Hardware Description[[1]](#footnote-1)

All components use TSMC 30 nm MOSFET technology. Some are designed at the transistor level, while others use provided standard gates or cells.

### Differential Flip Flop

Figure 10 Differential FF block diagram

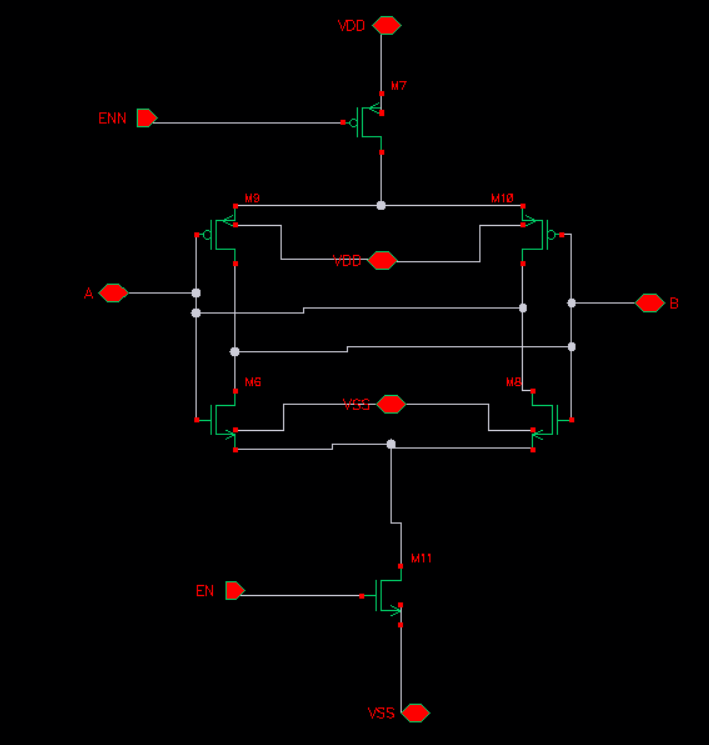
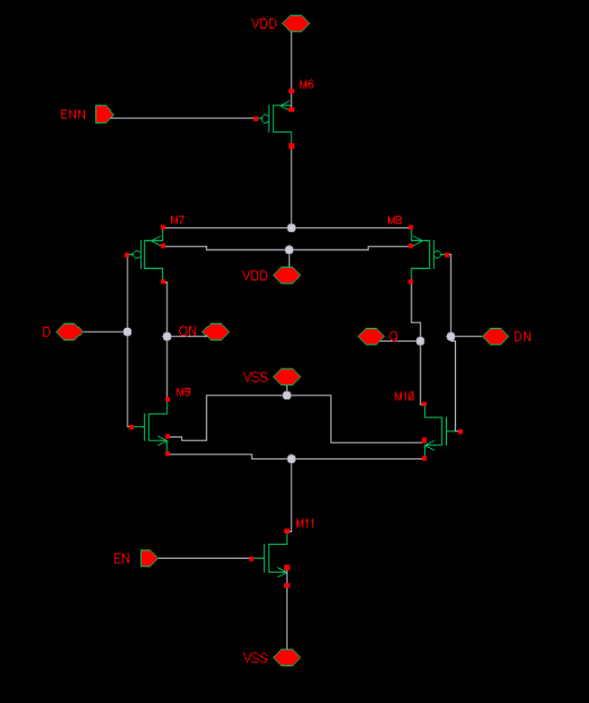


b

a

The Differential Flip Flop is made using two components 4 times as marked in (a) and (b). The first components is the differential inverter and the second components is back-to-back inverter to remove noise and insure strong ‘0’ and ‘1’ at output.

Print 8 Transistor level blocks of differential FF

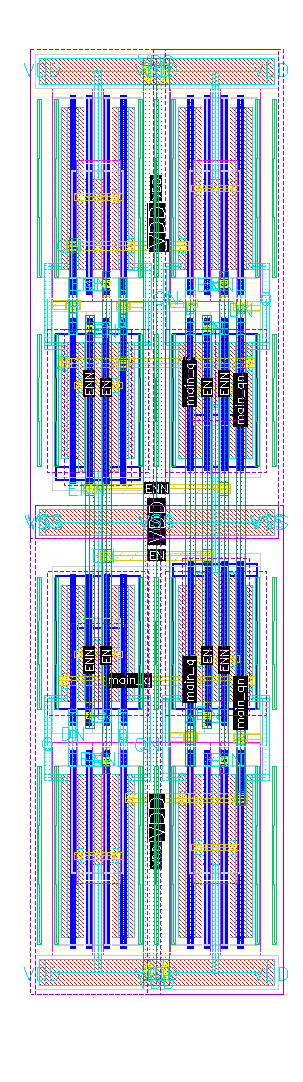


Defining the MOSFET that gets the EN/ENN signal as “ENN MOSFET” , and the other MOSFET that as “INVERTER MOSFET”, these are the sizes of the components :

|  |  |  |
| --- | --- | --- |
| Component | Width | Number of fingers |
| ENN PmOS | 1.4u | 2 |
| INVERTER PMOS | 1.26u | 1 |
| INVERTER NMOS | 900n | 1 |
| ENN NMOS | 1u | 2 |
| Fingers have the same width, 30nm technology | | |

Table 3 Transistors size of differential FF

Print 9 Layout of differential FF



a

b

b

a

Using multiple fingers for the MOSFET receiving the EN signal is planned before layout, to simplify connection to the matching NMOS and PMOS.

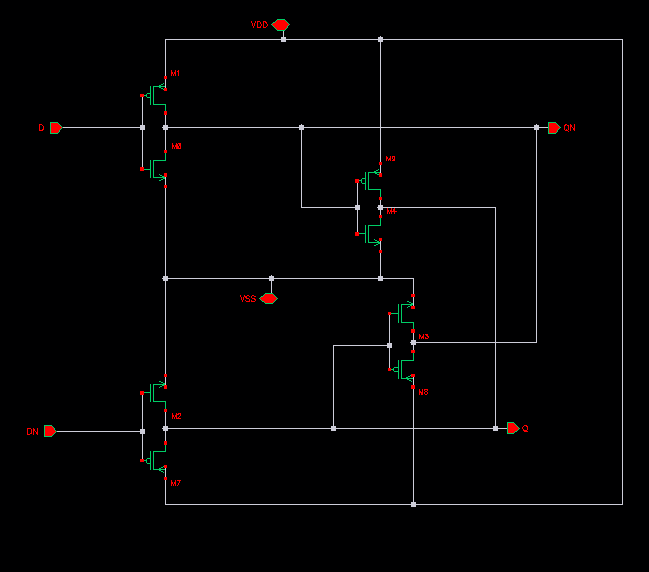
In the layout, two sets of (a), (b) are placed based on their block connections. On each level, (a) and (b) are aligned for easy linking. A cross pattern between the two levels is used to connect ENN/EN and to simplify connections between the two (a), (b) sets.

### Differential Delay

Diff buffer

Figure 11 Block diagram of Differential Delay unit

The differential buffer is a simple two not gates with cross-couples not gates to reduce noise and sharpen the output signal. The same setup is used for both the Clock and Data differential buffer, with different sized to introduce different delays.



Print 10 Transistor level of one differential delay unit

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Component | Data Width | Number of fingers DATA | Clock Width | Number of fingers CLOCK |
| Inverter pmos | 840n | 1 | 2.805u | 3 |
| INVERTER nmops | 600n | 1 | 1.955u | 3 |
| cross coupled pmos | 420n | 1 | 420n | 1 |
| cross coupled nmos | 300n | 1 | 300n | 1 |
| Fingers have the same width, 30nm technology | | | | |

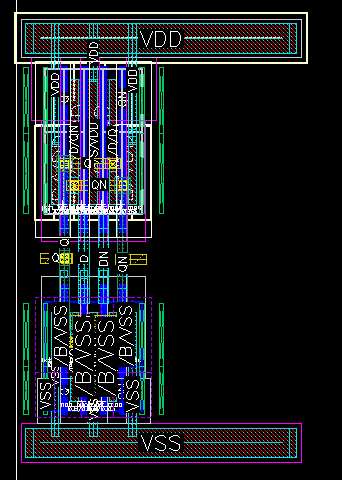
Table 4 Transistor sizes for Clock and Data delay elements

Using larger clock components creates a delay difference between the clock delay line and the data delay line, which is the core principle of Vernier TDC operation.

A computer screen shot of a computer

AI-generated content may be incorrect.

Print 11 Data delay element (left), Clock delay element (right)



The layout consideration took into account the differences in PMOS and NMOS, a simple top-down layout is used. The Power line size follows the biggest components in the TDC, which is the Flip Flop.

### TDC Element

Because of the cross-coupling nature of the TDC, where each delay element inverts the output and needed to flip the connection at the input of the next delay element, we chose to make an element using two Clock Delays, two Data Delays and two Flip Flops. Connecting 32 TDC elements will give us a TDC with 64 bit output.

After much consideration and testing a different floor placement routing, we chose to connect the TDC as follows:

Clock

Delay

FF

Data

Delay

Print 12 One unit of TDC Layout

Data delay line at the bottom, following Flip Flops at the middle, Clock delay line on top, and on top of the Clock delay lines – standard TSMC buffer with 4× drive strength for the connection between the Analog and Digital.

Each delay element has two clock inputs (clock, clock\_not), two data inputs (data, data\_not), two bit (q1,q2), two clock and two data outputs for the next element.

Figure 12 One unit TDC block diagram

delay

delay

Buffer

delay

Buffer

delay

FF

FF

… x64

data\_p

data\_n

clk\_p

clk\_n

data\_n

data\_p

clk\_n

clk\_p

data\_p

data\_n

clk\_p

clk\_n

q1

q2

Each TDC element is 3.64u in width and 14.23u in length.

### Full TDC

A grid with many colored lines

AI-generated content may be incorrect.

Print 13 Full TDC layout (trimmed)

The Full TDC is 116.48u in width, and 14.23u in length.

Each TDC element is connected to the next. With 32 elements, we achieve a 64-bit Vernier TDC. This TDC layout is used at the top level to connect the clock-not generator, integration, and handle power consumption considerations.

### Clock not generator

a

clk\_n

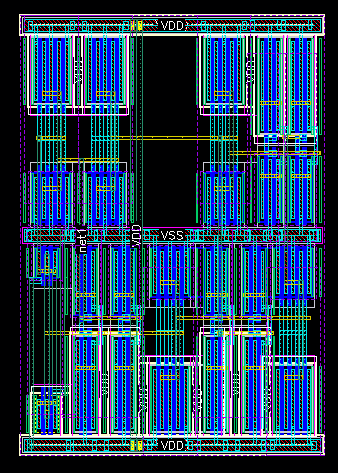
clk

clk\_p

b

Figure 13 Clock not generator block diagram

The clock-not generator is designed to take an input signal and produce both positive and negative phases, ensuring no phase shift so the TDC input clocks remain symmetrical. The design uses simple pass gate and inverters.



Print 14 Clock not generator Layout

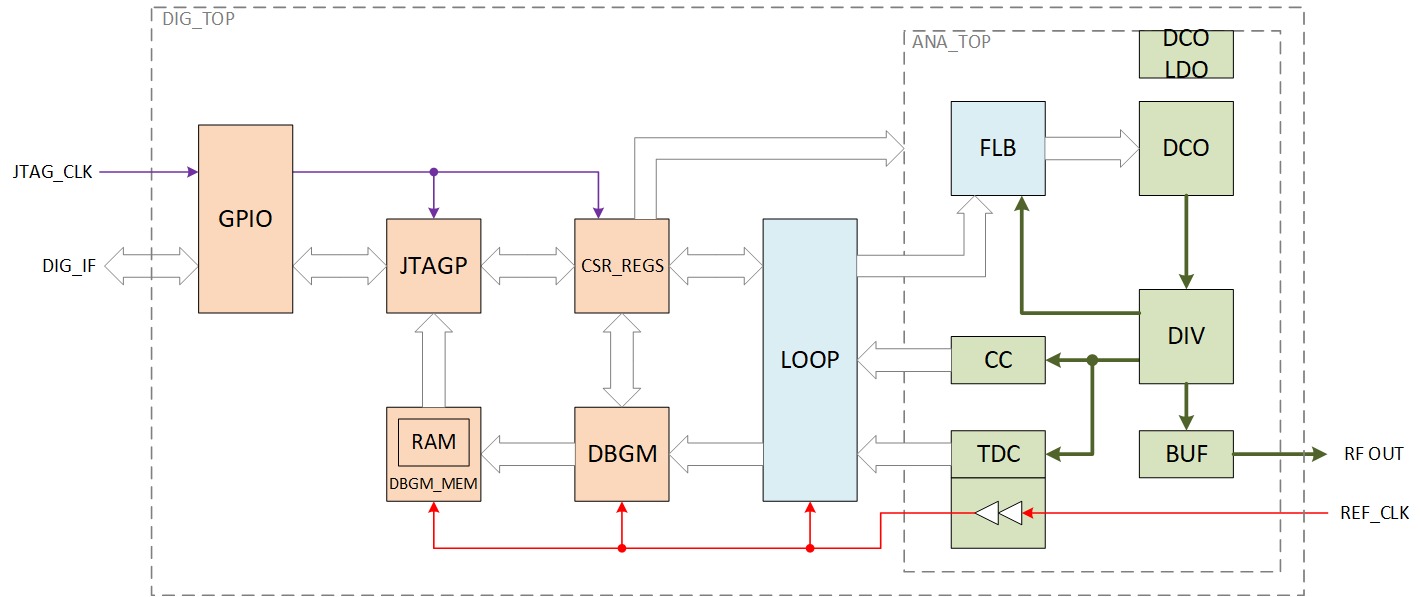
The top row is marked as (a) and the bottom row is marked as (b). The last back-to-back inverters are positioned on the top right corner for easy access to the input clock of the TDC.

b

a

|  |  |  |
| --- | --- | --- |
| Component | Width | N. of fingers |
| Inverter pmos | 4u | 4 |
| INVERTER nmops | 2.8u | 4 |
| B2B pmos | 3u | 2 |
| B2B NMOS | 2u | 2 |
| PASS GATE PMOS | 1u | 2 |
| PASS GATE NMOS | 750n | 2 |
| Fingers have the same width, 30nm technology | | |

### Buffer bus

We used a standard TSMC buffer to connect the TDC clock input, taking the buffer’s output as the clock. This way, a sinusoidal clock becomes a square signal

## Software Description

The project was implemented using Cadence Virtuoso, a professional software suited widely used for designing integrated circuits (ICs). It provides a complete environment for custom analog, digital, and mixed-signal design, covering all stages from schematic to layout and verification.

In the schematic editor, we built the circuit by connecting components such as transistors, resistors, capacitors, and current sources. This stage allows setting all electrical parameters and serves as the starting point for simulation.

The layout editor was then used to translate the schematic into the physical design. Here, we placed the geometric shapes that represent each component on silicon, added metal routing, and ensured proper connections between all elements. Special care was taken to meet design rule checks (DRC) and ensure the layout matched the schematic through layout versus schematic (LVS) verification.

For simulation, we used tools like ADE (Analog Design Environment) and Spectre to analyze the circuit’s behavior. We tested functionality, performance (such as speed, delay, and power), and ran checks under different conditions, including process corners and temperature variations.

Cadence Virtuoso’s integrated flow made it possible to move smoothly between schematic, layout, and simulation, ensuring reliable and efficient design work throughout the project.

# Analysis of results

|  |  |  |
| --- | --- | --- |
|  | Parameter | Real Time |
|  | Output range | 224ps |
|  | Meta-Stability | 99% Probability for success |
|  | Bubbles | No bubbles |
|  | Power | 9.1 mW |
|  | Quantization noise vs Jitter | Jitter more than 6dB below Q.noise |

Table 5 Results

From each simulation we can extract the data we need to see if we answer the spec given.

## Output range

Using Montecarlo simulation we can use the median to calculate the output range, which is , which is above the minimum required.

## Meta-Stability

Finding from the simulation we can calculate the % of success:

## Bubbles

Using the Ideal decoder we can see a monotony rise with the phase ramp, this with the Montecarlo simulation can conclude that there aren’t any bubbles in the design.

## Power

Power was calculated simply by taking the average current multiple by the power voltage. Shutting down the Data signal half of the clock period made the circuit use less current on switching and made the power consumption lower.

## Quantization noise vs Jitter

Simply by taking the Jitter from the simulation and using formula (4), we can see if we answer the specs:

Which means that the jitter noise is way lower than the Quantization noise.

# Conclusions and further work

The Time-to-Digital Converter (TDC) project achieved the main goals set at the start. The results from simulations and checks match the required specifications, showing that the design works as expected. However, there is still room to improve the power consumption, which will be addressed in the next design stages. Further testing is needed, especially when connecting the TDC to the top-level analog system. The project is not fully complete, as small adjustments may be required before tapeout, based on integration with other system parts. In the future, the focus will be on full system testing, optimizing performance, and preparing for tapeout, ensuring the design is ready for production and meets all required standards.

# Project Documentation

All project deliverables are to be documented (e.g. GitHub), this final project report should only include their description:

* Description of the project documentation (Hardware, software, additional simulations, user guide, etc.)
* The Documentation location (e.g. the GitHub link)
* Description of the project files

# References

[1] "Introduction to Design of Digital Radio Circuits for Communication," course, Tel Aviv University, coordinated by O. Degani.

1. [↑](#footnote-ref-1)