Project Report

High-Speed 8-bit Pipeline Current-Steering D/A Converter

Advanced VLSI Project, 2006

by

Asad Abbas Syed

Department of Electrical Engineering,
Linköping University of Technology
Project Identity
Advanced VLSI Project, 2006
Department of Electrical Engineering,
Linköping University of Technology

Author
Asad Abbas Syed
Masters is System On Chip
Department of Electrical Engineering (ISY)
Phone: +46737706315
Email: sasadrizvi@yahoo.com
Linköping University

Supervisor
Behzad Mesgarzadeh
Electronic Devices,
Department of Electrical Engineering (ISY)
Office: B-house 3A:519,
Phone: +4613-285719
Fax: +4613-139282
Email: behzad@isy.liu.se
Linköping University

Course Responsible
Atila Alvandpour
Professor, Electronic Devices,
Department of Electrical Engineering (ISY)
Office: B-house 3A:514,
Phone: +4613-285818
E-mail: atila@isy.liu.se
Abstract

This document describes the project "High-speed 8-bit pipeline current-steering D/A converter". This project is part of the course "TSEK01 - VLSI Chip Design Project" given at Department of Electrical Engineering, Linköping University.

The complete system consists of digital processing part and analog circuits, which mainly are current sources. For this project a segmented architecture is proposed, in such architecture, circuit is divided into two sub-DACs. For this project four MSB bits are implemented in Unary architecture and four LSB bits implemented in binary weighted architecture. Which means it will have a 15-cell current matrix plus four weighted current sources, each of which scaled by the factor of 2. In digital process part a dual path structure is used. Pipelining is another technique to increase throughput of circuit. In digital processing part logics is broken in a proper way to decrease the length of the critical path. For LSB bits which there are no need to special digital processing logics, just simple pipeline stages will be designed.
1. Introduction

This document describes the project "High-speed 8-bit pipeline current-steering D/A converter". This project is part of the course "TSEK01 - VLSI Chip Design Project" given at Department of Electrical Engineering, Linköping University. The project report gives detailed and in-depth information about the project. It describes the background, time plan, technical, design and other information of the project. This report will also be used to test the manufactured chips in another course (TSEK10).

1.1 Background

The evolution in the field of wireless communications and the mixed-signal area pushes the designer to put an in-creasing amount of effort in the integration of digital and analog systems on one chip. Consequently, the interface between these systems is becoming one of the most challenging blocks to de-sign in the telecommunication devices of today.

High-speed D/A converters used in wide-band transmitter applications require a wide dynamic range since they have to deal with multiple channels. High performance digital-to-analog (D/A) converters find applications in the areas of, e.g., HDTV and GSM. Because they are inherently fast and cost effective, CMOS current-steering D/A converters are the ideal candidates for such applications.

There are different ways to implement current-steering D/A converters, with different features. Some of them are as follows:

**R2R Ladder DAC**, which is a binary weighted DAC that creates each value with a repeating structure of two resistor values, R and R times two. This improves DAC precision due to the ease of producing many equal matched values of resistors or current sources, but lowers conversion speed due to parasitic capacitance.

**Binary weighted implementation**, every switch switches a current source to the output that is twice as large as the next least significant bit. The digital input code directly controls these switches. The advantages of this architecture are its simplicity (since no decoding logic is necessary) and the small required silicon area. On the other hand, a large DNL error and an increased dynamic error are intrinsically linked with this architecture. At the half-scale transition, 2(N-1)unit sources are switched on/off and other independent sources are switched off/on.

**Unary decoded architecture**, every unit current source is addressed separately. The digital input code is converted to a thermometer code that controls the
switches. The advantages of this architecture are its good DNL error and the small
dynamic switching errors. In this architecture, the D/A converter has a guaranteed
monotone behavior since only one additional current source has to be switched to
the output for one extra LSB. The major disadvantage of the unary decoded
architecture is the complexity, the area and the power consumption of the
thermometer decoder.

**Segmented architecture**, I combine the best of binary weighted and segmented
architecture. Most current-steering D/A converters are implemented using a
segmented architecture. In this case, the D/A converter is divided into two sub-
DACs: the B LSBs are implemented using a binary architecture while the (N-B)
MSBs are implemented in a unary way. In this architecture, a balance between
good static and dynamic specifications versus a reasonable decoder power, area,
and complexity can be found.

I will discuss the architecture in chapter three in more details. The last
architecture is proposed in this project. The concept of this architecture is to use a
current-matrix for four MSB bits and weighted current sources for four LSB bits,
which need less precision.

**1.2 Motivation**

The applications of high-speed DACs have been in video and computer graphics
applications, but recently the migration to wideband wired and wire-less
telecommunication standards and the evolution of radio transmitter architectures
toward the software-defined radio have created a need for high-speed, high-
resolution telecommunication DACs.

In the past the research and development of DACs have been heavily concentrated
on improving the static and, to some extent, the time domain specifications
(settling time, glitch area), almost totally neglecting spectral purity and other
frequency domain characteristics that are essential in telecommunication devices.

Practically all high-speed DACs are based on the current steering architecture, one
of the main reasons for this popularity being its capability of driving resistive loads
without buffering. A typical problem in these DACs is the rapid increase in
harmonic distortion when the signal frequency is increased. This is mainly due to
the glitches occurring at the code changes. The glitches are results of incoherent
timing of the current switches, non-optimal shape of the switch control waveforms,
and coupling of the digital signals to the analog output.

Attempts to reduce glitches include the use of latches to synchronize the switch
controls, circuits to generate

1.3 Project goal

The project goal is to design an integrated circuit (IC) in complementary metal-oxide semiconductor (CMOS) technology. Project members should learn the different steps of the IC design flow. That includes the given system architecture analysis, simulation, layout implementation and verification. The project students had an optional choice to manufacture the designed IC circuit on a chip.
2 Project descriptions

2.1 System description

The complete system consists of digital processing part and analog circuits, which mainly are current sources. As mentioned earlier a segmented architecture is proposed. In such architecture, circuit is divided into two sub-DACs. For this project four MSB bits will be implemented in Unary architecture and four LSB bits should be implemented in binary weighted architecture. It means it will have a 15-cell current matrix plus four weighted current sources, each of which scaled by the factor of 2. In digital process part a dual path structure is used. Pipelining is another technique to increase throughput of circuit. In digital processing part logics will be broken in a proper way to decrease the length of the critical path. For LSB bits which there are no need to special digital processing logics, just simple pipeline stages will be designed.

2.2 Important design metrics

The D/A converter has been designed for a high-performance application, meaning that proper measures must be taken to maximize the performance. The most important issue is the speed and throughput of the circuit. However the power consumption has been hold at a reasonable level.

2.3 Area, performance requirements

The table below summarizes the adder performance required by the customer. Each requirement has the given degree of priority. Three degrees of priority had been given: high, medium, and low. High was a firm requirement with no possibility of relaxation, while medium requirements have been relaxed with good motivation.

<table>
<thead>
<tr>
<th>No.</th>
<th>Requirement</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power supply voltage 3.3 V ± 5%</td>
<td>High</td>
</tr>
<tr>
<td>2</td>
<td>Operation frequency 1.5 GHz (Data rate 1.5 GS/s)</td>
<td>High</td>
</tr>
<tr>
<td>3</td>
<td>Integrate as many system components as possible on-chip</td>
<td>High</td>
</tr>
<tr>
<td>4</td>
<td>Schematic and layout must be verified by simulation</td>
<td>High</td>
</tr>
<tr>
<td>5</td>
<td>On-chip evaluation should be implemented, for full speed testing</td>
<td>High</td>
</tr>
<tr>
<td>6</td>
<td>Simulated chip power consumption &lt; 400mW at max. freq.</td>
<td>Medium</td>
</tr>
<tr>
<td></td>
<td>Description</td>
<td>Rating</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>7</td>
<td>Differential output swing for 50-Ohm output &gt;3V</td>
<td>High</td>
</tr>
<tr>
<td>8</td>
<td>Chip design area ~ ~ 1.2 mm²</td>
<td>High</td>
</tr>
<tr>
<td>9</td>
<td>Chip core area &lt; 700μm x 800μm = 0.56mm²</td>
<td>High</td>
</tr>
<tr>
<td>10</td>
<td>Total project pad count &lt; 17 (max 14 active + 3 power supply)</td>
<td>High</td>
</tr>
<tr>
<td>11</td>
<td>Design technology is AMS 4-Metal 0.35μm CMOS</td>
<td>High</td>
</tr>
<tr>
<td>12</td>
<td>The most important system nodes should have off-chip access pins</td>
<td>Medium</td>
</tr>
<tr>
<td>13</td>
<td>On-chip current densities &lt; 1 mA/μm</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 0.1 Summary of the adder performance required by the customer

All requirements in the table should be fulfilled in “typical”, “slow”, and “fast” process corners and temperature between 25 and 110 °C.

Figure 0.1 A 5mm² chip will be shared by 4 independent projects (4 teams). Each project will utilize a 700×800μm² area for core layout and 17 pads.
2.4 Simulation tool

The simulation tool used in this project is Cadence. Cadence products for EDA manage the entire process, including system design, logic synthesis, and layout of integrated circuits. Cadence simulation tools are organized around "platforms" targeted at various types of design tasks which include:

**Virtuoso**, a suite of programs for designing full-custom integrated circuits; includes schematic entry, circuit simulation, full custom layout, and extraction. Used mainly for analog, mixed-signal, RF (w/Agilent EEsof EDA), and standard-cell designs, but also memory and FPGA circuits.

**Encounter**, includes floorplanning, synthesis, test, and place and route-tools for creation of digital integrated circuits. Typically a digital design starts from Verilog netlists.

**Incisive**, includes tools for simulation and functional verification of Verilog, VHDL, and SystemC netlists. Includes formal verification, formal equivalence checking, hardware acceleration, and emulation.

**Silicon-Package-Board**, includes tools for co-design of integrated circuits, packages, and PCBs.

**Allegro** is used to design PCBs, as is ORCAD.

In this project I only employ Composer for schematic design Virtuoso for layout design.

Figure 2.2 shows the design flow of the simulation, in the cadence simulator.
3 Schematic Design simulation

The traditional method for capturing transistor-level or gate-level design is via the schematic editor. Schematic editors provide simple, intuitive means to draw, to place and to connect individual components that make up the design. The resulting schematic drawing must accurately describe the main electrical properties of all components and their interconnections. Also included in the schematic are the power supply and ground connections, as well as all "pins" for the input and output signals of the circuit. This information is crucial for generating the corresponding netlist, which is used in later stages of the design. The generation of a complete circuit schematic is therefore the first important step of the transistor-level design flow. Usually, some properties of the components (e.g. transistor dimensions) and/or the interconnections between the devices are subsequently modified as a result of iterative optimization steps. These later modifications and improvements on the circuit structure must also be accurately reflected in the most current version of the corresponding schematic.

3.1 Architectures

Current-steering D/A converters are based on an array of matched current sources that are switched to the output. Three different architectures are possible depending on the implementation of this array, namely the binary, the unary, and the segmented architecture.

![Figure 0.1 Comprehensive overview of the specifications of unary, binary, and segmented implementations](image-url)
Each architecture will briefly discuss including some advantages and disadvantages. A comprehensive overview is given in Figure 3.1.

3.1.1 The Binary Weighted Architecture

In the binary implementation, every switch switches a current to the output that is twice as large as the next least significant bit. The digital input code directly controls these switches. The advantages of this architecture are its simplicity (since no de-coding logic is necessary) and the small required silicon area. On the other hand, a large DNL error and an increased dynamic error are intrinsically linked with this architecture.

At the half-scale transition, $2N-1$ unit sources are switched on/off and $2N-1 -1$ other independent sources are switched off/on. Assuming a normal distribution for the unit current sources with a standard deviation $\sigma(I)$, this step has a $\sigma(\Delta I)$ determined by

$$\sigma^2(\Delta I) = \sigma^2(2^{N-1} \times I - (2^{N-1} -1) \times I)$$

$$\sigma^2(\Delta I) = 2^{N-1} \sigma^2(I) + (2^{N-1} -1)\sigma^2(I) = (2^N - 1)\sigma^2(I)$$

$$\sigma(\Delta I) = \sqrt{2^N - 1} \frac{\sigma(I)}{I} \text{ LSB}$$

This sigma, $\sigma(\Delta I)$, is a good approximation for the DNL. The $\sigma(\Delta I)$ at this most significant bit transition is approximately a factor $2N-x$ larger than at the other bit transitions (with the total number of bits and the number of the most significant switching bit).

3.1.2 The Unary Decoded Architecture

In the unary decoded architecture, every unit current source is addressed separately. The digital input code is converted to a thermometer code that controls the switches. The advantages of this architecture are its good DNL error and the small dynamic switching errors. In this architecture, the D/A converter has a guaranteed monotone behavior since only one additional current source has to be switched to the output for one extra LSB. The major disadvantage of the unary decoded architecture is the complexity, the area and the power consumption of the thermometer decoder.

Performing similar calculations for the unary architecture leads to the following results:

$$\sigma(\Delta I) = \frac{\sigma(I)}{I} \text{ LSB}$$
This formula mathematically represents the idea behind the unary decoding. The error between two consecutive codes is just the deviation on the additional unity current source. The DNL was defined as the maximum deviation at a single LSB transition. For an - bit converter, this means that the DNL is determined by the maximum when taking samples from a normal distribution with the sigma defined in above equations.

3.1.3 The Segmented Architecture

To get the best of both worlds, most current-steering D/A converters are implemented using a segmented architecture. In this case, the D/A converter are divided into two sub-DACs: the LSBs are implemented using a binary architecture while the MSBs are implemented in a unary way. In this architecture, a balance between good static and dynamic specifications versus a reasonable decoder power, area, and complexity can be found.

Since the segmented architecture is a mixture of the previous two architectures, the result for the most critical transition is of the same form.

\[
\sigma(\Delta I) = \sqrt{2^{B+1} - 1} \frac{\sigma(I)}{I} \text{ LSB}
\]

Note that the above formula for the segmented architecture is a general formula that is valid for the binary and the unary implementation.
3.2 DAC Building Blocks

3.2.1 Unary decoder:
Here I will explain the functionality of the unary decoder part of the D/A converter. The four most significant bits (b5 b6 b7 b8) are decoded in fifteen cells in this part of the converter:

![Figure 0.2 Schematic diagram of Unary Decoder](image1)

I use latches between the blocks and I use two parallel ways for every cell in order to reduce the total delay:

![Figure 0.3 Schematic diagram of a single cell in Unary Decoder](image2)
The In cell block is:

![In Cell Diagram](image)

**Figure 0.4 Schematic diagram of an In cell in Unary Decoder**

First I built and sized the schematic of the standard gates (Inverter, NAND, AND, NOR, OR, latch) and the In cell block, here I will define the basic cells and their sizing complete specification of the cells are given in APPENDIX B.

**Inverter:**

I have taken the minimum size of NMOS transistor as 0.4u which is small and better value. Now the PMOS transistor can be sized by parametric analysis and I choose the parameter where the rise time is equal to fall time i.e 1.2u m.

Sizing:

- NMOS transistor: 0.4um
- PMOS transistor: 1.2um

This is the fundamental for all the remaining circuits where the sizing of Pull Up Network (PUN) is 3 times than the Pull Down Network (PDN). I have also used two kinds more of inverters, one with double size and one with four times size. I have used them to set the same delay for every path and to get better waveforms in critical situations.

**NOR gate:**

From theory, PUN and PDN are the circuits with series or/and parallel circuits. To size any digital circuit I follow the worst case resistance. For example, two NMOS resistors added series the worst case is addition of two resistances. If they are added parallel, the worst case is one of the NMOS transistor resistances.

The PMOS transistors added series and NMOS transistors added parallel. So, to compensate the PUN with PDN, PMOS transistors must be sized 6 times the NMOS transistor which is also proved during the simulations.

Sizing:

- NMOS transistor: 1.8um
- PMOS transistor: 10.8um
**NAND gate:**

In this, PMOS transistors added parallel and NMOS added series. Considering the worst case (size of NMOS should be double of PMOS) and DEC1 (Size of PMOS is 3 times of NMOS), the overall sizing of all PMOS transistors must be 3/2 times of the NMOS transistors.

**Sizing:**

NMOS transistor: 4um  
PMOS transistor: 6um

Actually I have used OR and AND logic circuits, which can be designed by adding an inverter with an specific size to its output.

**In cell block:**

This circuit is a combination of series and parallel circuits in both PUN and PDN. PDN is series of MN0 and MN1|| MN2 and PUN is MP0|| series of (MP1 and MP2).

In order to get good waveforms, I decided to use 4.2um in the transistors of the PDN. The overall sizing of all PMOS transistors must be 3/2 times of the NMOS transistors so I used 6.3um in MP0 and 12.6um in MP1 and MP2 to compensate.

![Figure 0.5 Schematic diagram of an In cell](image)

**Latch:**

Theoretically the size will be minimum (0.4u) for the both PMOS and NMOS but I have got glitches during the simulation. After some simulations I increased both transistors to 2.4u m to get better waveforms.
I also have introduced two inverters after some latches to improve the waveforms. I have also used other sizes for latches in cases where I had to compensate the delay.

**Inverter in DFF:**

![Figure 0.6 Schematic diagram of an Inverter for a D-Flip-flop](image)

Here I will also discuss sizing of an inverter for a D-Flip-flop (DFF). If the same size has been used in both inverters, then it will definitely be seen some glitches due to different delays in the circuit. To solve this problem, I sized the inverter (DFF) to 4 times of the basic inverter which is used in all circuits.

**Sizing:**
- NMOS transistor: 1.6u m
- PMOS transistor: 4.8u m

The schematic for the rest of the cells is similar to this one. I only have to change the logic operators for every cell. I have also included pairs of inverters to compensate the delay when one of the inputs of the in cell block is directly one of the data bits.

The output of these cells goes to a final stage where the path is divided into two paths one for the switch and other for the current source.
Figure 0.7 Schematic diagram of the first cell

Figure 0.8 Schematic diagram of the final stage
In order to get a proper delay in both paths, a latch and an inverter have been added in one path and two inverters in the other.

![Figure 0.9 Schematic diagram of the DFF used to divide the clock.](image1)

I had to divide the clock signal into two signals (clk and clk_bar) with half frequency. I need these signals for the latches. I used a DFF to divide the clock as shown in the above figure.

I inserted buffers for the data bits and for both clock signals. These signals have long ways and I need the buffers to get better waveforms. After some simulations I decided to use buffers with six stages of inverters (1-3-27-81-243).

![Figure 0.10 Schematic diagram of the six stage buffer](image2)

After having these blocks, I was able to complete the unary decoder which is shown in the figure below:
3.2.2 **Binary weighted decoder:**

The four least significant bits are decoded in this part. I don't need the same precision as I need for the four most significant bits, so I don't use the unary decoder for them. Data bits are directly connected to the weighted currents where each bit has its own weighted current value. This is explained with more details later.

In order to compensate the delay, the LSB have a similar structure to the MSB. The
bits have the same buffers and latches. I use some inverters to compensate the delay of the logic gates and the in cell gate. I use the same last stage where the two outputs of each data bit go to the weighted currents.

Figure 03.13 Schematic diagram of Scaled Weighted Decoder
3.2.3 Current mirrors.

Both these parts need to use the current sources of their outputs. Here I have used the cascade current mirror circuits as current sources to produce the required currents flow in the circuits for the Binary and Unary Decoded parts.

A current mirror is a circuit designed to copy a current flowing through one active device by controlling the current in another active device of a circuit, keeping the output current constant regardless of loading.

![Figure 0.15 Basic current Mirror](image)

Transistor Q1 is connected such that it has a constant current flowing through it (due to R1 and Vs) and behaves as a forward-biased diode, and the current is determined mainly by the resistance R1. It is important to have Q1 in the circuit, instead of a regular diode, because the two transistors can be matched, and thus the two branches of the circuit will have similar characteristics.

The voltage at the base of Q1 will necessarily be the exact voltage that sustains the collector current. The second transistor, Q2, which then has the same base voltage, changes its own collector current so that the total effective resistance in the second branch of the circuit is the same as the total resistance in the first branch, regardless of the load resistor, R2. Since the total resistance in each branch is the same, and they are connected to the same supply, VS+, the amount of current in each branch is the same.

The Constant current flowing through R1 can be varied by altering the value of R1 to change the amount of current going through R2. Since R2 can change dynamically, and the current through it will stay the same, the current mirror is a
current regulator.

The output impedance of a current mirror can be increased by adding a cascade stage as shown in the figure:

![Figure 0.16 Cascade current Mirror](image)

Here MN12 is the current source and MN0 is a cascade serves to hold the drain of MN12 at a roughly constant voltage.

I need to use several current mirrors for the Unary and weighted structures at the same time. The advantage of the current mirrors is I can copy the current in the next level if I use same W/L lengths of the transistor and I can increase or decrease the current in the legs by changing the W/L.

\[ I_{L_{eq}} = I_{ref} \left( \frac{W}{L} \right)_{ref} \]

The total differential output swing for 50-Ohm output should be more than 3V. So, total current of the chip should be more than 30mA. I have to made the width of the transistors in series cascade current mirror circuit in such a way that each cell of Unary decoded should be more than 2mA and the weighted part mirrors are
to be scaled down by the factor 2. This is due to currents given to according to the Bit significance. The test schematic as shown in the figure and also presented the currents in each wing by DC analysis.

![Diagram of series cascade current Mirror](image)

**Figure 0.17 Series cascade current Mirror**

Now I need to use 15 current mirror wings for the Unary part with Width of transistor is 24u which shows the current 2.335mA and most significant bit of the weighted code needs more than 1mA which can get by scale down the transistor size to 12u. Similarly, I have scaled down others as well.
3.3 Transistor level simulation results

After the transistor-level description of a circuit is completed using the Schematic Editor, the electrical performance and the functionality of the circuit must be verified using a Simulation tool. The detailed transistor-level simulation of the design will be the first in-depth validation of its operation, hence, it is extremely important to complete this step before proceeding with the subsequent design optimization steps. Based on simulation results, the designer usually modifies some of the device properties (such as transistor width-to-length ratio) in order to optimize the performance.

The initial simulation phase also serves to detect some of the design errors that may have been created during the schematic entry step. It is quite common to discover errors such as a missing connection or an unintended crossing of two signals in the schematic.

The second simulation phase follows the "extraction" of a mask layout (post-layout simulation), to accurately assess the electrical performance of the completed design.

At the transistor level simulation for the digital part consist of introducing a specific digital input so I get a sinusoidal output. I calculated the values of the digital inputs previously and the results are:
The results for the unary and binary decoder without the current mirrors are:
Figure 0.14 Resultant waveform of the unary and binary decoder without current mirror
Figure 0.1 Output wave form of the transistor level final simulation
The outputs of the current mirrors are two ramps (as I expected) when the inputs are the same as the previous simulation. The differential output swing is: $2 \times (3.3 - 1.5) = 3.6V > 3V$. The clock frequency was 2GHz > 1.5GHz and the power consumption was 117.3mW < 400mW. So I fulfilled the requirements one of the specifications.

With the specific inputs I have got Sinusoidal output shown in figure 3.19 with the frequency of 2GHz and the voltage swing was 3.5V.
Figure 0.19 Output wave form from the DAC
4 Layout design simulation

The creation of the mask layout is one of the most important steps in the full-custom (bottom-up) design flow, where the detailed geometries and the relative positioning of each mask layer to be used in actual fabrication are described, using a Layout Editor. Physical layout design is very tightly linked to overall circuit performance (area, speed and power dissipation) since the physical structure determines the transconductances of the transistors, the parasitic capacitances and resistances, and obviously, the silicon area which is used to realize a certain function. On the other hand, the detailed mask layout of logic gates requires a very intensive and time-consuming design effort.

The physical (mask layout) design of CMOS logic gates is an iterative process which starts with the circuit topology and the initial sizing of the transistors. It is extremely important that the layout design must not violate any of the Layout Design Rules, in order to ensure a high probability of defect-free fabrication of all features described in the mask layout.

Much attention has been paid to the final layout of the chip, resulting in a very compact chip. The presented 8-bit D/A converter has an active area of only 0.56mm².

4.1 Current Sources

![Figure 0.1: Layout of Unary decoder](image-url)
4.2 Unary decoder

Figure 0.2: Layout of Unary decoder
4.3 Binary weighted decoder

Figure 0.3: Layout of Binary decoder
4.4 PADding

PADs are used to connect the chip to the outside world; it is a critical part of the chip assembly. In the lab library there are different types of PADs available which depends on the signal

<table>
<thead>
<tr>
<th>VDDPAD</th>
<th>vddcore</th>
</tr>
</thead>
<tbody>
<tr>
<td>APRIOP</td>
<td>OUTOC</td>
</tr>
<tr>
<td>APRIOP</td>
<td>vdd2</td>
</tr>
<tr>
<td>APRIOP</td>
<td>OUTBOC</td>
</tr>
<tr>
<td>GNDPAD</td>
<td>vss</td>
</tr>
<tr>
<td>APRIOP</td>
<td>VbOC</td>
</tr>
<tr>
<td>APRIOP</td>
<td>B8OC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VDDGNDCORNER</th>
<th>B1OC</th>
<th>B6OC</th>
<th>B1OC</th>
<th>B4OC</th>
<th>B1OC</th>
<th>B2OC</th>
<th>B1OC</th>
<th>clkOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDDGNDCORNER</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
<td>APRIOP</td>
</tr>
</tbody>
</table>

Figure 0.4: Block diagram shows the PADs setting

Pads and there descriptions

**APRIOP**
Analog Input/Output PAD
ESD protection included
Signal accessed in metal 2 on core-side of the PAD
Padcap ~1.2pF

**UNPROTPAD**
Analog Input/Output PAD
No ESD protection
Signal accessed in metal 2 on core-side of the PAD
Suitable for high-speed I/O and power supply with different voltage than core
**VDDGNDCORNER**
Default power supply PAD for this project
Provides supply voltage and ground to the PAD ring and the Core
VDD and GND will be accessed on-chip from the PAD-ring

**Figure 0.5: Layout of the PADs**

**CORNERFILL**
Fill cell that has been placed under the VDDGNDCORNER pad
Its Increases fill rate inPOLY1

**VDDPAD**
Separate power supply PAD
Includes ESD protection
Supply to PAD ring and core

**GNDDPAD**
Separate ground PAD
Includes ESD protection
Ground PAD ring and core

**PAD Ring**

Maximum amount of PADs is 15 (17)
The 2 PADs in the corner are fixed
Corner PADs can not handle large currents

**4.5 Decoupling capacitors**

I used decoupling capacitor to prevent transfer of high-frequency noise between electrical nodes. The most common use of decoupling capacitors is on power supply rails where they prevent voltage droop when transient voltage spikes/current spikes are experienced. These capacitors can be viewed as small localized energy reservoirs. I placed Decoupling capacitors in parallel with the two nodes to decouple, and as close as possible to the devices.

![Figure 0.6: Layout of Decoupling Capacitors](image)

This causes the capacitors to act as low-pass filters. The further the capacitors are from the device the more series resistance and series inductance is increased due to trace length and the more ineffective the low-pass filtering becomes. In addition to
being placed close to the devices, they are also placed as close to the voltage sources as possible. This prevents voltage drop due to ramp time from the source.

4.6 Pfill

Pfill is introduced into sparse regions of the layout to equalize the spatial density of the layout, improving uniformity of chemical-mechanical planarization (CMP). Pfill insertion for CMP uniformity changes the back-end flow with respect to layout, parasitic extraction and performance analysis.

Figure 0.7: Layout of PFill
4.7 Buffers

Figure 0.8: Layout of 3 stage Buffer
4.8 Final DAC

Figure 0.9: Layout of Final DAC
5 Post-layout Simulation

The electrical performance of a full-custom design can be best analyzed by performing a post-layout simulation on the extracted circuit net-list. At this point, there should be a complete mask layout of the intended circuit/system, and should have passed the DRC and LVS steps with no violations. The detailed (transistor-level) simulation performed using the extracted net-list will provide a clear assessment of the circuit speed, the influence of circuit parasitics (such as parasitic capacitances and resistances), and any glitches that may occur due to signal delay mismatches.

When the results of post-layout simulation were not satisfactory, I modified some of the transistor dimensions and/or the circuit topology, in order to achieve the desired circuit performance under "realistic" conditions, i.e., taking into account all of the circuit parasitic. This may require multiple iterations on the design, until the post-layout simulation results satisfy the original design requirements.

But the satisfactory results in post-layout simulation were still no guaranteed for a completely successful chip; the actual performance of the chip can only be verified by testing the fabricated prototype. Even though the parasitic extraction step is used to identify the realistic circuit conditions to a large degree from the actual mask layout, most of the extraction routines and the simulation models used in modern design tools have inevitable numerical limitations.

5.1 Design Rule Check (DRC)

The created mask layout must conform to a complex set of design rules, in order to ensure a lower probability of fabrication defects. A tool built into the Layout Editor, called Design Rule Checker, has been used to detect any design rule violations during and after the mask layout design. The detected errors are displayed on the layout editor window as error markers, and the corresponding rule is also displayed in a separate window. I perform DRC frequently, and make sure that all layout errors are eventually removed from the mask layout, before the final design is saved.

5.2 Layout versus Schematic Check

After the mask layout design of the circuit is completed, the design has been checked against the schematic circuit description created earlier. The design called "Layout-versus-Schematic (LVS) Check" has compared the original network with the one extracted from the mask layout, it proved that the two networks are indeed
The LVS step provides an additional level of confidence for the integrity of the design; it ensured that the mask layout is a correct realization of the intended circuit topology. But the LVS check only guarantees topological match: A successful LVS will not guarantee that the extracted circuit will actually satisfy the performance requirements. The errors which showed up during LVS (such as unintended connections between transistors, or missing connections/devices, etc.) have been corrected in the mask layout before proceeding to post-layout simulation.

5.3 Circuit Extraction

Circuit extraction is performed after the mask layout design is completed, in order to create a detailed net-list (or circuit description) for the simulation tool. The circuit extractor is capable of identifying the individual transistors and their interconnections (on various layers), as well as the parasitic resistances and capacitances that are inevitably present between these layers. Thus, the "extracted net-list" can provide a very accurate estimation of the actual device dimensions and device parasitics that ultimately determine the circuit performance. The extracted net-list file and parameters are subsequently used in Layout-versus-Schematic comparison and in detailed transistor-level simulations (post-layout simulation).
5.4 Results

The final results of the post layout simulation were much closer to the results of transistor level simulations. For the Ramp simulation clock frequency I have got is 1.54GHz > 1.5GHz, while it was 2GHz > 1.5GHz in transistor level simulation. The voltage swing is 3.4V > 3V, which is 0.2V less than transistor level. Power consumption has reduced to 112.3mW which was 117.3mW before, the total area I used for DAC is 0.222mm².
6 Problems

While the project I faced some problems which are describe here briefly
Divide clock frequency.
DFF device.
Layout problems.
Delay compensation.
Clock for DFF.
Strong buffers for layout.
Layout simulation problems.
Inverters after latch.
Delay problems.
Layout problems.
Pad selection.
Vdd division.
Decap division.
No antenna errors.
Outputs : 32mA.
In this report, High-Speed 8-bit Pipeline Current-Steering D/A Converter up to 1.54 GHz Operation frequency (Data rate 1.54 GS/s) has been presented. The D/A converter has a differential output swing for 50-Ohm output > 3.4V and chip power consumption < 112.3mW at maximum frequency. Much attention has been paid to the layout. Not only have the layout parasitic effects been iterated into the design phase, but the floor plan of the analog and digital block has also been studied to minimize the chip area. The D/A converter has been processed in a standard AMS 4-Metal 0.35μm CMOS design technology and has Chip design area ≈ 1.2 mm² with chip core area < 700μm x 800μm = 0.56mm², the total utilizing area for DAC is 0.222mm². The On-chip current density is < 1 mA/μm and the power supply voltage is 3.3 V ± 5%. Total project pad count < 17 (max 14 active + 3 power supplies)
APPENDIX A

Abbreviations

VLSI – Vary Large Scaled Integration

DAC – Digital-to-analog converter. A device that maps digital code words onto a continuous-time signal.

ADC – Analog-to-digital converter. Device that performs uniform sampling and amplitude quantization.

DSP – Digital signal processing. Includes various linear and nonlinear operations performed on data from a finite set, such as filtering, transformations, and coding.

SFDR – Spurious-free dynamic range. The difference, in decibel, between the fundamental tone and the largest spurious distortion term.

SNDR – Ratio of signal power to noise and distortion power in decibels, when a sinusoidal signal is applied as input signal.

SNR – Ratio of signal power to noise power in decibels, when a sinusoidal signal is applied as input signal.

LVS – Layout-versus-Schematic Check

DRC – Design Rule Check
## APPENDIX B

### Basic Cells For Layout

#### Latches

<table>
<thead>
<tr>
<th>Types</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scaled_Latch_for_DFF</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>1.65</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>1.65</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td>Outputs</td>
<td>output</td>
</tr>
<tr>
<td><strong>Scaled_Latch_bar_for_DFF</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>3</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>3</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td>Outputs</td>
<td>output</td>
</tr>
<tr>
<td><strong>Scaled_Latch</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td>Outputs</td>
<td>output</td>
</tr>
<tr>
<td><strong>Scaled_Latch_bar</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td>Outputs</td>
<td>output</td>
</tr>
<tr>
<td><strong>Scaled_Latch+inv</strong></td>
<td>Also use Scaled_inverter and Scaled_inverter_DFF</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td>Outputs</td>
<td>output</td>
</tr>
<tr>
<td><strong>Scaled_Latch_bar+inv</strong></td>
<td>Also use Scaled_inverter and Scaled_inverter_DFF</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, input, clk, clkb</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Outputs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Inverters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Types</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Scaled_inverter_DFF</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>1.65</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>4.8</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, A</td>
</tr>
<tr>
<td>Outputs</td>
<td>Abar</td>
</tr>
<tr>
<td><strong>Scaled_inverter</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>0.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>1.2</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, A</td>
</tr>
<tr>
<td>Outputs</td>
<td>Abar</td>
</tr>
<tr>
<td><strong>last_stage_inverter</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, d, vss</td>
</tr>
<tr>
<td>Outputs</td>
<td>, dbar, dlatch</td>
</tr>
<tr>
<td><strong>Scaled_inverter_in_cell</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>0.8</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, A</td>
</tr>
<tr>
<td>Outputs</td>
<td>Abar</td>
</tr>
<tr>
<td><strong>Buffers</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Types</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Buffer_6_stages</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>0.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>1.2</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, buffin</td>
</tr>
<tr>
<td>Outputs</td>
<td>, buffout</td>
</tr>
<tr>
<td>Buffer_3_stages</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>0.4</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>1.2</td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, buffin</td>
</tr>
<tr>
<td>Outputs</td>
<td>buffout</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Buffer_2_stages_large</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NMOS Size</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMOS Size</td>
<td>2.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inputs</td>
<td>vdd, vss, buffin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outputs</td>
<td>buffout</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Gates

### Types

#### Scaled_OR_gate

| NMOS Size | 1.8 |   |   |
| PMOS Size | 10.8| 2.4|   |
| Inputs    | A, B, vdd, vss | 7.2|   |
| Outputs   | A+B |   |   |

#### Scaled_AND_gate

| NMOS Size | 4   |   |   |
| PMOS Size | 6   | 2.4|   |
| Inputs    | A, B, vdd, vss | 7.2|   |
| Outputs   | AB  |   |   |

#### Scaled_in_cell

| NMOS Size | 4.2 | R2 | C  |
| PMOS Size | 6.3 | 4.2| 4.2|
| Inputs    | R1, R2, C, vdd, vss | 12.6| 12.6|
| Outputs   | , output |   |   |

## Current sources for Use

<p>| Inputs   | c1l, c2l, c3l, c4l, c5l, c6l, c7l, c8l, c9l, c10l, c11l, c12l, c13l, c14l, c15l, B1l, B2l, B3l, B4l, c1inv, c2inv, c3inv, c4inv, c5inv, c6inv, c7inv, c8inv, c9inv, c10inv, c11inv, c12inv, c13inv, c14inv, c15inv |   |   |
|          | c1inv, c12inv, c13inv, c14inv, c15inv |   |   |</p>
<table>
<thead>
<tr>
<th>Outputs</th>
<th>B1inv, B2inv, B3inv, B4inv, vdd, Vb, vss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Types</td>
<td>OUT, OUTB</td>
</tr>
<tr>
<td><strong>Fundamental</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>1</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>10</td>
</tr>
<tr>
<td><strong>Mirror</strong></td>
<td></td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td>NMOS Size</td>
<td>24</td>
</tr>
<tr>
<td>PMOS Size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>NMOS Size</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>C12</strong></td>
<td>24</td>
</tr>
<tr>
<td><strong>C13</strong></td>
<td>24</td>
</tr>
<tr>
<td><strong>C14</strong></td>
<td>24</td>
</tr>
<tr>
<td><strong>C15</strong></td>
<td>24</td>
</tr>
<tr>
<td><strong>B1</strong></td>
<td>1.5</td>
</tr>
<tr>
<td><strong>B2</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>B3</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>B4</strong></td>
<td>12</td>
</tr>
</tbody>
</table>
Modules For Layout

1. DFF
Inputs: vdd, clk, vss
outputs: clk_2, clkb_2
- Buffer_3_stages
- Buffer_2_stages_large
- Scaled_latched_for_DFF
- Scaled_latched_bar_for_DFF
- Scaled_inverter_DFF
- Scaled_inverter

2. Final_Dac
Inputs: B1-B8, clk_2, clkb_2, vdd, Vb, vss
outputs: OUT, OUTB

A. Scaled_weighted_decoder
Inputs: B1-B4, clk, clkb, vdd, vss
outputs: B1l-B4l, B1inv-B4inv
  I. last_stage_inverter
  II. Scaled_weighted
    Inputs: clk, clkb, input, vdd, vss
    Outputs: output
    - Scaled_latch
    - Scaled_inverter_in_cell
    - Scaled_latch_bar
    - Scaled_inverter_DFF

B. Scaled_Unary_Decoder
Inputs: B5-B8, clk, clkb, vdd, vss
Outputs: c1-c15l, c1inv-c15inv
  I. last_stage_inverter
  II. Unary_Decoder
    Inputs: clk, clkb, B5-B8, vdd, vss
    Outputs: c1-c15
    - Scaled_1st_in_cell
      Inputs: clk, clkb, B5-B8, vdd, vss
      Outputs: c1
    - Scaled_Latch+inv
    - Scaled_Latch_bar+inv
- Scaled_2nd_in_cell
  Inputs: clk, clkb, B6-B8, vdd, vss
  Outputs: c2
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_OR_gate
  - Scaled_in_cell
  - Scaled_latch
  - Scaled_latch_bar
  - Scaled_inverter
  - Scaled_inverter_DFF

- Scaled_3rd_in_cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c3
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_OR_gate
  - Scaled_AND_gate
  - Scaled_in_cell
  - Scaled_latch
  - Scaled_latch_bar

- Scaled_4th_in_cell
  Inputs: clk, clkb, B7, B8, vdd, vss
  Outputs: c4
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_OR_gate
  - Scaled_in_cell
  - Scaled_latch
  - Scaled_latch_bar

- Scaled_5th_in_cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c5
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_OR_gate
  - Scaled_in_cell
  - Scaled_latch
  - Scaled_latch_bar
  - Scaled_inverter
  - Scaled_inverter_DFF

- Scaled_6th_in_cell
  Inputs: clk, clkb, B6-B8, vdd, vss
  Outputs: c6
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
- Scaled OR gate
- Scaled in cell
- Scaled latch
- Scaled latch bar
- Scaled inverter
- Scaled inverter DFF

- Scaled 7th in cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c7
  - Scaled Latch+inv
  - Scaled Latch_bar+inv
  - Scaled OR_gate
  - Scaled AND gate
  - Scaled in_cell
  - Scaled latch
  - Scaled latch_bar
  - Scaled inverter
  - Scaled inverter DFF

- Scaled 8th in cell
  Inputs: clk, clkb, B7, B8, vdd, vss
  Outputs: c8
  - Scaled Latch+inv
  - Scaled Latch_bar+inv
  - Scaled OR gate
  - Scaled in_cell
  - Scaled latch
  - Scaled latch_bar
  - Scaled inverter
  - Scaled inverter DFF

- Scaled 9th in cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c9
  - Scaled Latch+inv
  - Scaled Latch_bar+inv
  - Scaled OR gate
  - Scaled AND gate
  - Scaled in_cell
  - Scaled latch
  - Scaled latch_bar
  - Scaled inverter
  - Scaled inverter DFF

- Scaled 10th in cell
  Inputs: clk, clkb, B6-B8, vdd, vss
  Outputs: c10
  - Scaled Latch+inv
  - Scaled Latch_bar+inv
  - Scaled AND gate
  - Scaled in_cell
  - Scaled latch
  - Scaled latch_bar
  - Scaled inverter
- Scaled_inverter_DFF
- Scaled_11th_in_cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c11
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_AND_gate
  - Scaled_in_cell
  - ScaledLatch
  - ScaledLatch_bar
  - Scaled_inverter
  - Scaled_inverter_DFF
- Scaled_12th_in_cell
  Inputs: clk, clkb, B7, B8, vdd, vss
  Outputs: c12
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_AND_gate
  - Scaled_in_cell
  - ScaledLatch
  - ScaledLatch_bar
  - Scaled_inverter
  - Scaled_inverter_DFF
- Scaled_13th_in_cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c13
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_OR_gate
  - Scaled_AND_gate
  - Scaled_in_cell
  - ScaledLatch
  - ScaledLatch_bar
- Scaled_14th_in_cell
  Inputs: clk, clkb, B6-B8, vdd, vss
  Outputs: c14
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_AND_gate
  - Scaled_in_cell
  - ScaledLatch
  - ScaledLatch_bar
  - Scaled_inverter
  - Scaled_inverter_DFF
- Scaled_15th_in_cell
  Inputs: clk, clkb, B5-B8, vdd, vss
  Outputs: c15
  - Scaled_Latch+inv
  - Scaled_Latch_bar+inv
  - Scaled_AND_gate
○ Scaled_in_cell  
○ Scaled_latch  
○ Scaled_latch_bar

**PAD Description**

<table>
<thead>
<tr>
<th>PAD Selection</th>
<th>Description</th>
<th>Input/Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Vdd</td>
<td>In/Out</td>
</tr>
<tr>
<td>2</td>
<td>Vss</td>
<td>In/Out</td>
</tr>
<tr>
<td>3</td>
<td>B1</td>
<td>Input</td>
</tr>
<tr>
<td>4</td>
<td>B2</td>
<td>Input</td>
</tr>
<tr>
<td>5</td>
<td>B3</td>
<td>Input</td>
</tr>
<tr>
<td>6</td>
<td>B4</td>
<td>Input</td>
</tr>
<tr>
<td>7</td>
<td>B5</td>
<td>Input</td>
</tr>
<tr>
<td>8</td>
<td>B6</td>
<td>Input</td>
</tr>
<tr>
<td>9</td>
<td>B7</td>
<td>Input</td>
</tr>
<tr>
<td>10</td>
<td>B8</td>
<td>Input</td>
</tr>
<tr>
<td>11</td>
<td>Clk</td>
<td>Input</td>
</tr>
<tr>
<td>12</td>
<td>Vb</td>
<td>Input</td>
</tr>
<tr>
<td>13</td>
<td>OUT</td>
<td>Output</td>
</tr>
<tr>
<td>14</td>
<td>OUTB</td>
<td>Output</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
References

1. Course home page www.ek.isy.liu.se/courses/tsek01/

2. Project description and requirement specification of the Project "High-Speed 8-bit Pipeline Current-Steering D/A Converter"

3. A 10-bit 1-GSample/s Nyquist Current-Steering CMOS D/A Converter, Anne Van den Bosch, Student Member, IEEE, Marc A. F. Borremans, Student Member, IEEE, Michel S. J. Steyaert, Senior Member, IEEE, and Willy Sansen, Fellow, IEEE

4. JBIG Compression Algorithms for “Dummy Fill” VLSI Layout Data Robert Ellis, Andrew Kahng, and Yuhong Zheng, VLSI CAD Laboratory UCSD Department of Computer Science and Engineering


7. www.vlsi.wpi.edu/cds/