Phase Locked Loop Analysis and Design

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

Marcus Ratcliff

Certificate of Approval:

Thaddeus Roppel
Associate Professor
Electrical and Computer Engineering

Fa Foster Dai, Chair
Professor
Electrical and Computer Engineering

Guofu Niu
Alumni Professor
Electrical and Computer Engineering

George T. Flowers
Interim Dean
Graduate School

Phase Locked Loop Analysis and Design

Marcus Ratcliff

A Thesis

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama
December 19, 2008

Phase Locked Loop Analysis and Design

Marcus Ratcliff

_____

Signature of Author

_____

Date of Graduation

Thesis Abstract

Phase Locked Loop Analysis and Design

Marcus Ratcliff

Master of Science, December 19, 2008
(Electrical Engineering B.S., Auburn University, 2006)

64 Typed Pages

Directed by Fa Foster Dai

The components of the phase locked loop (PLL) circuit discussed in this thesis are designed for use in a transmit receive (TR) module contracted out to Auburn University by United States Space and Missile Defense (USSMDC) based in Huntsville, AL. IBMs SiGe 8hp technology is considered to be on the cutting edge of the radio-frequency integrated circuit design world, and was needed to meet the constraints set forth by the objectives of the TR module. There will be a brief introduction to PLLs, followed by a more in-depth look at architectures chosen for use. This will be followed by simulations and comparisons of the architecture blocks used and how they interact with other blocks.

Acknowledgments

The author would like to thank Fa Foster Dai, acting chair professor, for basis and continual help on design. For participating on thesis committee thanks to Guofu Niu and Thaddeus Roppel. Finally I'd like to mention Mark Ray and William Souder for multi-modulus divider and voltage controlled oscillator schematics used in Cadence simulations.

Style manual or journal used Journal of Approximation Theory (together with the style known as "aums"). Bibliograpy follows van Leunen's *A Handbook for Scholars.*

Computer software used The document preparation package TEX (specifically LATEX) together with the departmental style-file `aums.sty`.

Table of Contents

List of Figures

Chapter 1

Introduction

The phase locked loop (PLL) is one of the most common frequency synthesizers in use today [1]. There are many applications for the PLL in today's growing integrated circuit design field, like the TR module this PLL is being designed for. Figure 1.1 shows the PLL's placement within a 2005 design proposal of the TR module. The PLL constructed here is used to impose the crystal oscillator's frequency stability characteristic on that of the voltage controlled oscillator (VCO) used in the loop.

There are many ways of looking at the basic building blocks of a PLL system. Figure 1.2 shows the PLL broken up into five fundamental blocks, each of which will be discussed and expanded upon later.

The crystal oscillator in the system, as already discussed, is used for its frequency stability but is also used for low phase noise characteristics. When used correctly with a PLL, the high phase noise and unstable frequency stability of a VCO can be improved.

The phase frequency detector (PFD) of the system is in control of keeping the VCO in phase with the carrier signal. When a change in phase between the carrier signal and the VCO occurs, the dc control voltage of the PFD will shift up or down to change the frequency of the VCO in an attempt to track the carriers frequency.

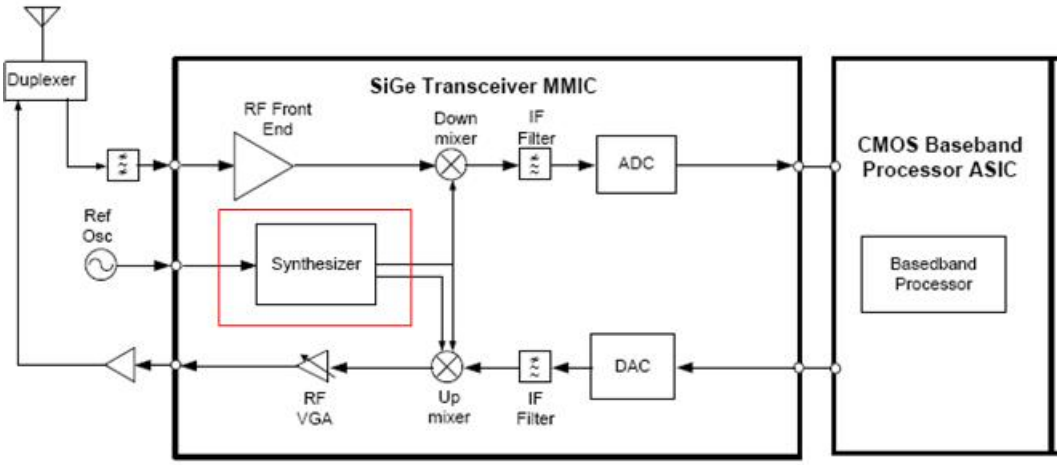

Figure 1.1: Design proposal of TR module

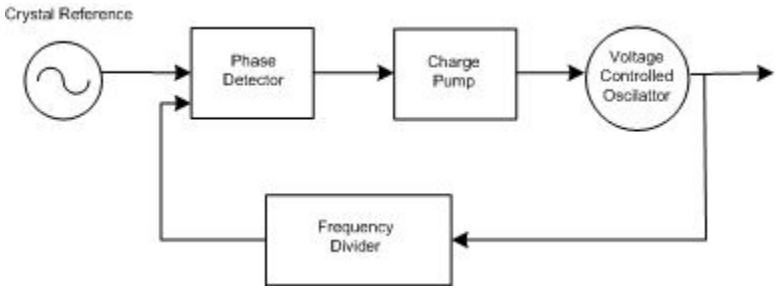

Figure 1.2: Basic PLL structure

The control voltage used to shift the frequency of the VCO is varied through the use of a charge pump inserted into the PLL circuit. The charge pump is a major contributor of phase noise, therefore it required extra attention. There are many variations of charge pumps, the one chosen for implementation will be discussed in further detail later.

The VCO in the system is a tuneable oscillator that is controlled by increasing or decreasing the voltage applied to change its frequency. A brief overview of this component will be presented in a later section.

The frequency divider of a PLL simply lowers the output frequency of the VCO by a programmed amount. The two basic choices for a frequency divider are to divide by an integer (integer-N) or divide by a fraction (fractional-N) [2]. The fractional-N frequency divider was chosen for its finer step size, higher reference frequency capabilities, and coverage of a broader range of frequencies. This synthesizer will be discussed in further detail in the following chapter.

Chapter 2

Synthesizer Architectures

To acquire a fully programmable synthesizer, a fractional-N with multimodulus divider (MMD) was used [2]. Figure 2.1 shows the general case of a fractional-N frequency synthesizer with an MMD.

The setup of this general case is relatively simple to grasp by evaluating the equations produced by the diagram. The step size of the fractional-N architecture is given by

$$StepSize = \frac{f_r}{RF}$$

Where f_r is the reference frequency, R is the f_r divisor, and F is the size of the accumulator. This leads one to the output frequency (f_o) of this architecture as being

$$f_o = \frac{f_r}{R}(I + \frac{K}{F})$$

where K is a user defined fractional-divider, and I is the integer portion of the of the loop divisor.

It is important to look at each of these variables and note how they work together. For instance, it is imperative to keep R as small as possible to minimize in-band phase noise from the oscillating crystal. It is also crucial to keep the f_r divisor fixed to keep the resulting comparison frequency unchanging. The significance of this will be seen more clearly after further discussion. Since F is the size of the accumulator, it is also a given to say that its bit size is given by $log_2 F$, and that an overflow occurs at the output whenever the input is

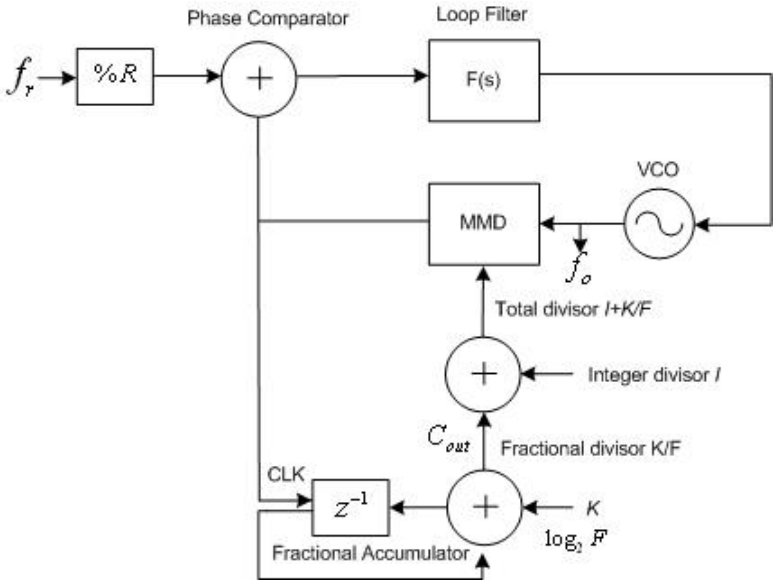

Figure 2.1: A fractional-N frequency synthesizer with an MMD ([3])

equal to or larger than F. Since F must be a fixed size due to the limited number of bits reachable in present hardware implementation, K is the only user programmable parameter within this system. K ranges from one to its maximum F. By gaining an understanding of the synthesizer design, it can now be seen that it is an integral part within the building of the PLL. Using this application the user is able to define what range is needed.

A popular form of MMD topology is the use of cascaded 2/3 cells [14]. This is a form used within this PLL. With an n-bit modulus control signal, the MMD division ratio is given by [2]

$$N_{MMD} = P_1 + 2^1 P_2 + ... + 2^{n-2} P_{n-1} + 2^{n-1} P_n + 2^n$$

This gives a corresponding programming range of 2^n to $2^{n+1}-1$. Knowing this, one can see that a wide programming range can be reached. Further investigation of this topology with a variation of this technique will be discussed within the Multiple Modulus Divider chapter.

Chapter 3

System Overview and Analysis

At this point it is important to look more in depth at each component within the PLL and see how they operate. To do this, each block will be broken down as discussed earlier in the introduction and it will be shown how they interact together. In the following chapters the architectures of each of these blocks will be reviewed and their implementation within SiGe technology will be discussed.

3.1 Phase Detector and Charge Pump

Viewing the basic PLL structure in Figure 1.2, the connected blocks will be evaluated. The first thing to notice is the dual signals that are received by the PFD, the output of the crystal oscillator (v_{IN}), and the output of the frequency divider (v_{FD}). It can be assumed that the input from the crystal oscillator is of the form

$$v_{IN}(t) = V_{IN}sin(w_{IN}t + \theta_{IN})$$

It can also be assumed that the output of the frequency detector which is fed by the VCO is of the form

$$v_{FD}(t) = V_{FD}sin(w_{FD}t + \theta_{FD})$$

Knowing the form of these signals, it is imperative to notice that the phase detector essentially multiplies its two inputs together to acquire a phase difference. Also noting the purpose of the PLL, which is to eventually lock, it can be assumed that the frequency is

locked leaving w_{IN} and w_{FD} equal. Using these characteristics the output of the PFD can be shown to hold the form of

$$v_{PFD}(t) = V_{IN}sin(wt + \theta_{IN})V_{FD}sin(wt + \theta_{FD})$$

Using trigonometric identities and assuming that the loop filter following the PFD will allow the neglect of higher order components, the following equation to describe the output of the phase detector can be reached

$$v_{PFD} = K_{PFD}[\theta_{IN} - \theta_{FD}]$$

This gives an equation for the output of the phase detector where K_{PFD} is equal to $\frac{V_{IN}V_{FD}}{2}$.

From this point it should be noted that for the purposes of the PLL presented here, a tristate phase detector was used. The architecture and characteristics of which will be looked at more in-depth in following chapters. For the benefit of understanding the relationship between the PFD and the charge pump this must be known since the output of a tristate phase detector produces two signals, UP and DOWN. The UP signal is telling the VCO to speed up to catch up in phase with that of the input signal. On the other hand the DOWN signal is attempting to slow down the VCO in phase allowing the input signal to catch up.

To input these signals into the VCO, the differential signals must be converted into a single analog signal. To achieve this goal a charge pump was used. The charge pump is made up of two controllable current sources connected to a common output, which is shown in Figure 3.1. These signals will therefore either charge or discharge the capacitors that, as will be seen in later chapters, are connected to the input of the VCO.

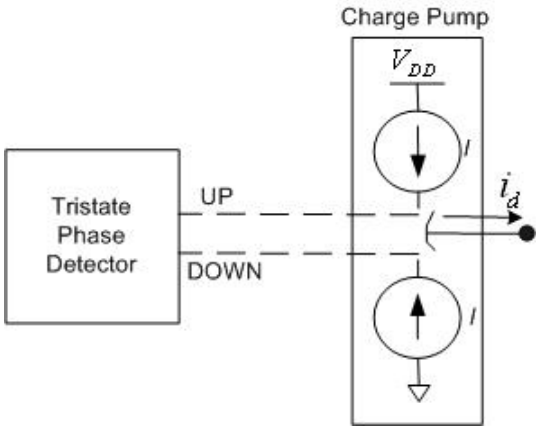

Figure 3.1: A tristate phase detector connected to a charge pump ([4])

Knowing the basics of the UP and DOWN signals and how they associate with the charge pump, the resulting characteristics can be evaluated. From this point it should now be clear to see that with an UP signal present the output will be charged up. Contrastingly with a DOWN signal, discharging of the output should occur with current flowing out of the charge pump. An important relationship to notice is that

$$i_d = I\frac{\tau}{T} = (\frac{I}{2\pi})(\theta_{IN} - \theta_{FD})$$

where τ is time that current flows, T is the period, and I is the current that flows through the current sources within the charge pump. Looking at this equation it can be seen that i_d will be positive in the θ_{IN} leading θ_{FD} case, and negative for the opposite. Thus showing that the output is of the form wanted.

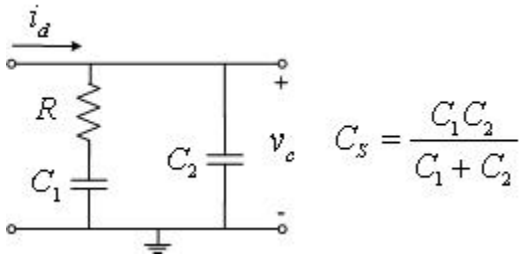

Figure 3.2: An example of a simple loop filter

3.2 Loop Filter

Although the loop filter that will be implemented within the PLL will be located off-chip, and will therefore not be discussed in detail here, it is important to review the qualities of this component to paint a clear picture. The majority of VCOs used today are dependent on voltage change to adjust frequency output. Knowing this, a loop filter is needed to change the output current produced by the charge pump into a voltage for use by the VCO.

Figure 3.2 shows an example of one of these loop filters. The current i_d is inputed into the filter from the charge pump, which is then converted to the control voltage (v_c) that is wanted for the VCO. To relate the loop filter's relationship to the charge pump v_c can be solved for by dividing i_d by the admittance of the filter (Y) [2].

$$v_c = \frac{i_d}{Y} = \frac{\frac{I}{2\pi}(\theta_R - \theta_o)(1 + sC_1R)}{s(C_1 + C_2)(1 + sC_sR)}$$

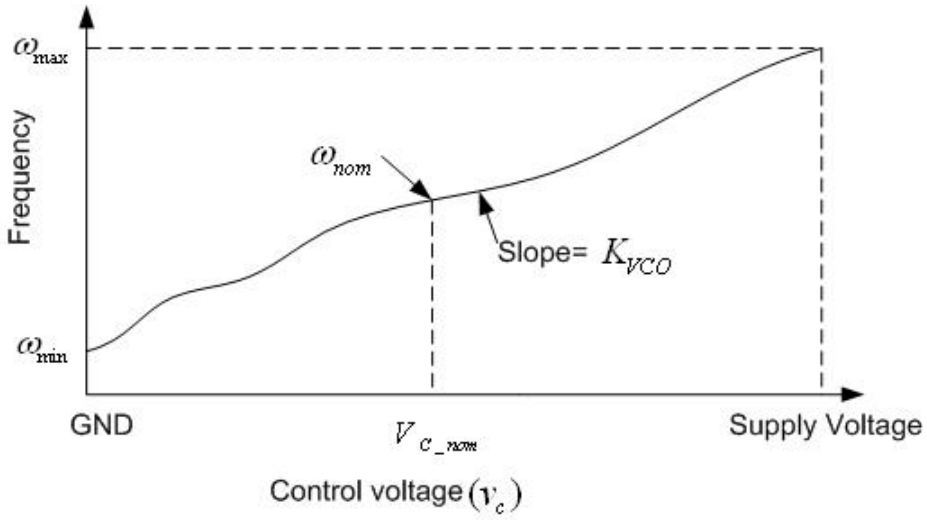

Figure 3.3: A typical VCO characteristic ([5])

3.3 Voltage Controlled Oscillator and Frequency Divider

The simplest way to show the implementation of a basic VCO is to view a typical VCO's characteristics, Figure 3.3 shows this. As the supply voltage increases, the VCO's output frequency also increases. When tying the VCO into the output of the off-chip filter, which is fed by the output of the charge pump, it is important to note these characteristics. The polarization of the charge pump may have to be switched, depending on both the VCO characteristics and the output of the phase detector.

The VCO is a signal generator. The frequency of the generated signal depends on the instantaneous value of the input voltage of the VCO. This input voltage will be the output from an off-chip filter and will be referred to as v_c. The VCO will operate at a frequency of f_{nom} which is dependent on the input v_c, this will also have a corresponding w_{nom}. Now

the output of the VCO can be defined as

$$v_{OUT} = cos((w_{nom} + K_{VCO}v_c)t)$$

K_{VCO} is sometimes referred to as VCO gain and is therefore specific to the type of VCO used. From this point the next step is to define the output frequency as

$$w_{OUT} = w_{nom} + K_{VCO}v_c$$

Since the goal is to relate this to the rest of the PLL a relationship between the VCO and phase needed to be found. To do this the following form was used

$$w = \frac{d\theta}{dt}$$

Knowing this it can be shown that the integral of w_{VCO} gives the following

$$\theta_{VCO} = \int w_{VCO}\, dt = K_{VCO} \int_0^t v_c(\tau)\, d\tau$$

Finally the transfer function can be found, including output phase of the VCO, using the laplace transform over time which is 1/s. This gives the form

$$\frac{\theta_{VCO}(s)}{v_c(s)} = \frac{K_{VCO}}{s}$$

Looking at this transfer function it should be noticed that the VCO can be considered as an integrator for the phase. Using this characteristic the relationship between the VCO and the frequency divider can be easily found. For simplicity, the transfer function will be extended to

$$\frac{\theta_{VCO}(s)}{v_c(s)} = \frac{1}{N}\frac{K_{VCO}}{s}$$

The structure and functionality of the PLL has now been covered completely, in a general sense. The next step is to look at each component in-depth and review the architectures that were implemented within the PLL built.

Chapter 4

Phase Detector

4.1 Architectures

There are many PFD designs present in the integrated circuit design community. A few of these architectures will be discussed to get an overview of the options that are available when choosing a PFD.

The most basic architecture for a PFD to consider would be the case of the exclusive OR gate (XOR). The truth table shown in Figure 4.1 gives the best explanation of how the XOR would act as a PFD. When both A and B inputs are close to equal phase the output will be low, for most cases, and therefore a logic zero. On the other hand, if A and B inputs are somewhat or completely out of phase, the output will be high and considered a logic one.

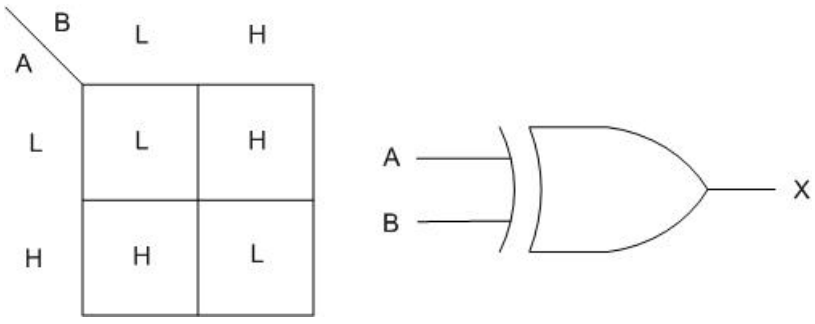

Figure 4.1: XOR

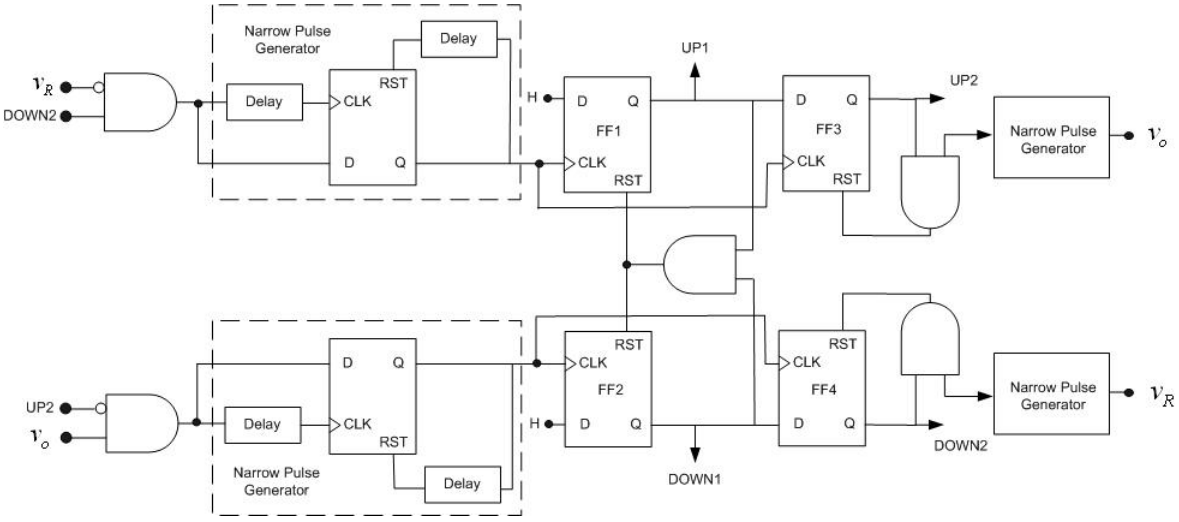

Figure 4.2: An example of a five-state PFD ([6])

One issue that will arise when using the XOR as a PFD is what to consider as the threshold voltage to assume logical one or zero. This can later be adjusted using the filter following the the PFD block. Knowing what characteristics are wanted out of a PFD from previous sections, the XOR can be easily viewed as a choice for a PFD within a PLL.

Now that a simpler version of a PFD has been reviewed, a broader look at a more complex version will be discussed. This version will be the five-state PFD circuit which can be seen in Figure 4.2. It consists of a tristate PFD, narrow pulse generators, and other circuitry that extend its state.

There are many advantages to increasing the states of the PFD. The main improvement is settling time. In [2] a comparison of tristate and five-state PFD settling times shows that there is a ten to fifteen microsecond decrease in settling time by increasing the states. While this is a notable improvement over the next architecture discussed, the additional circuitry outweighed the benefits.

Another thing to note in addition to settling time is the dead zone present within PFDs. For small phase differences between the two input signals the PFD will output narrow pulses. Depending on the size of this pulse, the charge pump may or may not be activated. This is due to the rise and fall times of the output signals. There may simply not be enough time for the output signal to reach a logic one. This results in a miscommunication between the two blocks and the PFD fails to tell the charge pump to go high. For purposes of understanding it can be stated that the loop will only respond to differences in phase greater than

$$Dead\ zone\ edge = \pm\frac{\tau\pi}{T}$$

Where τ is the rise time and T is the reference period [2]. Looking at this equation it can be seen that with a high reference frequency or a greater delay at the output an increase in the dead zone will be expected. This creates problems with locking the PLL, and can create a phase noise increase. Therefore this is an important aspect to remember when choosing a PFD to work with.

Finally, for the purposes of this design, a basic tristate phase detector circuit was chosen as shown in Figure 4.3. The circuit consists of two flip-flops tied to logical one, and an AND gate. In simulation, negative edge triggered flip-flops were used in both Verilog and Cadence. Knowing these facts, it can be understood that when both input clocks are high, at the negative edge, the reset will go high changing the output waveform. To visually show this in practice the state diagram in Figure 4.4 below has been provided. It should also be noted that many examples obtained from both simulations will be shown later within this chapter.

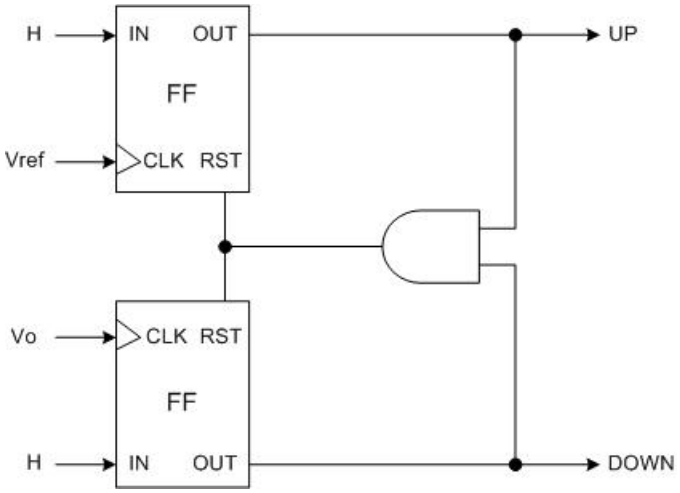

Figure 4.3: Tristate PFD layout

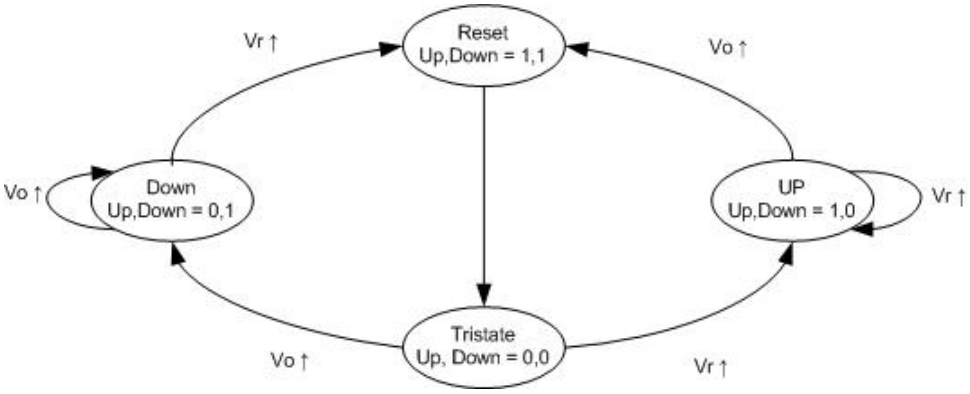

Figure 4.4: Tristate PFD state diagram

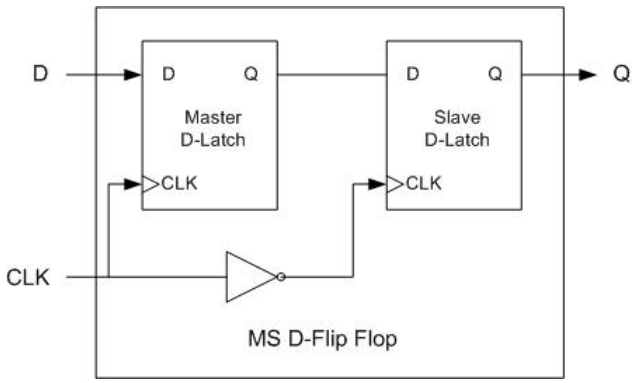

Figure 4.5: MS flip-flop using latches

The implementation of the phase detector is simple enough in Verilog, but Cadence simulation proved to be more difficult. Using 8hp technology and base layer gates the flip-flops needed for Cadence simulation were built. This was done by cascading two latches together by connecting the output of the first to the input of the second and tying all other inputs together which is shown in Figure 4.5.

The output of the second latch was then used as the output of the flip-flop seen in Figure 4.3. From this point in the Cadence phase detector design the only problem left to deal with was the current mode logic (CML) levels of each block. Figure 4.13 shows the block diagram layout of the phase detector within Cadence. To successfully connect the phase detector to the output of a future MMD and to other blocks, a total of five level shifters were needed. These needed components will now be discussed in the following section.

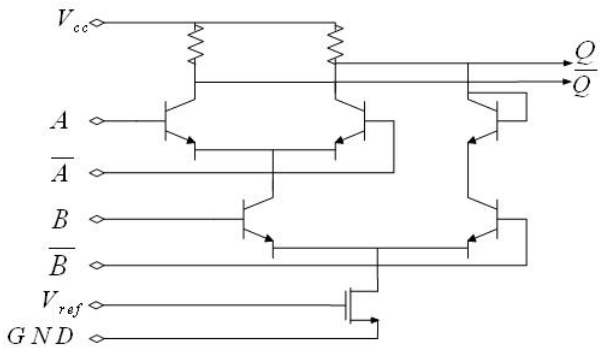

Figure 4.6: CML implementation of an AND gate

4.1.1 CML Combinational Circuits

For high-speed applications and for low switching noise, synthesizers do not always use standard complementary metal oxide semiconductor (CMOS) logic but use CML instead [2]. By assuming that given inputs are square waves it is easy for the user to map out functionality since the transistors will act like switches. At this point three blocks are known to be needed SiGe implementation. These consist of the previously discussed AND gate, latch with reset, and level shifter. From here these three will be discussed in detail. As this thesis progresses through new blocks within the PLL other circuits used will be presented accordingly.

The first and most fundamental block to look at is the AND gate. This gate's functionality is easier to visually understand assuming a square wave and that the transistors act like switches. Figure 4.6 shows the circuit diagram of an AND gate using CML. By observing the gate's truth table it can be seen that this structure is accurate. For all other cases, except when the differential inputs of A and B are logic one, the transistors are not turned on. This gives an output of logic zero which is expected.

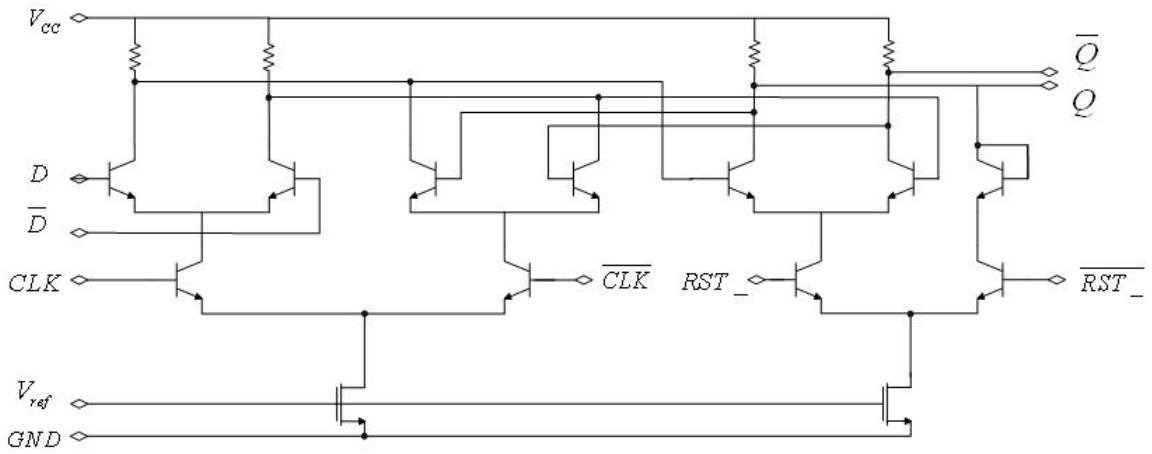

Figure 4.7: CML implementation of a latch with active low reset ([7])

The next CML circuit to look at is the latch with active low reset. There are many different circuits that can be used for this block but the one chosen uses three transistor levels giving the option of a low-supply voltage as shown in Figure 4.7. Since a supply voltage of 2.2 volts was decided on, this was not an option but a necessity.

Finally, as was previously discussed, a level shifter is needed to allow for smooth connectivity between the CML blocks. This is why a separate block with this circuity was needed to connect the architectures block-by-block. Figure 4.8 shows the circuit used to accomplish this task. Looking at the circuit an understanding of its capability can be seen very easily. The inputs (A_m and A_p) are inputted on the top level, dropped one level by two dummy bipolar junction transistors (BJT), and then outputted one level down. The voltage on the top differential inputs are 2.2 and 2.0 volts, which then leave the bottom at 1.3 and 1.1 volts. This block proved very useful and was used throughout the PLL.

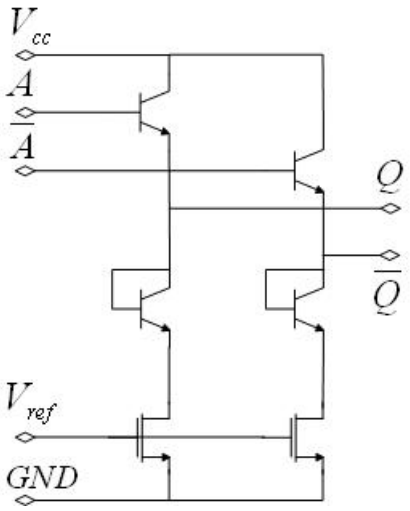

Figure 4.8: CML implementation of a level shifter

4.1.2 Reference Buffer

Now with a general understanding of CML, it was essential to add a reference buffer to the input of the PFD. This reference buffer consisted of a four inverter stage buffer, a CMOS to CML converter, and a feedback filter. This buffer is used to input a crystal oscillators output, which is a CMOS signal, and converts it to a CML differential signal after buffering.

Before jumping into how the reference buffer was built there are two circuits that should be introduced. These consist of a CML inverter and also the CMOS to CML converter. The simplest and first circuit discussed will be the CML inverter. As previously introduced, the inverter circuit needs little explanation since the transistors can be considered as acting switches. Figure 4.9a shows how this circuit was built.

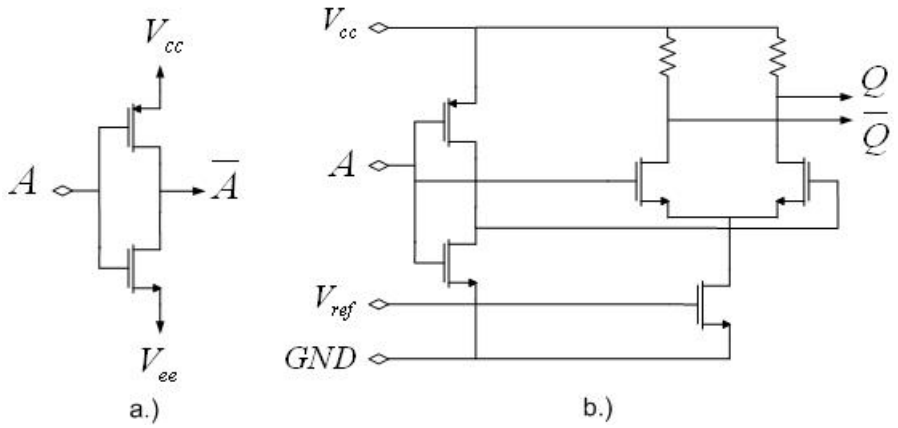

Figure 4.9: CML inverter(a) and a CMOS to CML converter(b)

Next, a CMOS to CML converter was needed, this can be seen in Figure 4.9b. This circuit is fairly simple to follow. If the input is CMOS level at 2.2 volts, then after entering this circuit the outputs (Q_p and Q_m) will be 2.2 volts and 2.0 volts respectively giving the differential CML outputs that were needed.

As stated earlier, CML is better used when inputting square waves. This was acquired by using the four stage inverter buffer. Figure 4.10 shows the resulting reference buffer that was set up to accomplish this task. The inverter portion of the buffer squares off the sine wave, while the CMOS to CML converter gives the differential outputs that were needed. Figure 4.11 shows the input and output characteristics of this buffer. The sine wave represents a typical oscillator input, while the square wave shows the CML square wave output the buffer creates.

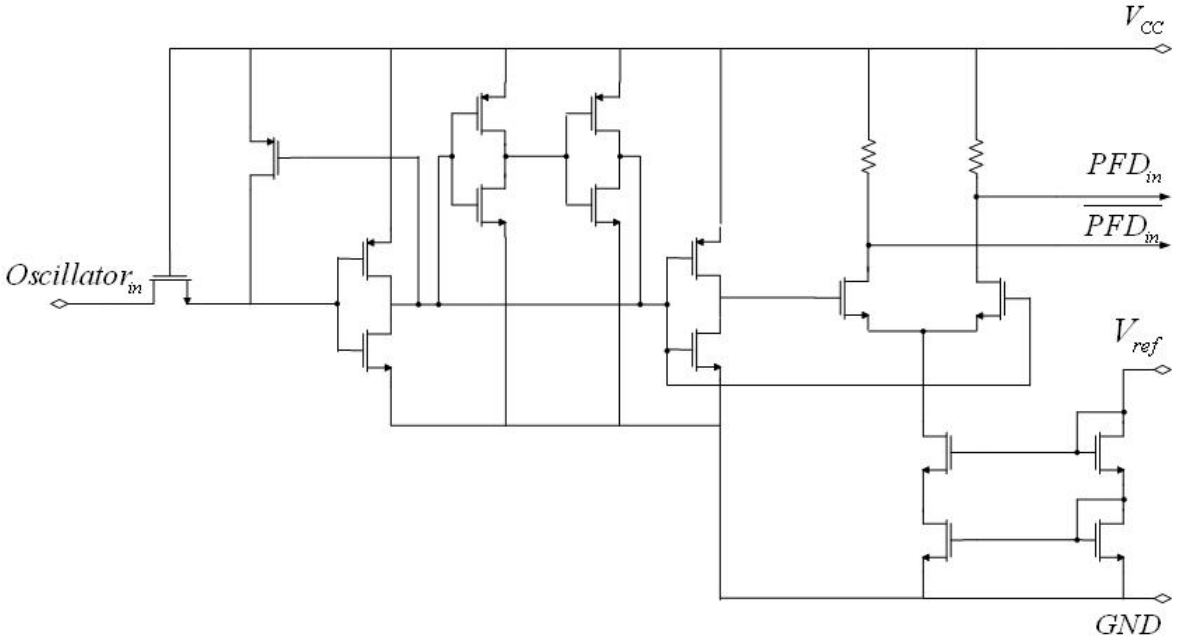

Figure 4.10: PFD reference buffer schematic

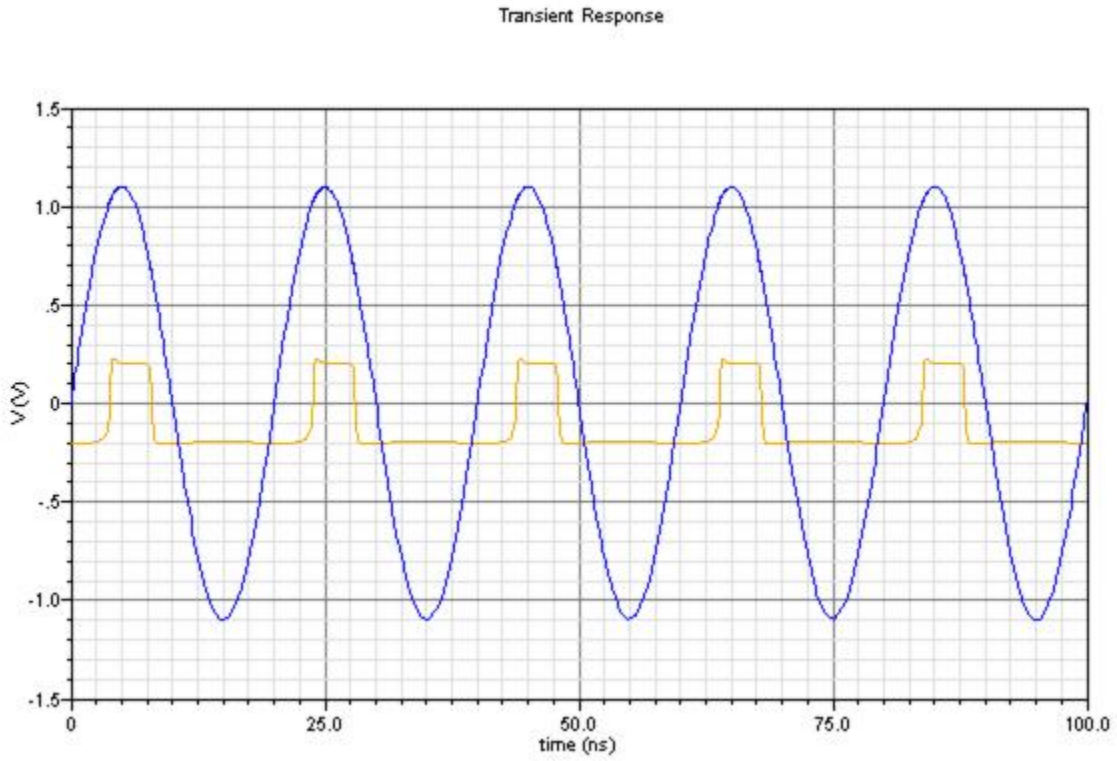

Figure 4.11: Reference buffer input output characteristics

Figure 4.12: Verilog Simulation output results. (a) 180 degrees out of phase (b) in-phase (c) slightly out of phase

4.2 Verilog Simulation

Knowing that Verilog results would be easily simulated and found, this was the starting point. The code used for each block and the testbench used for the simulation can be found within Appendix A. The simulation results can be seen in Figure 4.12. Three variations were used to test different situations. The different variations consist of one-hundred and eighty degrees out of phase, in-phase, and slightly out of phase respectively. Comparing these results with the conceptual view, Figure 4.3, it can be seen that the results match what would be expected.

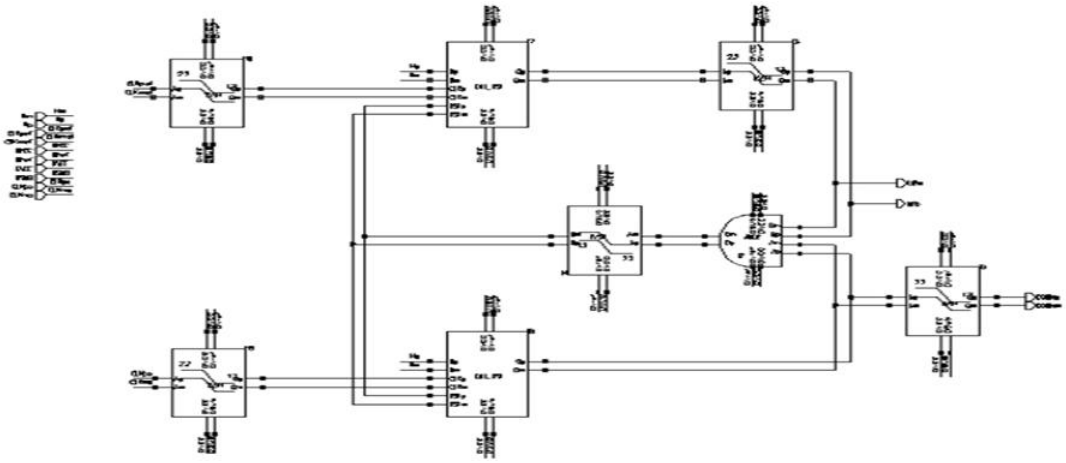

Figure 4.13: PFD schematic diagram in Cadence environment

4.3 Implementation in SiGe Technology

From this point the basis to begin the circuit design and simulation within the Cadence environment was met. Since the design of the PFD was already set up using previously discussed criteria, the circuit was ready to be built within Cadence itself, Figure 4.13, and compare the results to that of the Verilog outputs. Following the layout, conceptually the Cadence phase detector should work the same as the Verilog simulation and proved to do just that. The simulation results from Cadence can be seen in Figure 4.14. These are set up respectively to that of the Verilog simulation output. Comparing the two it was known that the tristate phase detector was successful in both simulation environments. Now the next task was to build a charge pump for use within this PLL.

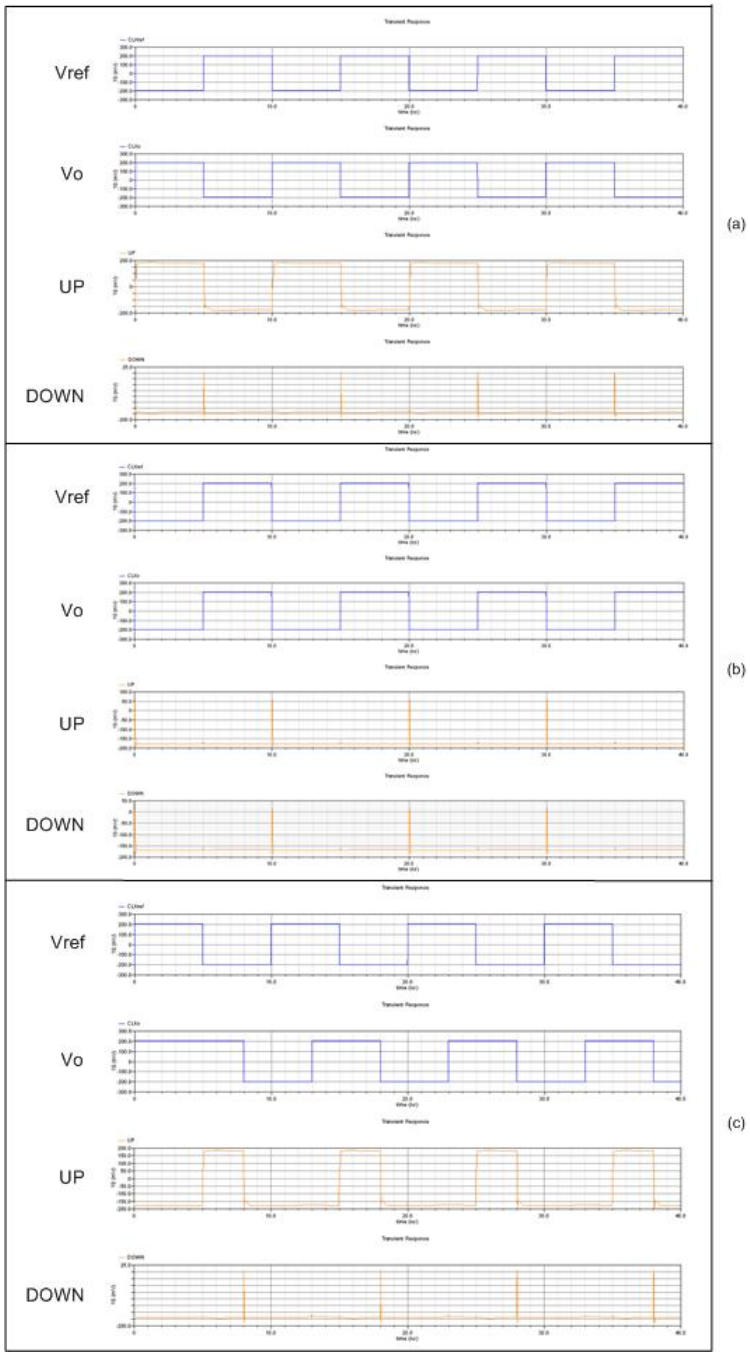

Figure 4.14: Cadence simulation output results (a) 180 degrees out of phase (b) in-phase (c) slightly out of phase

Chapter 5

Charge Pump

5.1 Architectures

The design used for the charge pump can be seen in Figure 5.1. It was constructed using schematics from [2]. As seen in the schematic, bipolar transistors were used for the inputs since they have lower flicker noise and also operate faster as switches. With this schematic the current is transferred to unused resistors when low and are tied to the current source transistors when high.

Transistor W/L ratios were the major factor when building this circuit. Larger W/L ratios are desired to lower the phase noise since noise here can affect the PLL circuit more drastically than other places. Since the inputs to the charge pump were on the second level, level shifters were used to ensure correct connectivity and functionality. This meant the outputs leaving the phase detector needed to be set to 1.3 and 1.1 volts for a logic one. The VCO that would be driving this charge pump also had to be considered, ratios and transistor sizes affect the current flow through the VCO that drives the change in frequency.

Later in the VCO overview it will be seen how the set up of the VCO distinguishes how the output of the charge pump should be polarized. Knowing all of these factors that needed to be considered, the building and testing of the charge pump within the Cadence simulation environment was the next step.

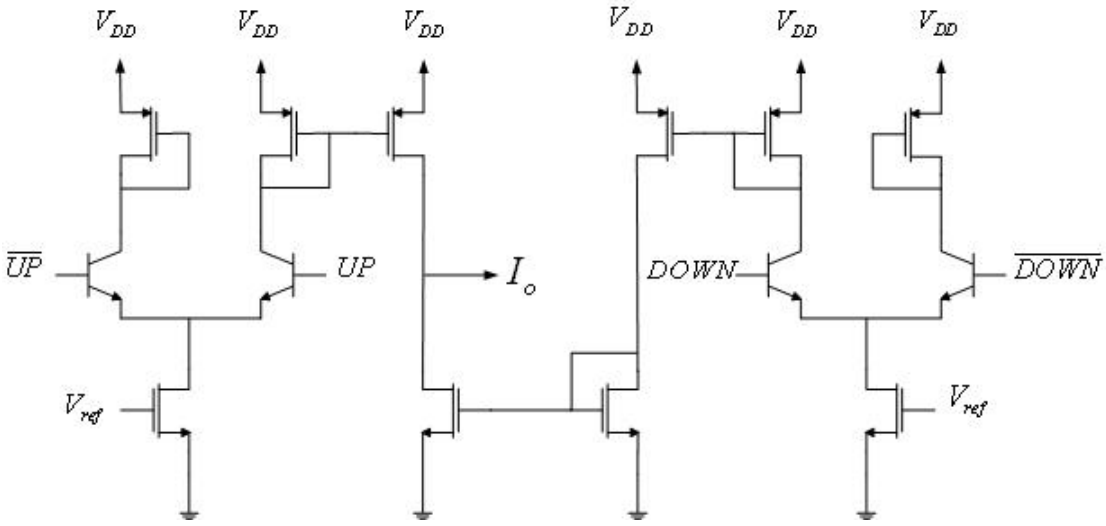

Figure 5.1: Charge pump schematic ([2])

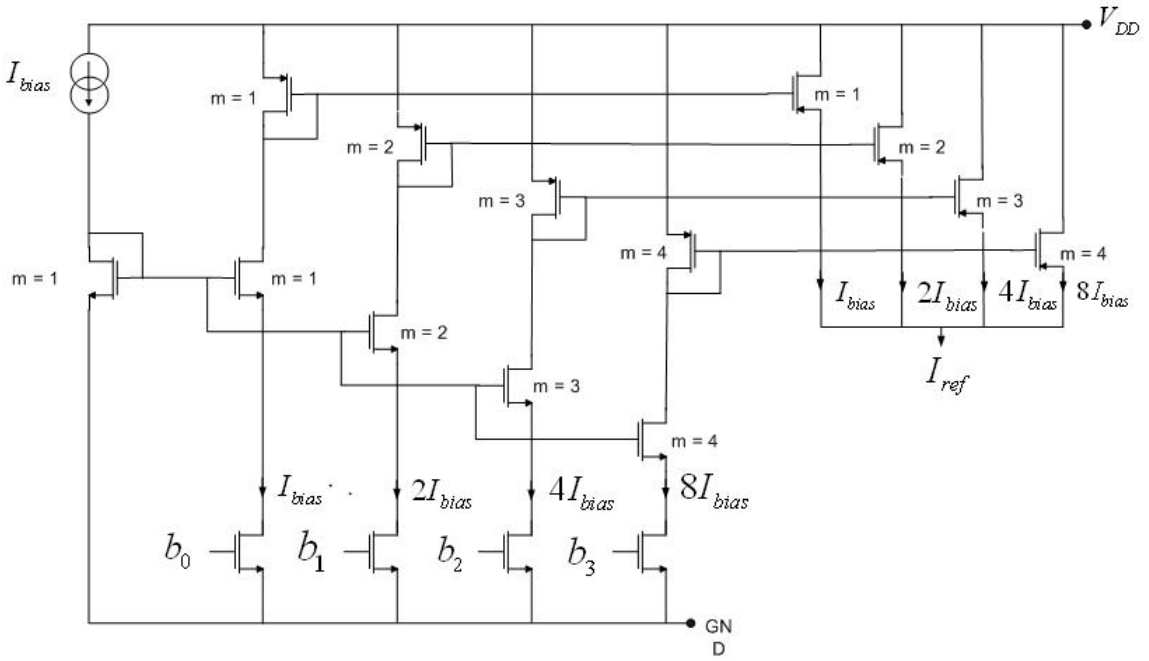

Figure 5.2: Charge pump programmable bias schematic with range from 1 to 15 Ibias ([8])

5.1.1 Charge Pump Programmable Current Source

A valuable addition to a charge pump within a PLL is a programmable current bias. An example of one of these circuits can be seen in Figure 5.2 [2].

The schematic shown gives the user a programmable range of anywhere from a chosen and implemented I_{bias} up to 15 I_{bias}. Having a programmable current bias gives the user the freedom of adjusting the current flowing into the charge pump. This is useful for both adjusting the loop bandwidth and also improving phase noise performance. The latter alone is reason enough to add this circuit to any PLL since, given that a charge pump is present, it is a major contributor to phase noise. Having this circuitry serves as a fail safe since many

measurements made within simulations are not always a perfect representation of what one might get back from manufacturing a chip.

From here a representation of this circuit in equation form is needed to better understand its importance. Due to the simplicity of the circuit it is easy to see that due to the current mirrors within the schematic, the current can be programmed in binary steps. Using this it can be seen that

$$I_{ref} = (8b_3 + 4b_2 + 2b_1 + b_0)I_{bias}$$

5.2 Implementation in SiGe Technology

To test the charge pump properly it was tied to the PFD that had already been tested successfully. Since the phase detector had already been designed with differential inputs and outputs it was easily connected to the charge pump that was designed to input differential signals for use within the test bench. From this point the signals from the previous phase detector tests were used to check the charge pump outputs and make sure it was working properly with the UP and DOWN signals.

For demonstration purposes, the simulation results in Figure 5.3 show the response of the charge pump with given UP and DOWN signals. As expected the charge pump's voltage rises to approximately 2 volts at the rising edge of the UP signal and degrades while the UP input is low. This is what should be expected from the charge pump. From here a filter off chip would be added between the charge pump and the VCO to properly feed the VCO.

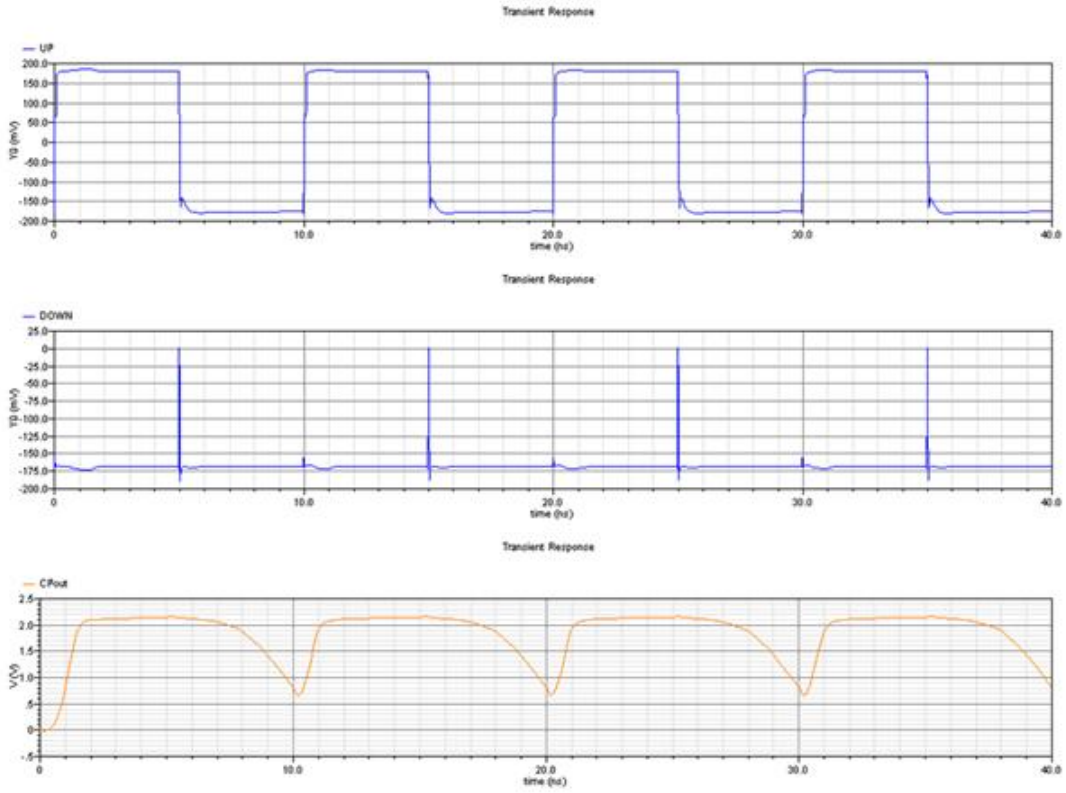

Figure 5.3: Cadence simulation output results of Figure 4.14 (b) showing UP, DOWN, and charge pump outputs respectively.

Chapter 6

Multiple Modulus Divider

6.1 Architectures

As stated earlier a review of the approach and topology used in the PLL's frequency divider block will be discussed. An approach found in [2] is what was used and is referred to as a generic MMD architecture. As stated earlier, many MMD's made today use the cascaded 2/3 dual modulus cell architecture [14]. Here the same approach is used but it will be ended into a divide by P/P+1 dual-modulus cell. This approach saves on both overall die area and also power. In addition, since it is a fractional-N synthesizer, it can achieve a higher reference frequency and a finer step size since it will be constantly swapping the loop division ratio between integer numbers, thus on average dividing by a fractional number [12]. Figure 6.1 shows a block diagram of the architecture.

Knowing the general layout of the MMD wanted for construction it was now imperative to begin finding how the MMD wanted would be laid out. For demonstration purposes a unit step size of (S=1) will be assumed, in the Phase Locked Loop Synthesizer Design

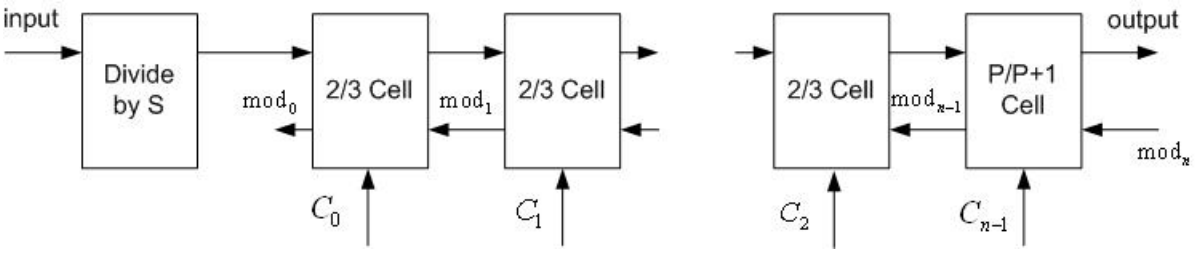

Figure 6.1: Multi-modulus divider block diagram

chapter this will be further discussed. Knowing this the output period is given by

$$T_{output} = (2^{n-1}P + 2^{n-1}C_{n-1} + 2^{n-1}C_{n-2} + ... + 2^1C_1 + C_0)T_{input}$$

T_{input} and T_{output} are the output and input periods respectively and the C terms are the control bits that are user-programmable. In the Integrated Circuit Design for High Speed Frequency Synthesis book the following approach is given and used to find the number of 2/3 cells needed and also the division ratio of P [2].

1. *Assume that the required division ratio is from D_{min} to D_{max}. The division ratio range is $(D_{max} - D_{min} + 1)$.*

2. *If the required range is greater than the minimum division ratio, D_{min}, the MMD is referred to the architecture in [14].*

3. *The implemented MMD range, defined from M to N, can be larger than the required range. Initially set $M = D_{min}$.*

4. *Now the number of cells required becomes $n = \lceil log_2(D_{max} - M + 1) \rceil$ where function $\lceil a \rceil$ denotes rounding a to the nearest integer towards plus infinity.*

5. *The division ratio for the last cell can be found from $P = \lfloor M/2^{n-1} \rfloor$ where function $\lfloor a \rfloor$ denotes rounding a to the nearest integer towards zero.*

6. *If $M/2^{n-1}$ is not an integer, reset $M = P * 2^{n-1}$ and go to step 4.*

7. *If $M = 2^{n-1}$ is an integer, we have to decide recursively whether using a single P/P+1 cell or using a combination of a 2/3 cell and a $\frac{\lfloor P/2 \rfloor}{\lfloor P/2 \rfloor + 1}$ cell will achieve lower current consumption and smaller die size as discussed later.*

8. *The final MMD architecture is thus a combination of stages with*

$$(\frac{2}{3})_1 \rightarrow (\frac{2}{3})_2 \rightarrow ... \rightarrow (\frac{2}{3})_{n-1} \rightarrow (\frac{P}{P+1})_n$$

If we are using all 2/3 cells then the total number of cells required is $[log_2(D_{max} + 1] - 1]$.

Having this approach to finding and building the general MMD that was wanted, it was easy to "plug and chug" to find the resulting architecture. The first step to do this was to find the division range wanted for the MMD. Knowing the operating frequency needed for the PLL (13 Ghz) it was found that X-band radar transceivers with this technology were required to use a division range from 131 to 154 with unit increment [12]. To show how this is done the approach purposed above will be revisited and values known will be inserted to find the resulting architecture.

1. Number of divisor steps:

$$D_{max} - D_{min} + 1 = 154 - 131 + 1 = 24$$

2. Minimum number of cells required:

$$N = log_2(D_{max} - M + 1) = log_2(154 - 128 + 1) = 5$$

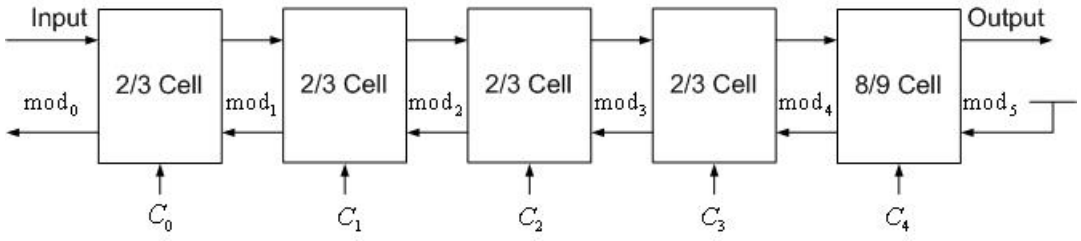

Figure 6.2: MMD architecture for PLL synthesizer design

3. Division ratio of the last stage:

$$P = M/2^{n-1} = 8$$

4. Division ratio can be programmed in the range of:

$$2^4 * 8 + 2^4 * C_4 + 2^3 * C_3 + 2^2 * C_2 + 2^1 * C_1 + C_0 = 128 \sim 159$$

After this process the architecture of the MMD was found, and can be seen in Figure 6.2. Due to step two five cells were needed, and due to step three the P value found gave a resulting 8/9 dual modulus divider for the MMD to end in.

From this point both the final architecture needed and the components within this architecture were accounted for. Now a review of the components individual architectures beginning with the 2/3 cell will be reviewed. As discussed earlier within the PFD chapter CML has been chosen for its high-speed operation. As a result the previously presented CML gates were useable and layouts presented in [2] were used to construct needed cells. Figure 6.3 shows the architecture used for the 2/3 block within the MMD.

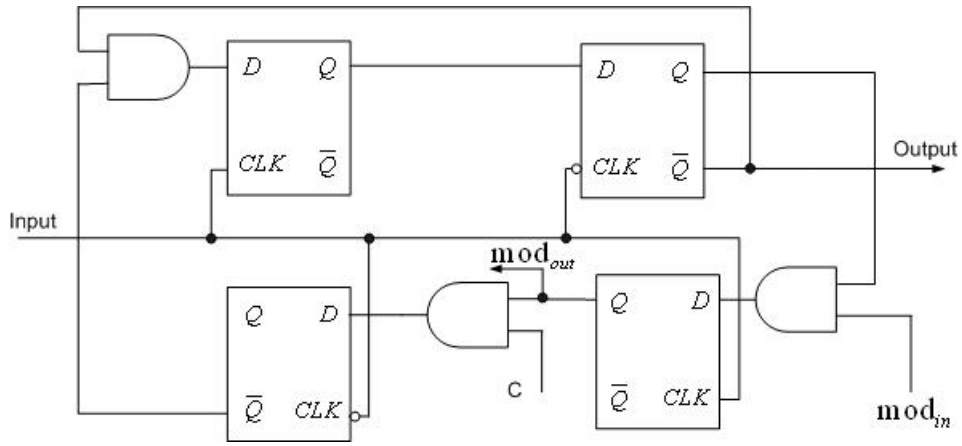

Figure 6.3: Block diagram of a divide by 2/3 cell with mod control ([9])

The divide by 2/3 cell divides the input frequency by either two or three depending on the mod_{in} and C inputs. As can be seen by the block diagram, the output period is twice that of the input unless mod_{in} and C are both logic high at which point the output period will be three times the input. The resulting output, mod_{out}, is at the same time period as the output when mod_{in} is equal to one but with a differing duty cycle. Since the resulting block diagram shown in Figure 6.2 is known, it was important to note how it would affect the connected blocks. For this architecture the end P/P+1 cell must be equal to logical one resulting in all other mod_{in}s being high for one clock cycle during a simultaneous cycle. This results in an extra input cycle at the output giving an instantaneous division ratio of three for that cell, given the control input (C) is equal to logical one [12].

The final block needed for the architecture was the 8/9 cell seen in Figure 6.4. The 8/9 cell follows the same logic as the 2/3 cell and is simply extended for the P/P+1 cell.

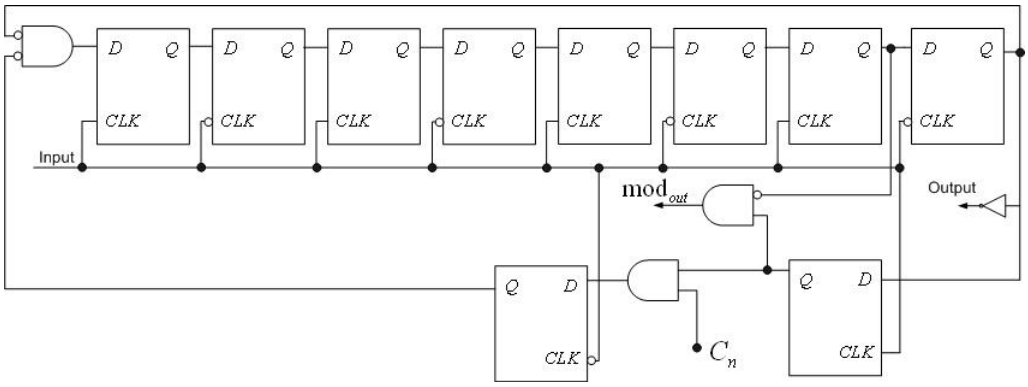

Figure 6.4: Block diagram of a divide by 8/9 cell ([10])

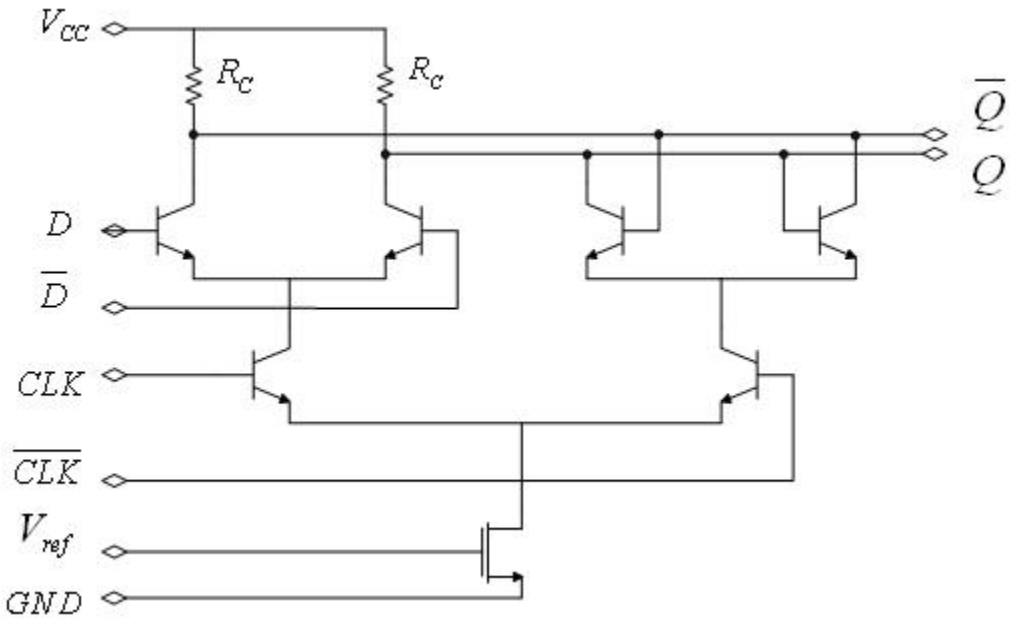

Figure 6.5: CML implementation of a latch

6.2 Implementation in SiGe Technology

To implement this in SiGe technology using the Cadence simulation environment, each individual component needed to be built using CML. As seen earlier in this paper, an AND gate has already been introduced and reviewed. This leaves the latch as the only other immediate component needed for implementation. Earlier a latch with active low reset was presented for use within the PFD, but since the reset option wasn't needed, a separate block was needed. The latch schematic used can be seen in Figure 6.5. As expected, the circuit remains more or less unchanged but with less components.

Using this latch the 2/3 cells and also the 8/9 cell was ready to be built. Before doing this the drive current was adjusted for the cells within the MMD to save on power. Since the speed is reduced by each cell the drive current was decreased cell by cell. To show how and why this was done, Table 6.1 [12] has been provided.

Table 6.1: Maximum input frequency and drive current for each cell

Cell	Fin (GHz)	Drive Current (μA)
0 (2/3)	13.84	500
1 (2/3)	6.92	250
2 (2/3)	3.46	125
3 (2/3)	1.73	125
4 (8/9)	0.865	125

With both the components built and drive current distribution set up the cells and ultimately the MMD itself were ready to be built. Using Cadence, this was done in a block-by-block fashion using both Figures 6.3 and 6.4 as a basis. After building each individual cell and compensating for mismatched voltage logic levels using the previously discussed level shifters (Figure 4.8), the cells were ready to be connected together to construct the MMD. This was done using the architecture seen in Figure 6.2.

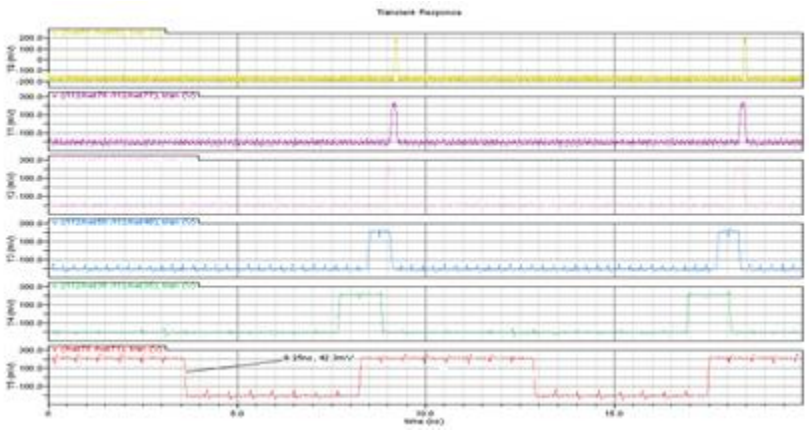

Figure 6.6: Simulated MMD output with mod outputs for divide by 128 with input at 13.84 GHz ([12])

Figure 6.6 shows the resulting mod_{out} signals for the successfully simulated MMD with a 13.84 GHz input signal and a divider ratio programmed as 128. With this set up the MMD gives an output frequency of 108.125 MHz. With successful test results the next step was to move on to the VCO block.

Chapter 7

Voltage Controlled Oscillator

VCOs are at the heart of most frequency synthesizers in some form or another. When choosing a VCO there are three important aspects of its operation that need to be considered consisting of phase noise, power consumption, and its tuning swing. The task of choosing which VCO to use within the PLL was chosen and built by William Souder. For a complete overview of the PLL the VCO architecture he chose will be discussed and a schematic of his final VCO will be presented.

7.1 Architecture

To get a basic understanding of the VCO the $-G_m$ architecture will be looked at in order to gain a basic understanding of its properties. This type of VCO was chosen since the final VCO used within the PLL was an improvement on this basic circuit.

Figure 7.1 shows the basic structure of the $-G_m$ oscillator. A nice aspect of this $-G_m$ oscillator is its simplicity. By simple inspection the input impedance of the oscillator can be seen as

$$Z = \frac{-2}{g_m}$$

This then leads one to the condition for oscillation found in [2]. This condition is dependent of r_p, the parallel resistance of the resonator, and can be seen here

$$g_m > \frac{2}{r_p}$$

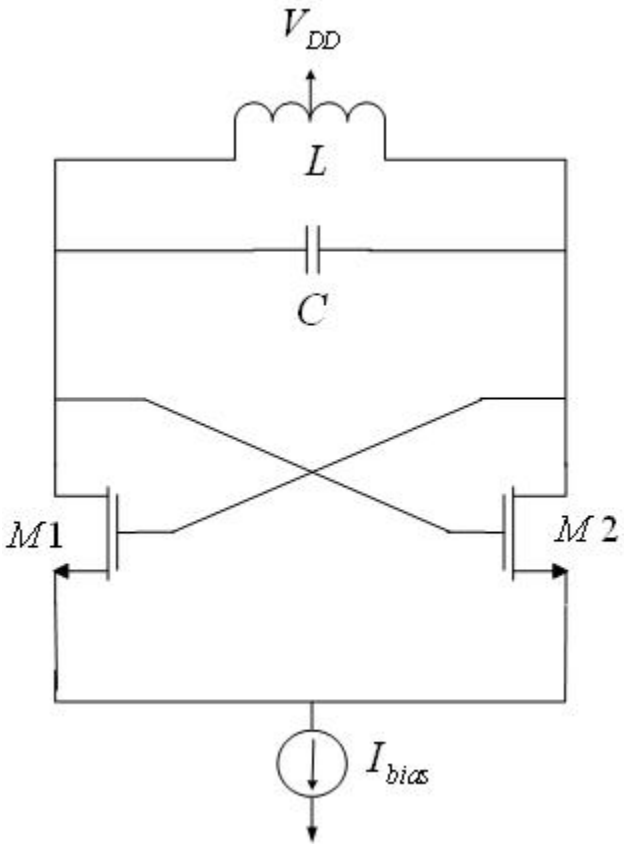

Figure 7.1: A basic -Gm oscillator

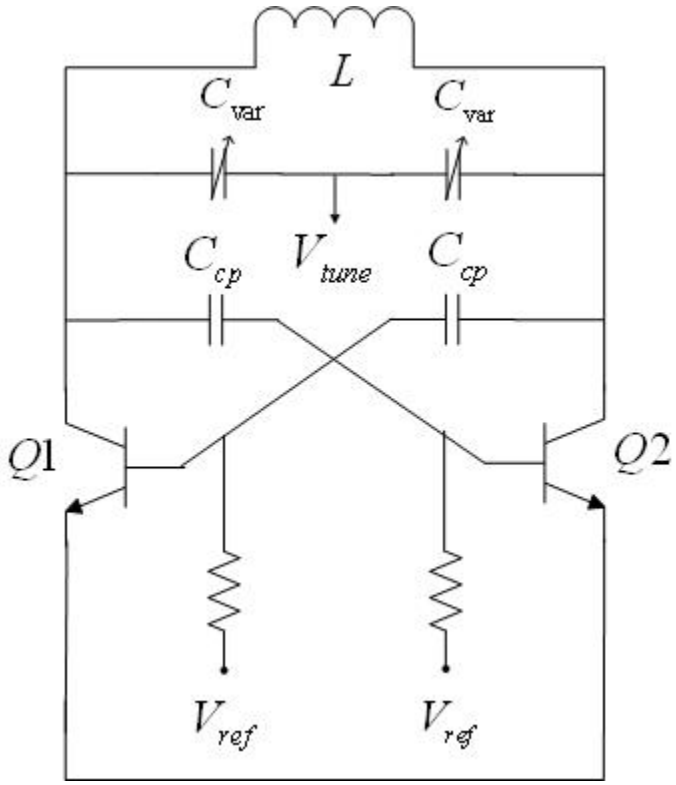

Figure 7.2: Cross-coupled -Gm oscillator ([13])

With this basic understanding of important aspects of the $-G_m$ oscillator reviewing the cross-coupled $-G_m$ oscillator is the next step. This oscillator is the one that was chosen for use in the PLL presented here. Looking at Figure 7.2 it can be seen how this circuit compares to the previously discussed oscillator. This VCO was chosen for its wide tuning range and good phase noise characteristics. These characteristics were improved by the addition of the varactors and also the coupled capacitors.

Chapter 8

Phase Locked Loop Synthesizer Design

All of the SiGe technology implementation of individual blocks have been presented, the task of tieing the PLL together is the next step. Some additions to the blocks discussed previously had to be made to make it work effectively. Figure 8.1 shows the resulting block diagram of the PLL. This diagram will now be discussed on a block-by-block basis.

8.1 Block Layout and Descriptions

As the entirety of the PLL being constructed has been moved through, each of the separate building blocks have been presented in detail. In addition to this, each have been tested separately and achieved the outputs that were wanted. The next task to undertake was tying all these blocks together. As can be seen in the final PLL layout in Figure 8.1, it was not as simple as plugging each block into the next. The additions needed will be looked at and reasons why they were added will be discussed.

For logistics reasons the PFD will be the beginning point and each block added will be discussed as movement around the loop takes place. The first block, not previously discussed, is the VCOs output buffer. This buffer was needed for a couple of reasons. The output of this buffer drives both the output of the PLL and the input to the MMD. If the VCO was connected directly to the MMD and the output pin the VCO's output would simply be unable to drive all of that circuitry. Another important reason for this addition is phase noise improvement. Loading the VCO causes mass amounts of phase noise, the buffer is used to keep this from happening.

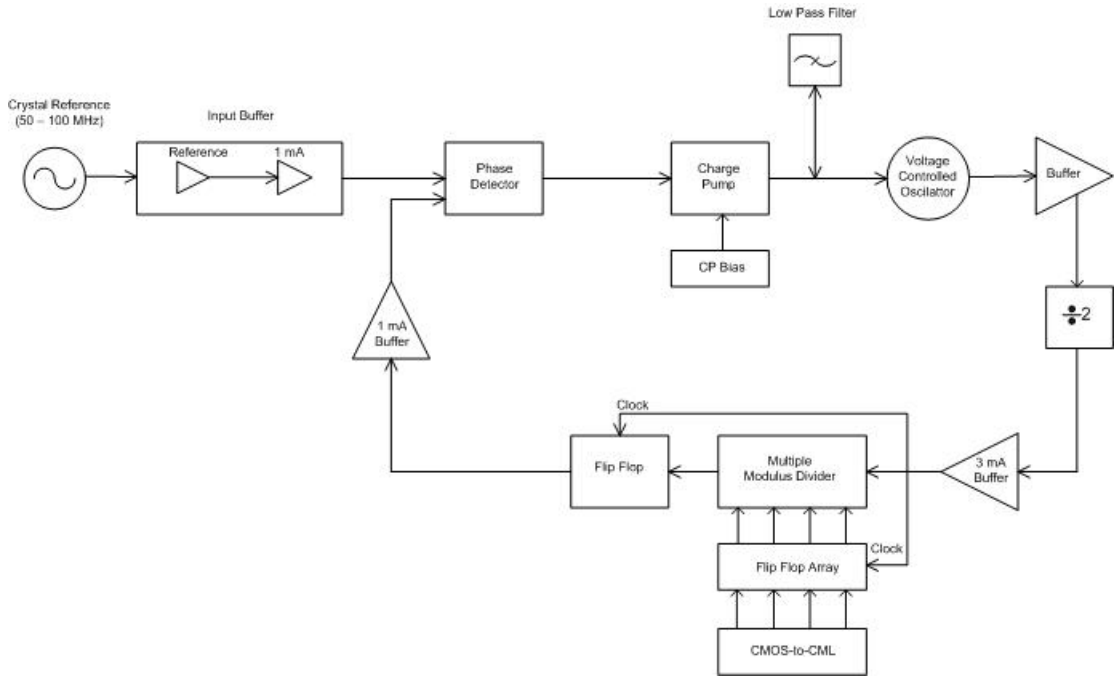

Figure 8.1: Block diagram of a phase locked loop RFIC implemented in SiGe technology

The next unknown block encountered within the loop is the divide by two block. This block was briefly discussed in the MMD chapter and can be seen in Figure 6.1 under the title of "Divide by S." This was an important addition to the PLL for purposes of lowering the output of the VCO. The current coming out of the VCO is simply too high for the beginning 2/3 cell within the MMD. Using the divide by two cell the output is scaled down to a manageable level.

Following the loop the next stop is the three milliampere buffer. This buffer was needed to boost the VCO output enough to drive both flip-flop arrays and also the MMD.

Moving on, the next block considered will be the CMOS to CML bank for the MMD. As discussed earlier, the MMD will be user-programmable by using control bit inputs. These inputs needed to be converted from CMOS inputs to CML inputs. Using the previously discussed circuitry found in Figure 4.9b a bank was constructed for this task to be met.

The next additions needed were the flip-flop arrays. These flip-flop arrays and the MMD are all running on the same clock from the output of the VCO. This had to be done for synchronization purposes. Without this addition, the outputs from the MMD would become a mess of signals without contributing to the purpose of the PLL.

Finally the last addition of the loop is reached, the one milliampere MMD output buffer. This was needed to drive the input of the PFD. Without this the logic the PFD could become inaccurate and results could vary.

Chapter 9

Conclusions

This thesis has covered the entire process of designing a PLL for a system. To begin with the basic structure of a typical PLL was presented. Looking at this structure the process of choosing a synthesizer for use within a PLL was shown, which was the fractional-N for this case. The next step taken was choosing what type of circuits to be used within the PLL, for the purposes presented here CML was that choice. From this point it was simply choosing, building, and testing the architectures for each individual component needed for the PLL. Finally after all of these tasks were completed the last step was tying all of the components together and identifying where additions needed to be made.

The PLL presented here has been fabricated in SiGe technology as seen in Figure 9.1 with dimensions of 2.4mm by 1mm. All the components have been successfully tested individually within the Cadence environment, and have been presented. In addition to this there have also been successful open loop simulations presented. From here the next step is to complete tests on the fabricated chip and identify where improvements are needed.

Figure 9.1: Phase locked loop RFIC layout in Cadence environment (2.4mm x 1mm)

There is much work to be done consisting of anything from using more complex blocks to improve settling time, or even something as simple as increasing W/L ratios to improve phase noise. The writer of this thesis hopes to continue work on this PLL after graduation while an employee of USSMDC.

Bibliography

[1] Kevin MeClaning, Tom Vito, "Radio Receiver Design" Noble Publishing Corporation, 2000.

[2] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis" Artech House, Inc., 2006.

[3] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 24)" Artech House, Inc., 2006.

[4] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 48)" Artech House, Inc., 2006.

[5] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 45)" Artech House, Inc., 2006.

[6] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 185)" Artech House, Inc., 2006.

[7] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 140)" Artech House, Inc., 2006.

[8] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 219)" Artech House, Inc., 2006.

[9] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 167" Artech House, Inc., 2006.

[10] John Rogers, Calvin Plett, and Foster Dai, "Integrated Circuit Design for High-Speed Frequency Synthesis (p. 174)" Artech House, Inc., 2006.

[11] "The ARRL Handbook for Radio Communications 83rd edition" The American Radio Relay League, Inc., 2005.

[12] Mark Ray, William Souder, Marcus Ratcliff, Foster Dai, J.David Irwin, "A 13Ghz Low Power Multi-Modulus Divider Implemented in 0.12μ SiGe Technology"

[13] John Rogers, David Rahn, Calvin Plett "A Study of Digital and Analog Automatic-Amplitude Control Circuitry for Voltage-Controlled Oscillators" IEEE Journal of Solid State Circuits, VOL. 38, NO. 2, pp. 352-356, February 2003

[14] Cicero Vaucher, Igor Ferencic, Matthias Locher, Sebastian Sedvallson, Urs Voegeli, and Zhenhua Wang "A Family of Low-Power Truly Modular Programmable Dividers in Standard 0.35-μm CMOS Technology" IEEE Journal of Solid State Circuits, VOL. 35, NO. 7, pp. 1039-1045, July 2000

Appendices

Appendix A

Verilog Program Code

A.1 Flip-Flop Code

```
module dffa (q,d,clk,rst);

output q;
reg q;
input d;
input clk;
input rst;

always @(posedge clk or negedge rst) begin: _dffa_logic
    if ((rst == 0)) begin
        q <= 0;
    end
    else begin
        q <= d;
    end
end

endmodule
```

A.2 Test Bench

```
‘timescale 1ps / 1ps  // ref_time_unit/precision
module testbed();

  reg d,d2,clk,clk2;
  wire q,q2,reset;

  dffa1 A1(q,d,clk,reset,);
  dffa1 A2(q2,d2,clk2,reset);
  nand A3(reset,q,q2);

initial
  begin
      d = 1’b1;
      d2 = 1’b1;
  end

initial
  begin
        clk = 1’b1;
      #25000 clk = 1’b1;
      #25000 clk = 1’b0;
      #25000 clk = 1’b0;
      #25000 clk = 1’b1;
      #25000 clk = 1’b1;
      #25000 clk = 1’b0;
      #25000 clk = 1’b0;
      #25000 clk = 1’b1;
      #25000 clk = 1’b1;
      #25000;
  end
initial
  begin
        clk2 = 1’b0;
      #25000 clk2 = 1’b0;
      #25000 clk2 = 1’b1;
      #25000 clk2 = 1’b1;
      #25000 clk2 = 1’b0;
```

```
      #25000 clk2 = 1'b0;
      #25000 clk2 = 1'b1;
      #25000 clk2 = 1'b1;
      #25000 clk2 = 1'b0;
      #25000 clk2 = 1'b0;
      #25000;
  end

initial
  begin
   $monitor ($time,,, "d=%d d2=%d clk=%d clk2=%d reset=%d q=%d q2=%d",
   d,d2,clk,clk2,reset,q,q2);
   #250000 $finish;
  end

endmodule
```