A Stochastic Time-to-Digital Converter for Digital Phase-Locked Loops

by

Kerem Ok

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented August 26, 2005
Commencement June 2005

ACKNOWLEDGEMENTS

I would like to thank my advisors Dr. Un-Ku Moon and Dr. Karti Mayaram for their valuable guidance over the course of the last two years. I would also like to thank members of my thesis committee for their useful feedback.

Thanks to Pavan Hanumolu and Gil-Cho Ahn for always being supportive and providing helpful insight into design issues. Thanks to everyone else in lab for being open to answering questions and putting up with me during soccer.

Thanks to AFRL for making my assistantship possible by funding this research project.

Thanks to Tufan for always being ready to go lifting or to Platinum (Thanks Dave) or Senor Sam's with me, acting as my airport shuttle, doing house or driving around Corvallis. Many thanks to Cumhur for helping a lot with organizing the defense and always saying yes to sushi. Thanks to both for late night Shari's runs, daily doses of Dutch Bros./Starbucks and cigars on the roof or at the MU quad.

Thanks mom and dad, I hope I can raise my kids the way you raised me.

Finally, thanks Bebegim for always being patient with me, always supporting me and always being so sweet.

TABLE OF CONTENTS

Page

1. INTRODUCTION 1

2. FREQUENCY SYNTHESIS 3

2.1. Frequency Synthesis Using PLLs 4

2.1.1 Synthesis Using Analog PLLs 5
2.1.2 Synthesis Using Digital PLLs 8

3. STOCHASTIC TIME-TO-DIGITAL CONVERTER 10

3.1. Theory of Operation 10

3.2. Dynamic Range vs. Resolution Trade-Off 12

3.3. System Level Simulations of the STDC 13

3.3.1. Effect of Changes in Offset Distribution 14
3.3.2. Effect of Different Input Signal Rise-Times 14

3.4. Implementation 17

3.4.1. Delay and Slew Control 17
3.4.2. Latch and Sampling Flip-Flop 18
3.4.3. 84:7 Encoder 20
3.4.4. Master-Slave Flip-Flop 23

3.5. Layout 25

3.6. Circuit Level Simulations 26

4. CONCLUSION 27

BIBLIOGRAPHY 29

LIST OF FIGURES

Figure Page

2.1. Generic PLL Block Diagram 4

2.2. Charge-Pump PLL Block Diagram 5

2.3. Digital PLL Block Diagram 8

3.1. Fine TDC Conceptual Diagram 11

3.2. Simulink Model of the Stochastic TDC 13

3.3. Transfer Characteristics of the STDC for Different Offset Voltages 15

3.4. Transfer Characteristics of the STDC for Different Rise Times 16

3.5. Delay and Rise Time Control Circuit 18

3.6. Arbiter Schematic 19

3.7 7-to-3 Encoder 21

3.8. 28-to-5 Encoder 22

3.9. 84-to-7 Encoder 22

3.10. Master-Slave Divide-by-2 Flip-Flop 23

3.11. Overall Block Diagram 24

3.12. Layout of the STDC Core 25

3.13. Transistor Level Simulation Results for Different Rise Times in Spectre 26

STOCHASTIC TIME-TO-DIGITAL CONVERTER

1. INTRODUCTION

The ever-increasing demand and continuously expanding market for wireless communication systems has fueled research in new and established architectures for improving quality and reducing cost. Developments in IC processing technology have enabled shrinking dimensions and made CMOS a viable alternative to the more traditional RF processes using GaAs or BJTs. This has enabled integration of digital baseband processing with the analog front end on one low cost chip. [1]

One of the most challenging steps in this integration has been realizing low noise and low power frequency synthesizers, which are used in every transmitter/receiver subsystem. Among other methods, indirect synthesis using phase-locked loops (PLLs) has received great attention due to its desirable noise characteristics. A whole spectrum of analog, semi-digital and fully-digital PLL implementations have been presented [2, 3, 4, 5]. Fully-digital implementations show the greatest promise with regards to low power operation and the possibility of full integration with the baseband.

This work presents a stochastic time-to-digital converter (STDC) that may find use with a digital phase-locked loop (DPLL). Most DPLL implementations to date realize the time-to-digital converter (TDC) using a chain of delay cells, where the smallest quantization step is defined by the unit delay used in the chain. Therefore the resolution using this method is limited by the smallest gate delay defined by the process technology used in the implementation. As the resolution of the TDC directly affects the spurious noise performance of the DPLL, it is desirable to make the quantization step in time digitization as small as possible. The STDC provides a means of increasing the resolution of time-to-digital conversion beyond an inverter delay, by exploiting random mismatch in a number of latches.

A brief overview of the crucial need for low noise and low power frequency synthesis in today's wireless communication systems is provided in Chapter 2. The different methods of frequency synthesis are also briefly reviewed in this section.

Chapter 2 also presents a brief background on fully analog, semi-analog and fully digital PLL implementations used for frequency synthesis. The role of the STDC within an ADPLL and the improvements it provides over conventional methods is explained. Chapter 3 introduces the concept of stochastic time digitization and gives an overview of the architecture and building blocks. Chapter 4 presents some concluding remarks.

2. FREQUENCY SYNTHESIS

Every wireless communication system is based on transmitting data between isolated transceivers by modulating a carrier signal as per the information content to be transmitted. The generation of this carrier signal is one of the most challenging aspects in building transceivers geared towards wireless standards such as WLAN or GSM. This is due to strict noise requirements as well as a high desired frequency of operation with low power consumption.

The noise performance of a synthesizer is most often characterized by phase noise, which is a measure of the spectral purity of the system output. This topic has been extensively studied in literature and is an active area of research [6,7]. It is known that all synthesizers exhibit phase noise, and that this phenomenon degrades signal transmission quality by allowing noise at other frequencies to fold into the band of interest. These complications are well described in the literature and will not be explained in detail here. The significant point is that reduction of phase noise is crucial for increasing transmission rates while keeping bit-error rates acceptable.

Frequency synthesizers realized to date can be grouped into three categories with respect to the techniques used: direct synthesis, indirect synthesis and hybrid synthesis. Direct synthesis refers to the use of a look up table (realized by a ROM) along with a digital-to-analog converter (DAC) and a smoothing filter to produce a desired frequency [8,9]. Although this architecture is capable of very low settling times, it is limited to low frequencies (mainly due to the DAC) and is power hungry. Indirect synthesis is realized by locking to a low frequency crystal and producing some multiple of the input frequency at the output through the use of a negative feedback loop [10]. The popular PLL architecture falls in this category. Hybrid realizations use a combination of these two techniques. In this thesis, the focus is on methods of indirect synthesis. Specifically, an all-digital PLL implementation is considered.

2.1. Frequency Synthesis Using PLLs

A generic PLL block diagram is provided in Fig 2.1. The PLL is comprised of a phase detector (PD), a loop filter (LF) and a voltage-controlled oscillator (VCO) in the forward path, and a frequency divider with a division ratio of N in the feedback path. The PD generates a signal proportional to the phase error between the divided VCO signal and the reference clock. The loop filter processes this error signal to produce a control signal for the VCO and to drive the average phase error at the PD inputs to zero through the negative feedback loop. This implies a phase and frequency locking condition at this point, which in turn means that the VCO output frequency is N times the reference frequency. Hence a higher frequency output signal is obtained by indirect multiplication of a low frequency input signal.

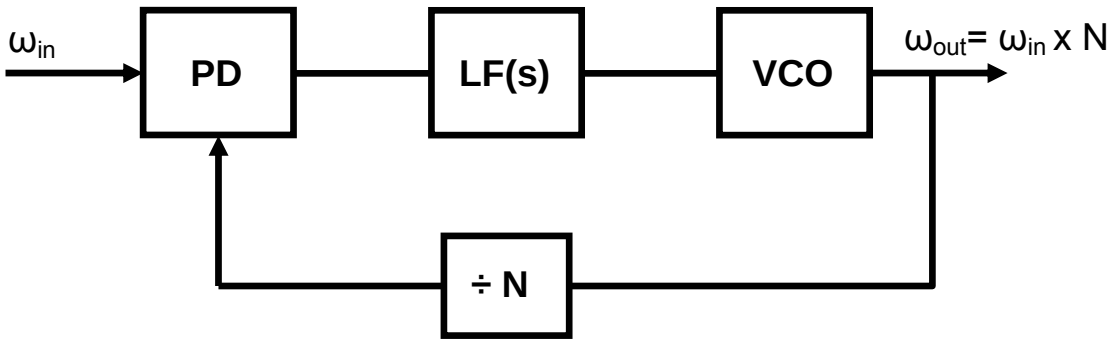


Fig. 2.1 Generic PLL block diagram.

2.1.1. Synthesis Using Analog PLLs

A commonly used architecture for an analog PLL implementation is shown in Fig 2.2. This architecture incorporates a sequential PD, a charge-pump (CP), an analog LF and an analog oscillator in the forward path. The PD generates up and down pulses whose widths are proportional to the phase error depending on whether the VCO is lagging or leading the reference ω_{in}. These pulses are converted to current in the CP, and the current is then integrated onto the loop filter capacitors to produce the VCO tuning voltage. Looking at the PLL in the phase domain, it can be seen that the VCO introduces a pole at DC, while the LF introduces two poles and one zero. The lower frequency pole is realized by C_1 and produces integral control, while the zero is realized by the resistor and provides proportional control. C_2 is used as a ripple bypass capacitor, which creates a third pole. Ripple bypassing enables acceptable spurious performance at the synthesizer output. The zero is required to keep the system stable by moving the second pole away from DC and reducing the phase shift generated by the three poles in the loop. A ring VCO based PLL is shown in Fig 2.2 and the VCO can be replaced with an LC VCO if desired.

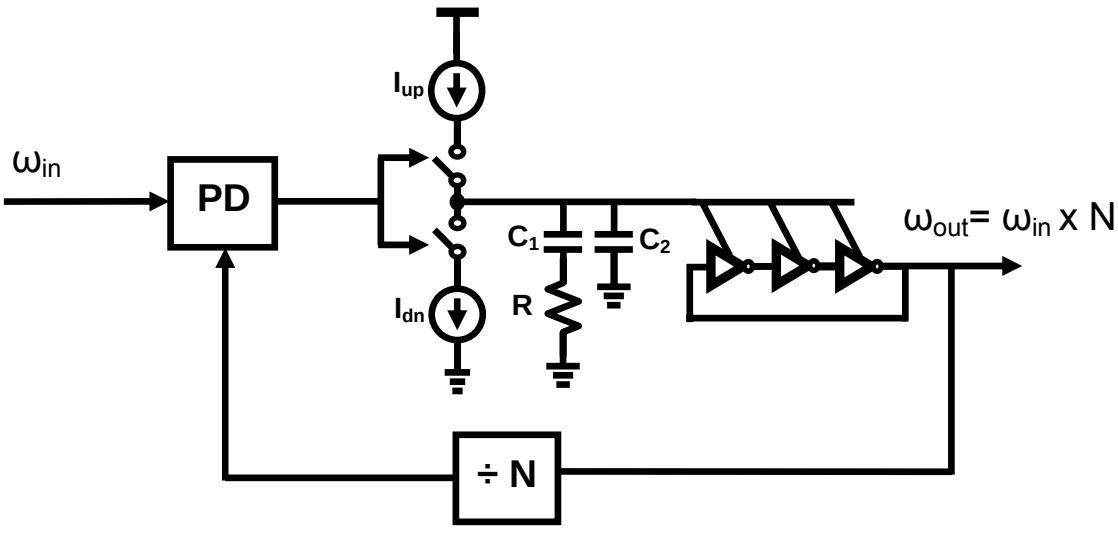


Fig. 2.2 Charge-pump PLL block diagram.

The architecture as shown in Fig 2.2 is capable of generating an integer multiple of the input reference frequency at the output. This creates a fundamental trade-off between noise performance and channel selectivity for this architecture. Since the input frequency is generated by a crystal, it cannot be increased arbitrarily, which places an upper limit on the loop bandwidth, dictated by stability requirements first highlighted in [11]. A high bandwidth is desirable for high rejection of VCO noise as well as for fast small-signal settling performance. However, as the reference crystal frequency is increased to allow for higher bandwidth, the minimum resolvable channel width is also increased because of integer multiplication. For the more stringent standards of today's wireless protocols, these constraints are extremely undesirable.

To overcome these problems, fractional-N techniques have been proposed, where the feedback divider is capable of finer division beyond integer multiples of the reference [12]. This structure ideally enables attaining high bandwidth and high channel resolution simultaneously. A higher frequency crystal can now be used, since the desired channel resolution can be realized through fractional division. Moreover, the loop bandwidth can be increased while keeping the loop stable under all PVT conditions. This is an indispensable feature for current wireless standards as the demanding settling time, noise and channel selectivity requirements make it impossible to meet these specifications with integer-N architectures while keeping cost reasonable.

Fractional division is realized on an average fashion by dividing by different integer numbers over different fractions of a reference period. Because of the periodic nature of changes in the integer division ratio for attaining fractionality, this architecture suffers from poor spurious response. The periodicity is analogous to the problem in integer-N architectures, with the added problem of the spurs appearing at a lower offset frequency (the fractional frequency) from the carrier.

Earlier solutions to this problem included *phase-interpolation* [12] whereby the phase error contained in the accumulator in the feedback path is converted into an analog voltage by a DAC and subtracted from the phase detector output. While in theory this architecture should remove the spurs entirely, in practice, mismatch and

other non-idealities in the DAC result in imperfect cancellation at the phase detector output. Due to its high complexity and associated cost, this technique is unfeasible except in test equipment.

More recently, this problem has been almost exclusively resolved through the use of $\Delta\sum$ techniques to randomize the division and represent the desired ratio again in an average fashion [13, 14, 15]. In this architecture, the $\Delta\sum$ receives a channel select word as its input, and outputs a bit stream with an average value that is equal to the input word. The output bit stream controls the division ratio such that the desired fractional ratio is obtained while avoiding periodicity. With this approach, the energy of the spurious tone in the output spectrum is distributed into the noise floor, thus improving the spurious performance greatly. In reality, the bit stream coming from the $\Delta\sum$ is not completely random, and its statistical properties depend on the $\Delta\sum$ order. In general, a higher order $\Delta\sum$ loop implies whiter quantization noise and smaller spurs [16].

An important consideration in this architecture is setting the loop bandwidth carefully such that the $\Delta\sum$ noise transfer function corner frequency is above the loop filter low-pass transfer curve 3dB frequency. If the loop bandwidth is set too high, the quantization noise will start rising in-band and degrade the output spectral purity. Setting the bandwidth too low will increase the noise due to the VCO appearing at the output, and is thus undesirable. The design of the noise shaping loop is beyond the scope of this work and will not be described here.

2.1.2. Synthesis Using Digital PLLs

After the brief overview of analog PLL architectures, digital implementations are considered next. Digital PLLs (DPLLs) have been receiving increasing attention recently due to their relatively lower cost, faster design turnaround, ease of migration to a newer, smaller process technology and less susceptibility to PVT variations. These promising features become more pronounced as gate dimensions are shrunk further and analog process options do not follow at the same rate as digital device dimensions. [1,10]. Furthermore, as loop parameters are controlled by a set of digital words, self-calibration methods are more easily implemented. [17]

A generic DPLL block diagram is given in Fig 2.3. In a digital PLL, the phase detector and charge-pump combination usually found in an analog PLL is replaced by a time-to-digital converter (TDC). This block is followed by a digital loop filter (a PI controller) and a digitally-controlled oscillator in the forward path. The oscillator may be composed of analog cells controlled by a digital word, i.e., a digitally-controlled analog oscillator (DCAO). The divider can be made identical to that in an analog PLL, with similar design considerations such as giving extra attention to the power hungry high-speed first stage.

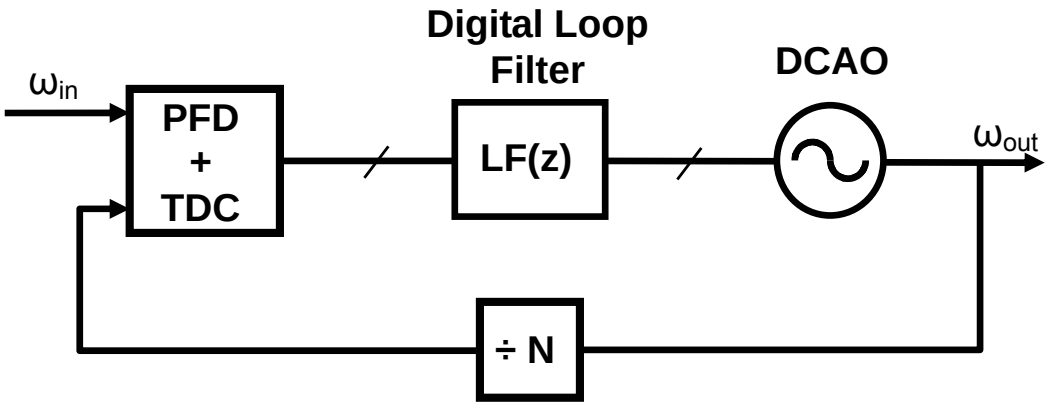


Fig. 2.3 DPLL block diagram.

The resolution of the TDC and the step size of the DCAO have a significant impact on the performance of a digital PLL and need careful design effort. The TDC performs quantization in time of the phase difference between the reference and the DCAO signal. Therefore, any phase error that is smaller than an LSB of the TDC will go uncorrected. Another way to think of this is that the loop during such an instance is open, as no correction is being made. This means the phase error has to accumulate until it reaches an LSB so that it can be detected and corrected. On the other hand, the finite frequency resolution of the DCAO implies that a DPLL can only be in a state of dynamic lock, unlike an analog PLL with a fully analog VCO whose control voltage is continuously variable. Therefore in a locked state, the control word of the DCAO will periodically change to maintain a minimum average phase error, which results in spurious tones at the DPLL output. Techniques where the LSB of the DCAO control word is dithered to spread the spurious noise energy into the noise floor have been presented in the literature [18].

Almost all TDC implementations for a DPLL implementation rely on the use of a chain of delay cells, whereby, the time window to be digitized is expressed in terms of the unit delay used in the chain as in [18]. The delay chain can also be made of logarithmically weighted delay cells, such that a larger time window can be covered as in [17]. The resolution in these implementations is limited by the smallest gate delay possible in the process used. For the 0.35μm process used in this research project, this delay is around 100ps. The process used in [18] has a minimum inverter delay of 40ps, which is more than enough for the described design to meet GSM specifications. The main goal of the system described in this thesis is to increase the time detection resolution beyond a single inverter delay and thus improve the DPLL performance. The idea has common elements with the system presented in [19], the most significant difference being the dependence of the STDC described here on a majority decision scheme decided by the statistical distribution of arbiter outputs and their polarities. Using this system as a fine TDC in a DPLL along with a coarse TDC implemented the traditional way is a possible application.

3. STOCHASTIC TIME-TO-DIGITAL CONVERTER

The stochastic time-to-digital converter (STDC) is intended to make up the fine loop of the phase detector in a DPLL. This block aims to increase the resolution of time detection beyond a single inverter delay, which is the best that can be expected from the inverter chain approach used in the coarse loop. A high resolution in time detection improves the noise performance of the overall system and is thus desirable.

3.1. Theory of Operation

The operation of the STDC relies on the finite voltage offset between the two inputs of a set of arbiters on a given wafer. These arbiters have as their inputs the two rising edges that define the time window to be digitized by the STDC. Within a PLL, the two inputs would be given by the reference input and the DCAO output (possibly divided). The latches are responsible for determining which edge comes first, such that the DCAO edge coming earlier than the reference edge could be represented by a 1 at the latch output, while the opposite situation would output a 0, or vice versa.

The fact that the latches will exhibit finite mismatch and will never be identical suggests that identical inputs to separate latches may generate conflicting answers. Using a decision scheme that processes all the latch outputs and produces an output word corresponding to the majority of the latch decisions, the time window of interest can be digitized with higher precision than possible with a chain of inverters. The output behavior depends on the statistical distribution of the voltage offset among the latches. A larger standard deviation implies a larger dynamic range at the expense of lower resolution. This will be clarified by explaining the operation using an example.

Suppose the two inputs (represented by edges A and B) have a considerably large phase error between them, such that all of the latches unambiguously state edge A (reference) comes earlier than edge B (DCAO). Furthermore, suppose that this condition is represented by a 1 at the latch outputs. This situation would represent the saturation condition where the inputs represent a timing error beyond the TDC

dynamic range. As the phase error is reduced, because of the finite rise-time of the input signals as well as the finite voltage offset on the latch inputs, some latches output a 1 while others output a 0. This is because the voltage offset shifts the thresholds of the latches randomly. In this case, the number of 1's will be greater than the number of 0's, and this information is reflected on the output word as an increase in the DCAO speed. The amount by which the number of 1's exceeds the number of 0's is a measure of the phase error. As the phase error is lowered, this difference will decrease. In the extreme case, where the phase error is exactly zero, roughly half of the latches will state that edge A came earlier, while the remaining latches state otherwise. The output word will be approximately zero on average, but will fluctuate about this value from edge to edge, which will reflect on the loop filter input. If an infinite number of latches could be used, for zero phase error, exactly half of the latches would output a 1 and the rest a 0. Fig 3.1 provides a conceptual picture of the STDC.

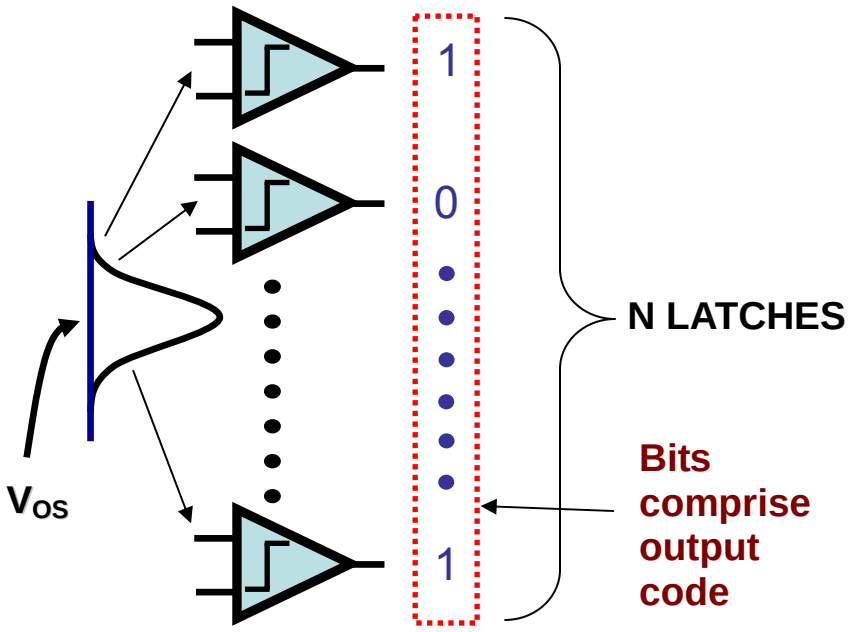


Fig. 3.1 Fine TDC conceptual diagram.

3.2. Dynamic Range vs. Resolution Trade-Off

The saturation point and the resolution of the TDC are determined by the standard deviation of the voltage offset on the latches. A larger standard deviation suggests a wider dynamic range such that the saturation point (where all latches have the same output level) is pushed out. However, this also means that a fixed number of latches operate over a wider distinguishable range, which implies a lower resolution. On the other hand, a small standard deviation on the voltage offsets will cause the latches to saturate with a lower phase error. However, at low phase errors the latches will be operating over a smaller range such that resolution is higher.

The statistical distribution of the offsets significantly affects the distribution of the output codes. This offset is modeled as a normal distribution with properties that depend on device sizing. Specifically, the expected offset distribution for a given design can be calculated using Pelgrom's coefficients [20], which can be obtained from process data.

Another point to note is that input signal rise times affect the STDC output code vs. input phase error transfer curve similarly. A large rise-time implies a larger dynamic range with the STDC saturation points pushed out. Conversely a fast rising signal is less susceptible to arbiter offsets and thus the dynamic range is reduced. Thinking at the extreme case of an infinitely fast signal makes this effect easier to understand. Considering a square wave input to the arbiters, we can deduce that regardless of the offset, the arbiter output trips when the square wave rising edge arrives. Hence for signals with a rise time of zero, offsets would change nothing. In this respect, a large (small) rise time has the same effect as a large (small) distribution in the arbiter offsets.

3.3. System Level Simulations of the STDC

A number of cases have been analyzed in MATLAB to demonstrate the operation of the STDC. The input frequency is fixed at 40MHz, as that is the desired channel separation for the DPLL application where the STDC will be used. Two different offset distributions of 20mV and 40mV 1-σ are examined, assuming the offsets can be modeled using the normal distribution. One hundred latches are used initially to more easily interpret the operation. This value affects the outcome accuracy and has implications on the required die area. More latches imply a higher resolution but also a larger die area. The Simulink model is shown in Fig. 3.2. The setup as shown makes it possible to change the rise time of either of the two TDC inputs. The cumulative latch decisions are plotted against time as the phase error is swept from -100ps to 100ps. At each time step, 50 samples are taken, which correspond to 50 different voltage offset distributions. In an actual implementation this would arise from 50 different chips. The dark lines through the centers of the distributions shown in Figs. 3.3 and 3.4 are the averages of all 50 iterations taken at each time step. In other words, it is the average of the codes that 50 sample chips would produce given the sweep of phase error as in Figs. 3.3 and 3.4. As expected, the average trend shown =by the dark line is reasonably linear and crosses very close to the origin. If an infinitely large sample space were used, such that the codes produced by infinitely many sample chips could be averaged, the dark line would cross the origin and be perfectly linear.

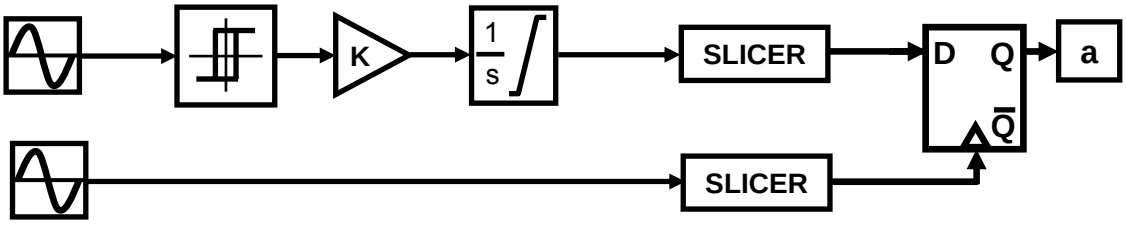


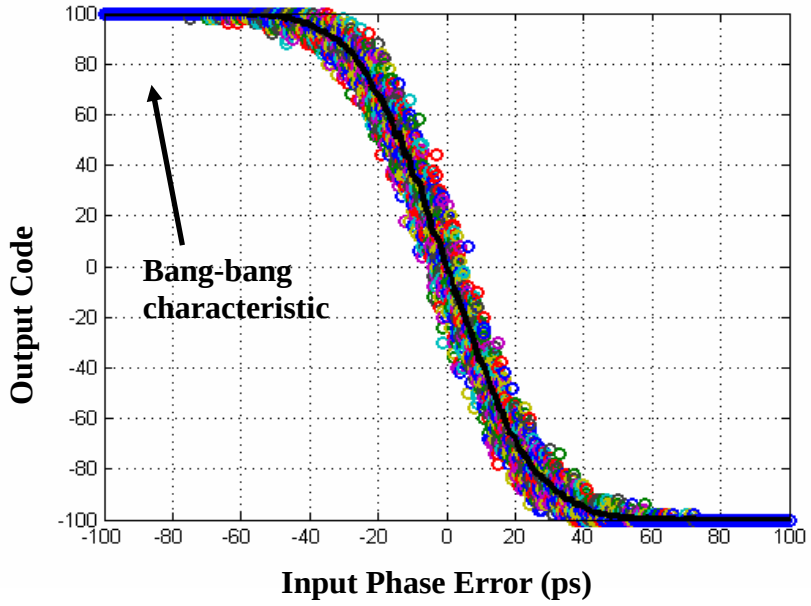
Fig. 3.2 Simulink model of the stochastic TDC.

3.3.1. Effect of Changes in Offset Distribution

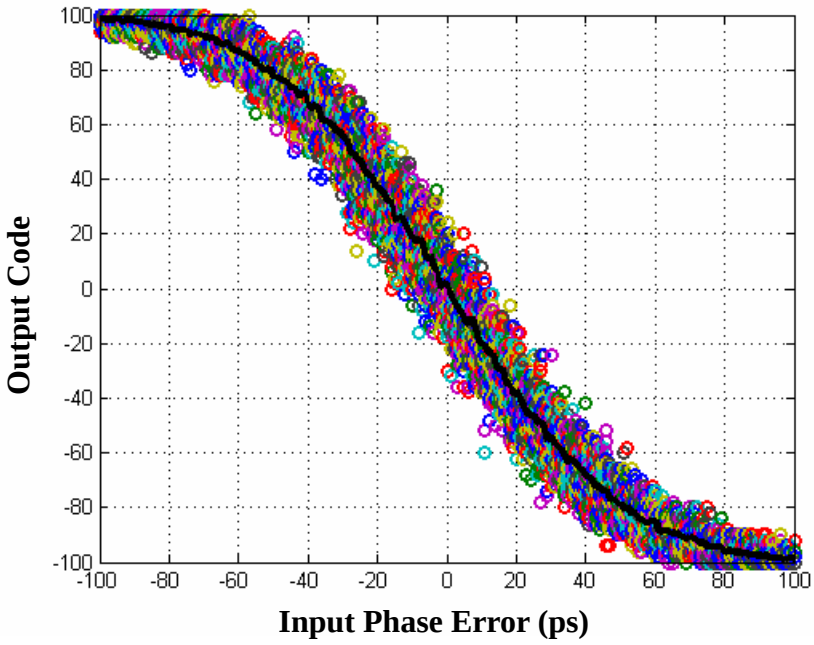
The phase error between the TDC inputs is swept while the input rise-time is fixed at 2ns. The results are shown in Figs 3.3 (a) and (b) for 20mV and 40mV 1-σ offsets respectively. The former shows signs of saturation at around ±20ps phase error, while the latter starts saturating around ±40ps. Beyond this point the gain of the transfer curve decreases, and the TDC operation takes on a bang-bang characteristic, with all the latches giving the same output. As seen in the figure, a larger spread in the voltage offsets increases the dynamic range by pushing the saturation points out, at the expense of reduced resolution within the linear region. Looking at the dark line showing the average, the gain around the origin for the 20mV case is 4 code levels per picosecond time difference, while that for the 40mV case is 2 code levels per picosecond. This gain scales between the test cases shown in Figs. 3.3 and 3.4 as long as the linear region is considered.

3.3.2. Effect of Different Input Signal Rise-Times

For a fixed 20mV 1-σ offset the effects of different rise-times are observed. Two rise-times of 4ns and 8ns are used, with the results shown in Figs 3.4 (a) and (b), respectively. It is worth noting that the low rise time case is similar to the effect observed when a narrow offset distribution is examined. As the rise time is increased, the dynamic range expands and resolution decreases as explained previously.

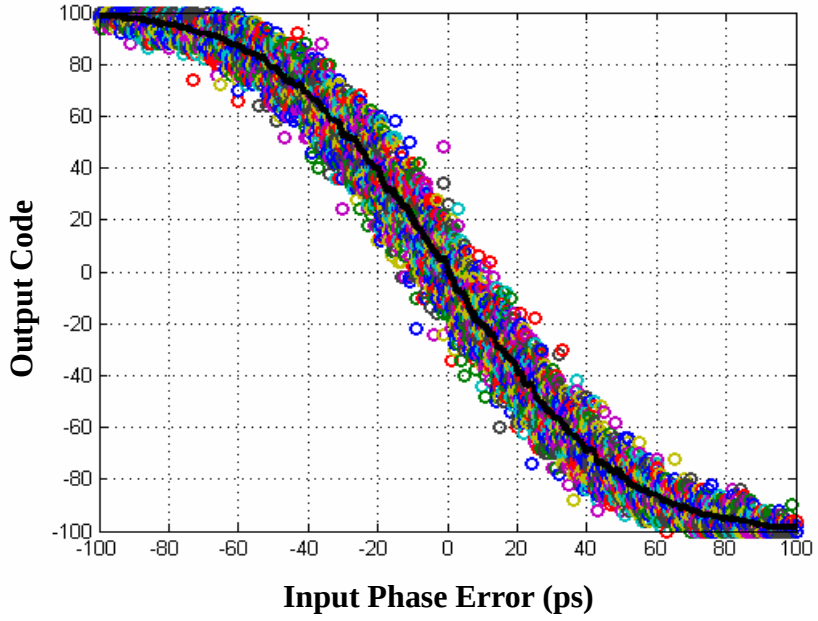


(a)

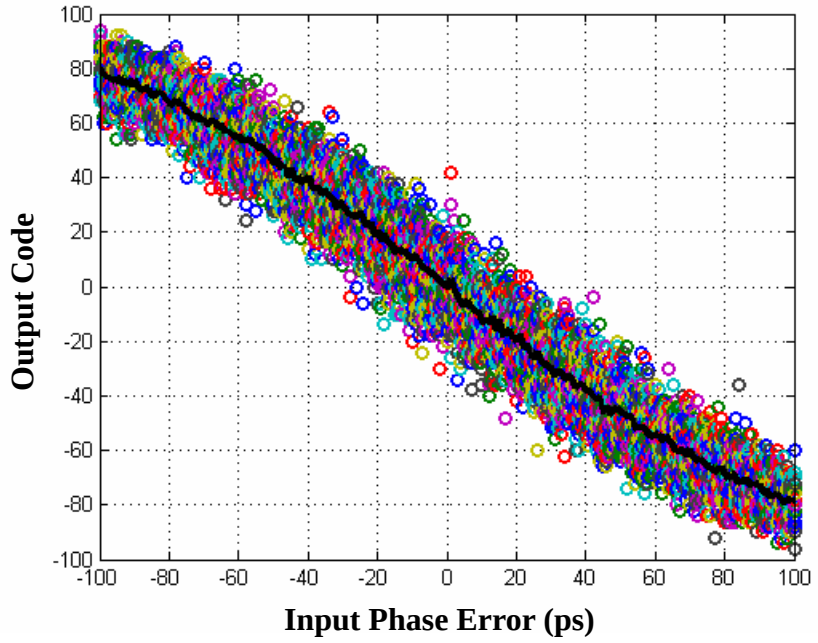


(b)

Fig. 3.3 Transfer characteristics of the stochastic-TDC for different offset voltages. (a) 20mV 1-σ offset, (b) 40mV 1-σ offset.



(a)



(b)

Fig. 3.4 Transfer characteristics of the stochastic-TDC for different rise times. (a) 4 ns rise time, (b) 8 ns rise time.

3.4. Implementation

The first implementation on a test chip is intended to enable testing with external stimuli as well as observing the output codes when the STDC is operated along with a DPLL. Both situations make use of the same circuitry, which is described next.

3.4.1. Delay and Slew Control

For characterizing the STDC, it is desirable to have a means of controlling the phase delay to be digitized. Moreover, the rise time of the input signals needs to be controllable to enable comprehensive testing, because as described earlier, signal rise times have the same effect as offset voltage distributions within the STDC. This functionality is incorporated using a set of current starved inverters and a common-source (CS) amplifier stage with controllable bias current. The inverters give a programmable delay while the CS stage enables variable slew rate. Both inputs to the arbiters pass through an independent delay and slew control stage, with a total of four controls overall. With this arrangement, one edge can be advanced or delayed with respect to the other, and the slew rate can be modified as desired to test the dynamic range of the converter. To test the STDC with external stimuli, the controls of the delay and slew control circuit would be shorted together, such that no extra phase delay is introduced to any of the input edges. An example of such a test would be using the reference and DCAO rising edges coming from a DPLL designed with a conventional, delay-chain based TDC. This would enable comparing jitter numbers or noise spectra under locked condition, and help validate the performance improvement due to the STDC.

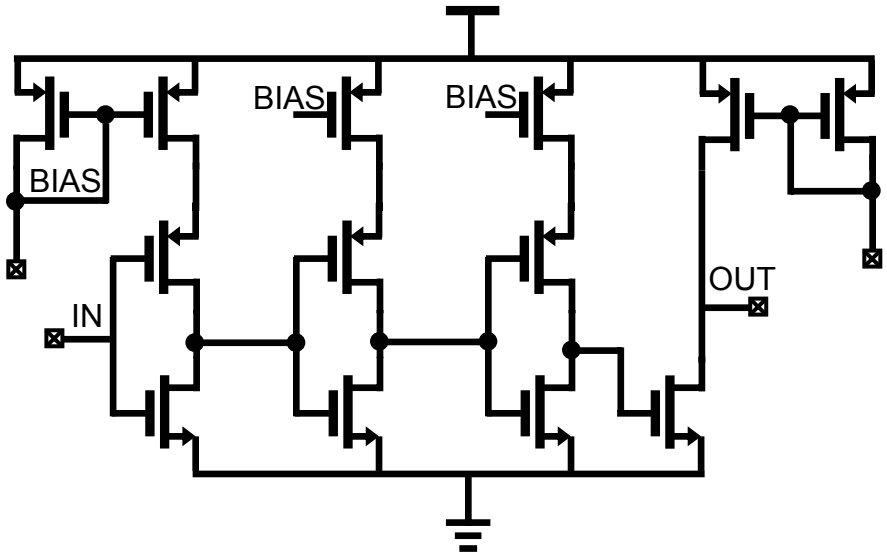


Fig. 3.5. Delay and rise time control circuit.

The circuit diagram is shown in Fig. 3.5. The control currents are generated using off-chip variable resistors. Decoupling capacitors (not shown) are placed between the bias nodes and VDD to minimize noise on supplies affecting the bias point and corrupting measurements. When testing the STDC with the ADPLL reference and the DCAO output, the delay controls can be shorted together at the output so as to not introduce any extra delay into the measurements. The delay components can generate a total delay between the signals of up to 1.2ns, which is more than enough to cover a coarse TDC bin of about ±100ps. The slew rate is adjustable between 1ns and 7ns.

3.4.2. Latch and Sampling Flip-Flop:

The latch is comprised of a cascade of the arbiter and an SR-latch. The SR-latch is included to help reduce metastability problems. A standard TSPC flip-flop implementation stores the output bits that come from the arbiter/SR-latch combination, before they are fed to the 84:7 encoder. The arbiter schematic is shown in Fig. 3.6. The positive feedback created by the cross-coupled gates enables latching.

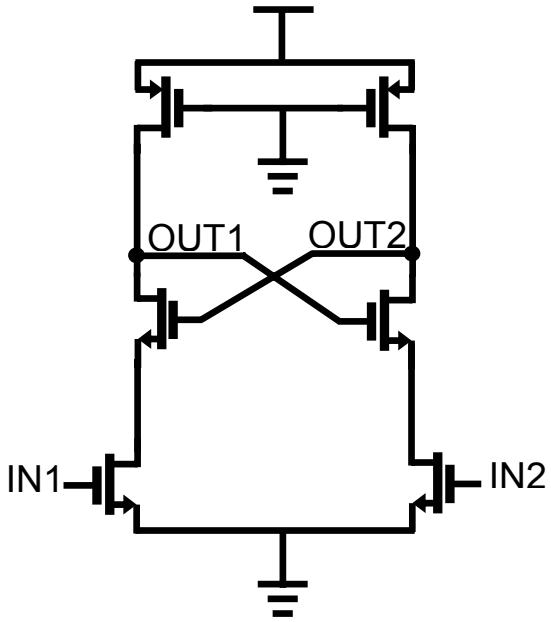


Fig. 3.6. Arbiter schematic.

The use of 84 latches makes the implementation of an encoder easier as seen in the next subsection. An encoder is necessary to observe the output codes from the latches off-chip. Since a large number of latches is required for statistical significance, some means of encoding the data make it possible to observe code distributions with realistic pin counts.

3.4.3. 84:7 Encoder

This circuit converts the 84 output bits coming from the flip-flops into 7 binary-weighted output bits. The low speed of operation makes a straightforward implementation possible. The 84 bits are handled in three groups of 28. The 28 bits are first divided into groups of 7 and converted into 3 bits using full-adders. The 7:3 encoder block diagram is shown in Fig. 3.7. After the 7:3 encoders, four 3-bit operands result, which are combined into two 4-bit operands using two 3-bit ripple carry adders. Then these 4-bit operands are combined using a single 4-bit adder to produce a 5-bit result. At this point, 28 output bits are represented as a 5-bit binary signal. Three such blocks are used to generate three 5-bit operands, which are combined using a three input 5-bit adder to finally produce a 7-bit word that represents the 84 output bits. The block diagrams for the 28:5 encoder and the 84:7 encoder are shown in Figs. 3.8 and 3.9 to clarify the signal flow.

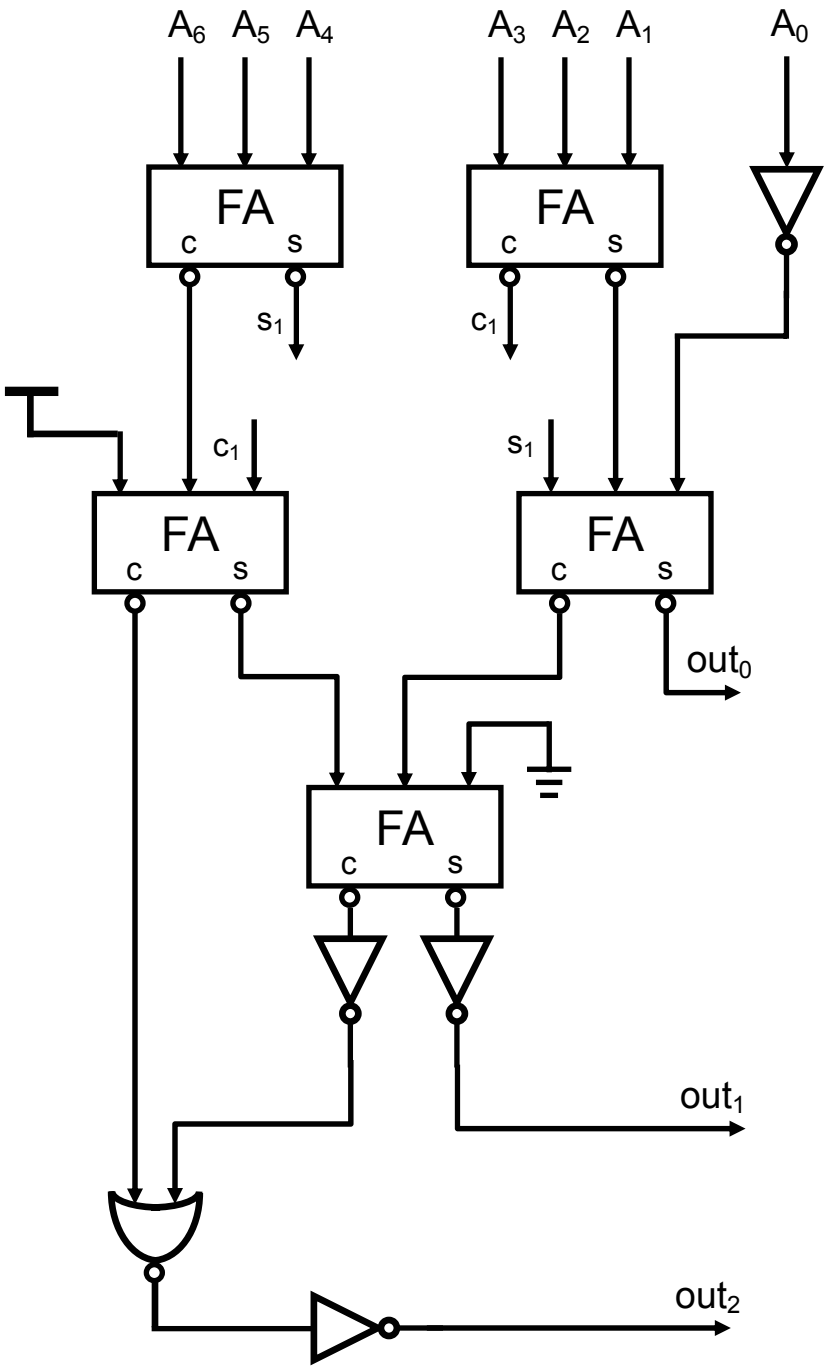


Fig. 3.7. 7-to-3 encoder.

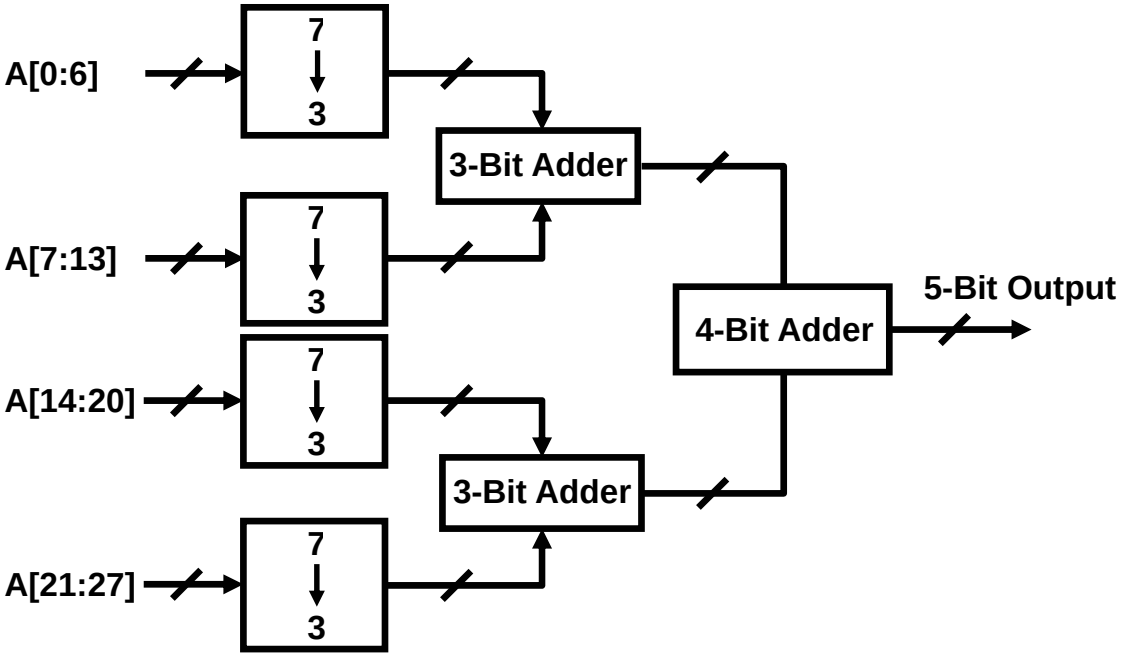


Fig. 3.8. 28-to-5 encoder.

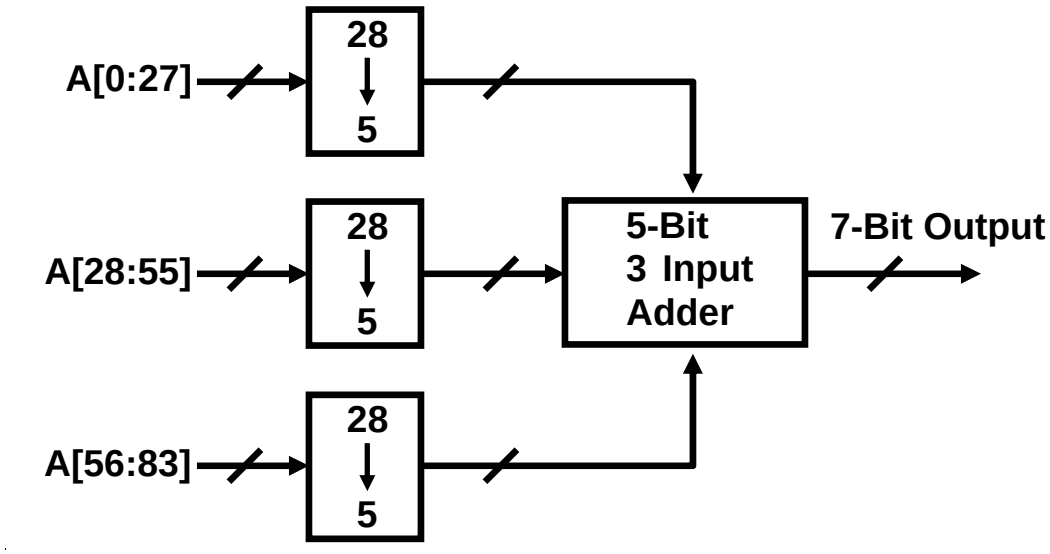


Fig. 3.9. 84-to-7 encoder.

3.4.4. Master-Slave Flip-Flop:

This configuration is used to generate the input signals to the slew and rise time control circuitry, as well as to generate the quadrature sampling edge that is used to capture the output bits in the sampling flip-flops. The low speed of operation enables using these cross-coupled NAND based latches to build a divide-by-2 circuit. The last stage of the divider in the PLL is implemented in the same way. The configuration is shown in Fig. 3.10. The block diagram showing the system and the signal flow is shown in Fig 3.11.

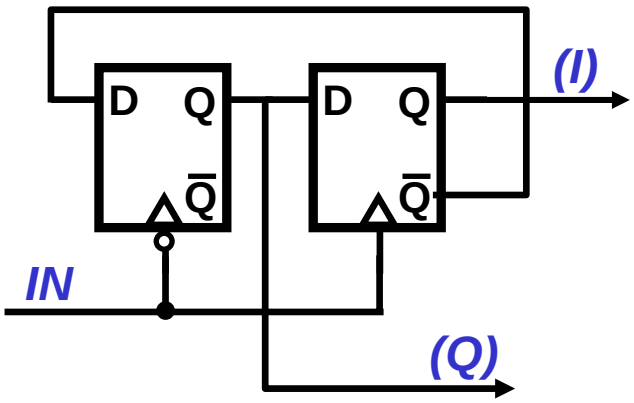


Fig. 3.10. Master-slave divide-by-2 flip-flop.

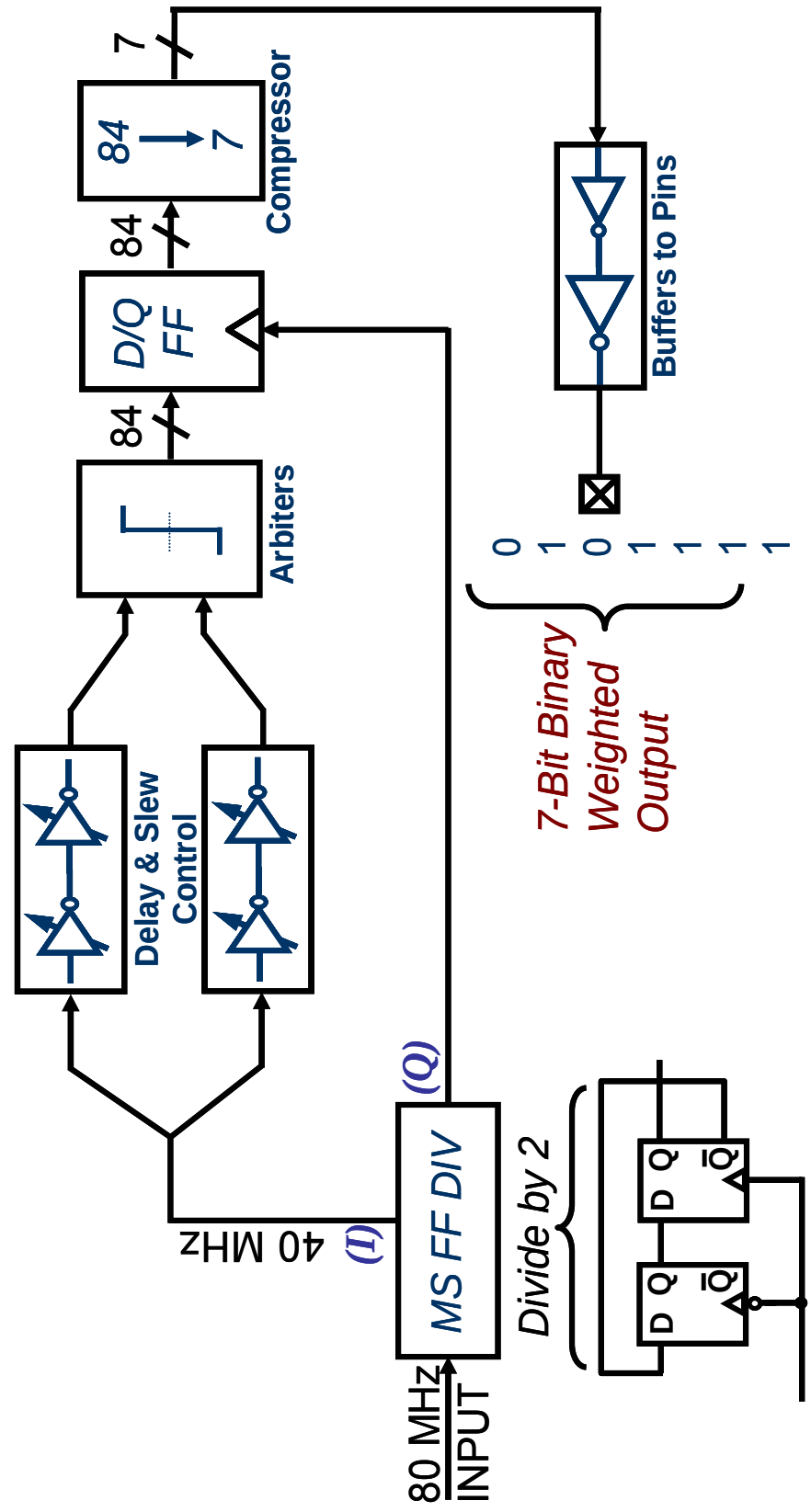


Fig. 3.11 Overall block diagram of test setup.

3.5. Layout

The STDC layout is shown below in Fig. 3.12. Most of the implementation is digitally intensive and thus not very susceptible to corruption by noise. The most important matching consideration in this layout is the routing delays of the two arbiter inputs which needs to be as similar as possible. If one of the inputs is delayed differently than the other, an offset error will be introduced into the measurement. To minimize this, matching layout techniques were employed and the arbiter inputs were routed very closely, which also reduces the effects of noise coupling [21,22].

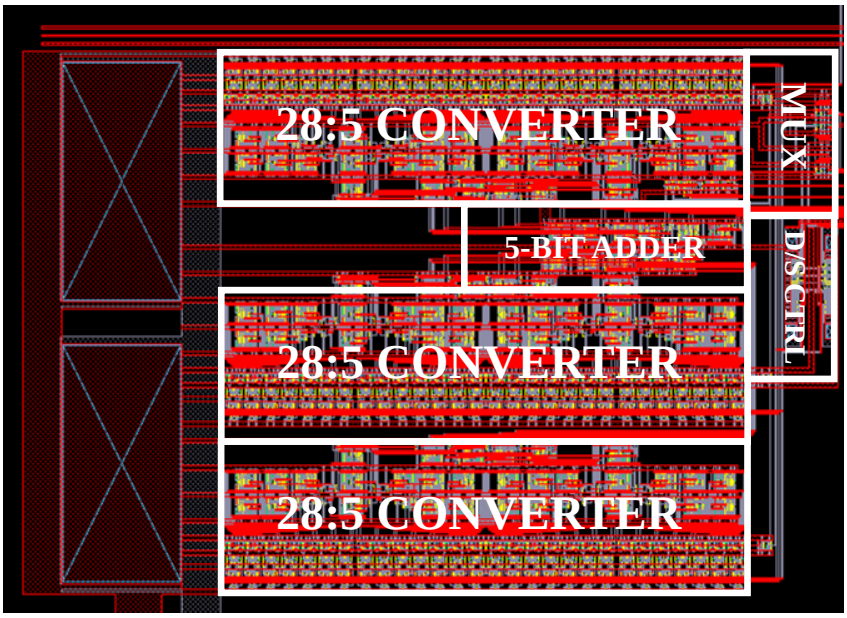


Fig. 3.12. Layout of the STDC core.

Separate buffer and chip supply rails were used, with extensive supply decoupling placed on both. The SOI process exhibits low capacitance between traces and the bulk, which caused noisy supply problems on the previous test chip where this process was used [23]. The core die area is approximately 1150µm x 1700µm. A 2-to-1 multiplexer switches between an external stimulus or the PLL reference and the DCAO divided output to be able to test the circuit under both configurations.

3.6. Circuit Level Simulations:

MATLAB simulation results with different input voltage offset distributions and different input signal rise times were presented in previous sections. Fig. 3.13 shows the results of full transistor level simulations of the designed STDC. Three cases are presented, one with the lowest expected offset amount of 20mV 1-σ and the others with the highest expected value of 40mV 1-σ tested for two different rise times. The offsets are introduced to the latches through a brute force method of including 84 voltage sources before the latches, whose values generate the desired offset characteristics. As seen in the plots the dynamic range for the 20mV case is much less than that in the 40mV case. The effect of the changing rise time on the dynamic range can also be observed.

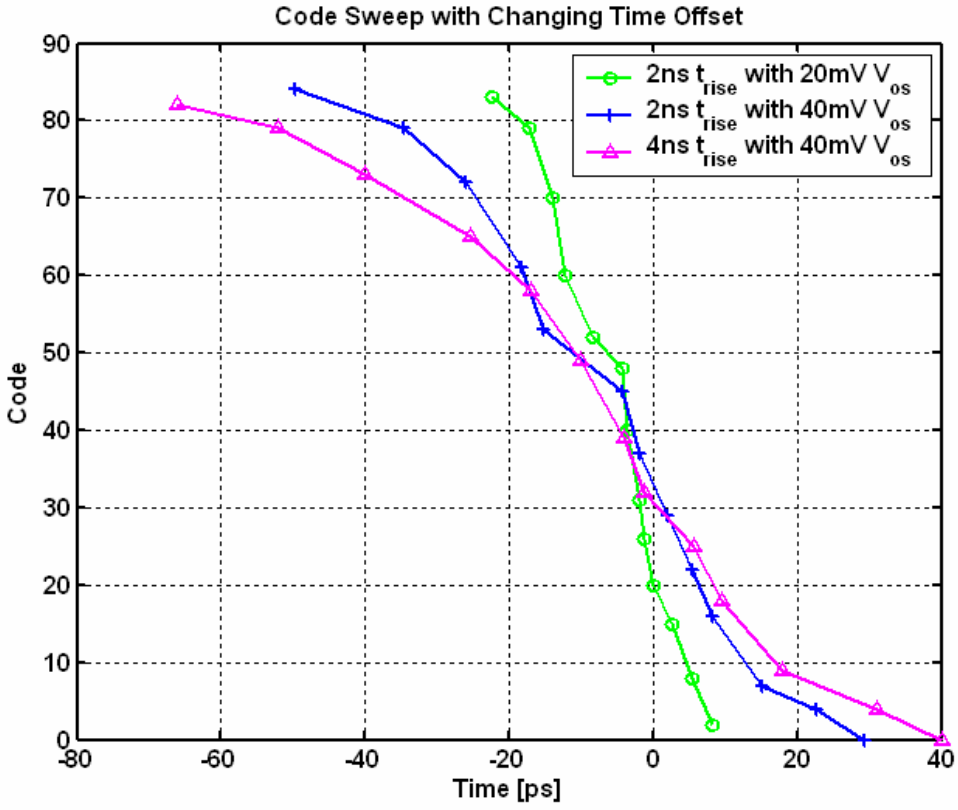


Fig. 3.13. Transistor level simulation results for different rise times in Spectre.

Performing this simulation with different distributions with a large sample space would ultimately create a similar plot as in the MATLAB plots of Figs. 3.3 & 3.4. As can be seen from Fig 3.13, with a means of distinguishing between codes, this STDC can be incorporated into a DPLL for high resolution time digitization. Although the transfer curve is not perfectly linear, the granularity is much less than that with a chain of delay cells. Therefore, it is reasonable to expect that, with proper implementation, the performance will be superior to DPLLs that incorporate the traditional TDC architecture.

For validation of the test chip, a number of sample chips will be used to generate the transfer curve of the STDC by sweeping the input phase error. The main objective of these tests will be to generate a response similar to those shown in Fig 3.13. The secondary goal will be to test the STDC with the DPLL that is on the same die. The reference input of the DPLL and its divided DCAO output will be multiplexed into the STDC. The output codes will be examined and compared with the bang-bang phase detector output of the DPLL to see whether finer resolution in the TDC would have provided desirable information.

4. CONCLUSION

This project examines the possibility of increasing the resolution for time-to-digital conversion (TDC) with a stochastic approach. The intended application is for digital phase-locked loops, where an increase in the time-to-digital converter resolution is desirable for improved spurious performance.

The STDC has been designed, simulated and fabricated in a 0.35μm SOI CMOS process. The simulation results indicate that this circuit is capable of time digitization with a resolution better than an inverter delay. The circuit level simulations follow a similar trend with system level simulations done in MATLAB. Measurements on a large number of chips will most likely provide a more comprehensive picture of the stochastic trend exhibited by the TDC.

Future work regarding this project could include implementing an ADPLL with the STDC incorporated into the block. The STDC could be further refined to be more area efficient. The 84:7 encoder could be shrunk in size by implementing a more efficient majority decision circuit. The brute force method of adding latch outputs used in this work is straightforward but takes up large die area and takes a long time to settle to the correct output word. Although this latter issue is not a problem with a 40MHz input reference, for higher input reference frequencies it might take up an unreasonably large fraction of the available reference period.

BIBLIOGRAPHY

[1] R. B. Staszewski, J. Wallberg, S. Rezeq, C. Hung, O. Eliezer, S. Vemulapalli, C. Fernando, K. Maggio, R. Staszewski, N. Barton, M. Lee, P. Cruise, M. Entezari, K. Muhammad, and D. Leipold, "All-digital PLL and GSM/EDGE transmitter in 90nm CMOS," *ISSCC Dig. Tech. Papers,* Feb. 2005, pp.316–317.

[2] I. Young, J. K. Greason, J. E. Smith, and K. L.Wong, "A PLL clock generator with 5 to 110 MHz lock range for microprocessors," in *ISSCC Dig. Tech. Papers,* Feb. 1992, pp. 50-51.

[3] I. Hwang, S. Lee, S. Lee, and S. Kim, "A digitally controlled phase-locked loop with fast locking scheme for clock synthesis application," in *ISSCC Dig. Tech. Papers,* Feb. 2000, pp.168–169.

[4] C. Chung and C. Lee, "An all-digital phase-locked loop for high-speed clock generation," *IEEE J. Solid-State Circuits,* vol. 38, pp. 347-351, February 2003.

[5] T. Hsu, B. Shieh, and C. Lee, "An all-digital phase-locked loop (ADPLL) based clock recovery circuit," *IEEE J. Solid-State Circuits,* vol. 34, pp. 1063-1073, August 1999.

[6] A. Hajimiri and T. Lee, "A general theory of phase noise in electrical oscillators," *IEEE J. Solid-State Circuits*, vol. 33, pp. 179–194, February 1998.

[7] B. Razavi , "A study of phase noise in CMOS oscillators," *IEEE J. Solid-State Circuits*, vol. 31, pp. 331–343, March 1996.

[8] J. Tierney, C. M. Radar, and B. Gold, "A digital frequency synthesizer," *IEEE Trans, on Audio Electroaccoust.,* vol. AU-19, pp. 48-57, March 1971.

[9] L. K. Tan and H. Samueli, "A 200 MHz quadrature digital synthezier/mixer in 0.8μm CMOS," *IEEE J. Solid-State Circuits,* vol. 30, pp. 193-200, March 1995.

[10] R. B. Staszewski, "Digital Deep-Submicron CMOS Frequency Synthesis for RF Wireless Applications," *Ph.D. thesis*, The University of Texas at Dallas, August 2002.

[11] F. Gardner, "Charge pump phase-lock loops," *IEEE Trans. Commun.*, vol. COM-28, pp. 1849–1858, Nov 1980.

[12] A. Marques, M. Steyaert and W. Sansen, "Theory of PLL fractional-N frequency synthesizers," *Wireless Networks* 4, pp. 79-85, 1998.

[13] T. A. D. Riley, M. A. Copeland, and T. A. Kwasniewski, "Delta–sigma modulation in fractional-N frequency synthesis," *IEEE J. Solid-State Circuits*, vol. 28, pp. 553–559, May 1993.

[14] S. Pamarti, L. Jansson and I. Galton, "A wideband 2.4-GHz delta-sigma fractional-N PLL with 1-Mb/s in-loop modulation," *IEEE J. Solid-State Circuits*, vol. 39, pp. 49–62, January 2004.

[15] M. H. Perrott, T. L. Tewksbury III, and C. G. Sodini, "A 27-mW CMOS fractional-N synthesizer using digital compensation for 2.5Mb/s GFSK modulation," *IEEE J. Solid-State Circuits*, vol. 32, pp. 2048–2059, December 1997.

[16] J. Candy and G. Temes, *Oversampling Delta-Sigma Data Converters*. NewYork: IEEE Press, 1992.

[17] J. Lin, B. Haroun, T. Foo, J.-S. Wang, B. Helmick, S. Randall, T. Mayhugh, C. Barr, and J. Kirkpatrick, "A PVT tolerant 0.18MHz to 600MHz self-calibrated digital PLL in 90nm CMOS process," *ISSCC Dig. Tech. Papers*, Feb. 2004, pp. 488–489.

[18] R. B. Staszewski, C.-M. Hung, K. Maggio, J. Wallberg, D. Leipold, and P. T. Balsara, "All-digital phase-domain TX frequency synthesizer for Bluetooth radios in 0.13µm CMOS," *ISSCC Dig. Tech. Papers*, Feb. 2004, pp. 272–273.

[19] V. Gutnik and Chandrakasan A, "On-chip picosecond time measurement," *2000 Symposium on VLSI Circuits - Digest of Technical Papers,* 2000, pp. 52-53.

[20] M. Pelgrom, J. Duinmaijer, and G. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1433-1440, October 1989.

[21] D. Johns and K. Martin, *Analog Integrated Circuit Design*. New York, NY: John Wiley & Sons, Inc., 1997.

[22] B. Razavi, *Design of Analog CMOS Integrated Circuits.* New York: McGraw Hill, 2001.

[23] A. Nemmani, M. Vandepas, K. Ok, K. Mayaram, and U. Moon, "Radiation hard PLL design tolerant to noise and process variations," in *CDADIC report*, February 2004.