Modeling of a Clock and Data Recovery Circuit with MATLAB/Simulink

Paul Lazar

1.0 Introduction and Background

Clock and data recovery (CDR) is one of the most challenging functions in modern serial data transmission systems. This function takes a serial stream of data, whose frequency spectrum may not include that of the data clock, and retrieves both the clock and the data. The retrieved clock may be used locally to clock local logic.

Data is typically transmitted in NRZ (non retutn to zero) format which is illustrated in fig. 1 below. The ideal recovered clock has its rising edges at the center of the symbol period.

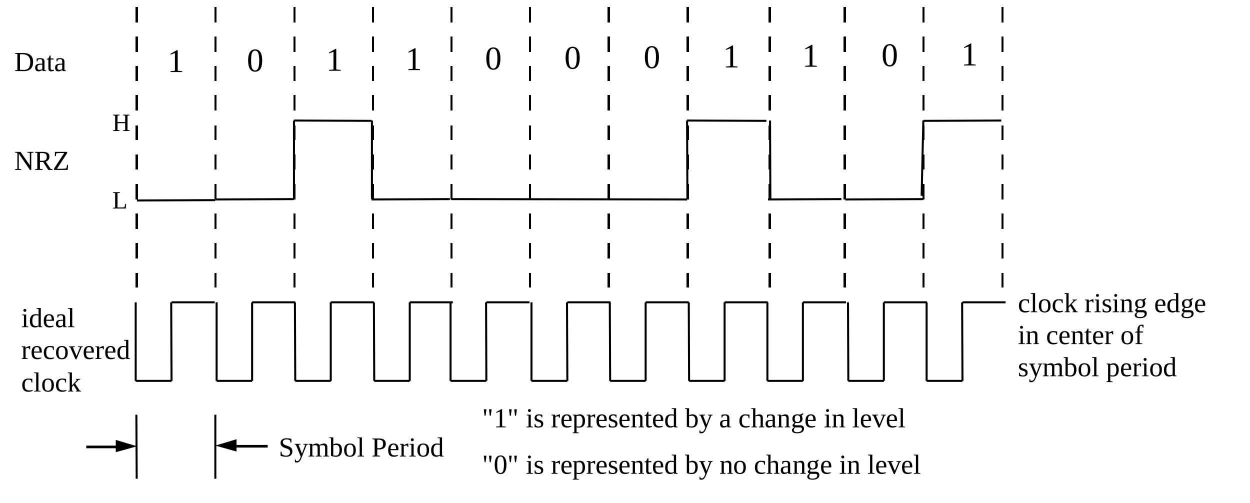
Data
1 0 1 1 0 0 0 1 1 0 1
H
NRZ
L
ideal recovered clock
clock rising edge in center of symbol period
Symbol Period
"1" is represented by a change in level
"0" is represented by no change in level

Fig. 1 NRZ data format

The NRZ format is used because its frequency spectrum is lower than that of other formats (such as RZ, return to zero) and therefore requires less bandwidth for transmission. Fig. 2 shows both NRZ and RZ formats and fig. 3 shows their normalized power spectral densities (PSD). The PSD may be calculated either by squaring the magnitude of the fourier transform of the data pattern or by computing the fourier transform of the pattern's autocorrelation function. The plots were produced with Matlab.

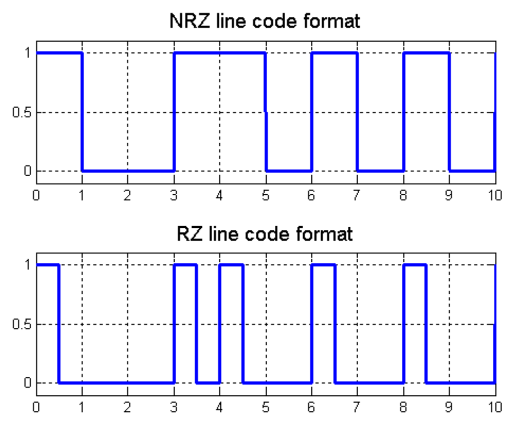

Fig. 2 NRZ and RZ formats

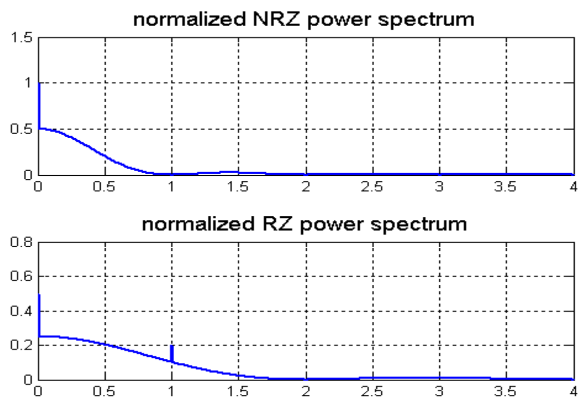

Fig. 3 NRZ and RZ power spectral density

For random data in unipolar NRZ or unipolar RZ format the PSD is given by [1]

$$\text{PSD\_nrz} = \frac{A^2T_b}{4} SINC^2(fT_b)\left[1+\frac{\delta(f)}{T_b}\right] \qquad \text{eqn 1.1}$$

and

$$\text{PSD\_rz} = \frac{A^2T_b}{16} SINC^2\left(f\frac{T_b}{2}\right)\left[1+\frac{1}{T_b}\sum_{n=-\infty}^{\infty}\delta\left(f-\frac{n}{T_b}\right)\right] \qquad \text{eqn 1.2}$$

respectively. The amplitude of the data is A and T_b is the symbol period. SINC(x) is the sin(πx)/ (πx) function and δ(t) is the Dirac delta function.

In practice, in order to avoid long sequences of all 1s or all 0s which may occur with the NRZ format, transitions are inserted after a certain length of 0s or 1s. A typical mapping is the 8B10B scheme which ensures that there are not more than five consecutive 1s or 0s. An 8 bit data portion is mapped to 10 bits. This reduces the data throughput by 25% but has advantages for DC line balance and clock recovery. 64B66B schemes which have less impact on data throughput are also employed. These techniques result in the power spectrum (fig. 4) being 0 at DC since such transitions are guaranteed, however there is still no spectral component at the data rate or multiples thereof. Note that the NRZ spectrum in fig. 3 has zeros at 1, 2, 3 etc., i.e. at multiples of the data rate, as does the spectrum in fig. 4. The 8B10B scheme does significantly increase the spectral content between normalized frequency 1 and 2.

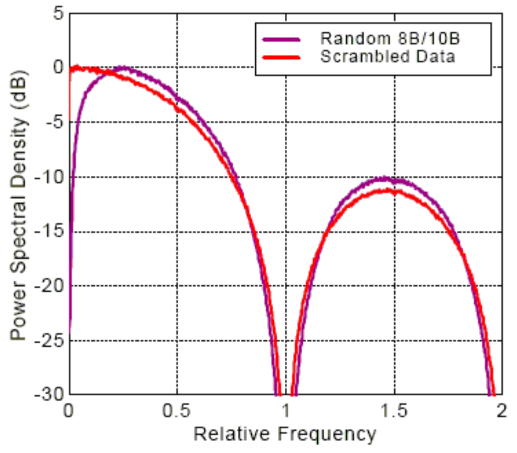

Fig. 4 8B10B code spectrum [2]

Harmonics at the clock rate, which is twice the data rate (fig. 1), can be generated by multiplying the NRZ or 8B10B data with itself, or with a delayed version of itself. It is well known that multiplying a sinusoid by another sinusoid results in harmonics of the sum and difference frequencies since sinASinB = ½(cos(A-B) + cos(A+B)). The spectral content at ½ and 1½ the data rate ceates the desired harmonics. The phase detectors used in clock and data recovery (CDR) circuits are often composed of logic gates and flip flops and are not construed as multipliers, but this is effectivley the function which they perform. For example, an XOR gate which compares the clock with the data may be considered to operate as a multiplier.

A block diagram of a CDR circuit is shown in fig. 5 below. The phase detector, low pass filter and voltage controlled oscillator comprise a phase locked looped (PLL). The PLL may be modeled as a linear feedback system when the loop is in lock. Such a model linearizes the behavior of the loop near a stable operating point, i.e. it’s lock state, and follows the general technique of modelling a non linear system as a linear one by approximating it as having only small excursions about the operating point. Some types of phase detector (PD), such as a bang-bang or Alexander PD, are very non linear in which case a suitably linearized model must be developed. In this work a Hogge type phase detector is used. It can be approximated by a linear model without too much difficulty. Referring to fig. 5, the phase detector compares the clock signal with the incoming data and produces a signal proportional to the phase difference. This difference signal passes through the low pass filter and is fed to the voltage controlled oscillator (VCO). In response to this input, the VCO adjusts the frequency of its oscillation thus causing the frequency of the clock out signal to adjust. The adjustment continues until the clock frequency is equal to the data input frequency. Thus the feedback loop causes the clock out frequency to track the data in frequency. If the data in frequency is stable the clock out frequency will settle to be equal to that of the data in frequency. If the waveforms of the clock and data are square waves we want the edges aligned as shown in fig. 1, so that the clock rising edge occurs at the middle of the symbol period. Also to be noted is that the clock transitions at twice the data in rate, which is why we are interested in the second harmonic of the data in rate as previously mentioned. A separate flip flop which clocks out the data is shown in fig. 5, however if the phase detector is composed of flip flops and logic the data may be brought out from the phase detector directly.

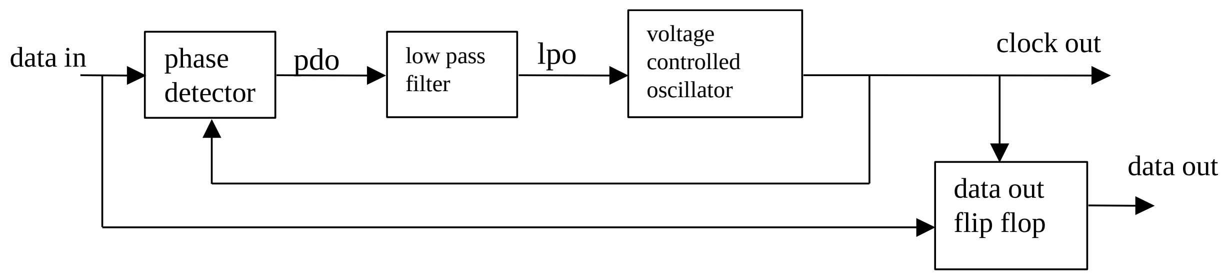

Fig. 5 Clock and Data Recovery Block Diagram

2.0 The Phase Locked Loop (PLL)

The model of the PLL is created by first modelling the individual components and then assembling these into the PLL stucture.

2.1 Voltage Controlled Oscillator (VCO)

With a zero volt input the VCO oscillates at its free running or center frequency of f_vco0. A non zero voltage, v_in, applied to the VCO input causes its frequency to change in proportion to the input voltage. Thus, the VCO output frequency is

f_vco(t) = f_vco0 + (Kvco*v_in(t))   eqn 2.1

where Kvco is the gain constant of the VCO. Thus Kvco*v_in(t) represents the change in the VCO frequency in response to the input v_in(t). Phase is the integral of frequency, by definition, hence the change in the phase of the VCO output is given by ph_vco(t) = $\int K\_vco * v\_in(t)dt = K\_vco * \int v\_in(t)dt$ eqn 2.2

Taking the Laplace transform, and recognizing that integration corresponds to multiplying the transform by 1/s, we have

PH_VCO(s) = Kvco*V_in(s)/s eqn 2.3

or

PH_VCO(s)/ V_in(s) = Kvco/s eqn 2.4

2.2 Low Pass Filter (LPF)

The low pass filter generally consists of an RC network similar or analogous to that shown in fig. 6. The transfer function of the filter, LPF(s), is given by

LPF(s) = $\dfrac{Vout(s)}{Vin(s)} = \dfrac{1+s\tau_z}{1+s\tau_p}$ eqn 2.5

where $\tau_p = (R1+R2)\cdot C2$ and $\tau_z = R2\cdot C2$.

This is known as a lead-lag filter and the component values are such that $\tau_z << \tau_p$. This makes the zero frequency, which is determined by $1/\tau_z$, much larger than the pole frequency which is determined by $1/\tau_p$.

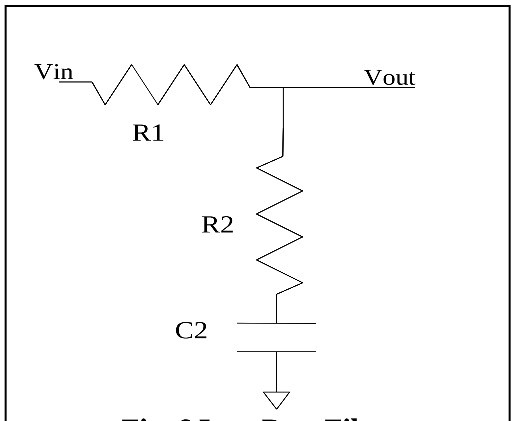

Fig. 6 Low Pass Filter

2.3 Phase Detector (PD)

The function of the phase detector is to generate a signal which is proportional to the phase difference of its input signals. With the loop in lock and with relatively small phase changes of the data input, the phase detector is modeled by [3][4]

V_pdo(t) =Kpd* (ph_data(t) - ph_vco(t)) eqn 2.6

Where V_pod(t) is the output of the phase detector, ph_data(t) is the phase of the input data and ph_vco(t) is the phase of the VCO output, the clock signal. Taking Laplace transforms of both sides we obtain:

V_pdo(s) = Kpd*(PH_data(s) – PH_vco(s)) eqn 2.7

2.4 PLL in lock; linear model

The linearized model of the PLL can now be constructed with the components as described. A signal flow diagram showing the transfer function of each block is shown in fig. 7.

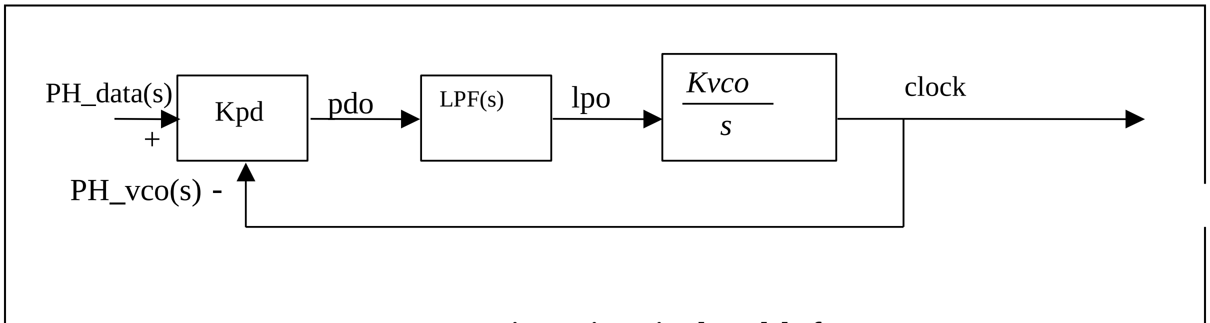

Fig. 7 Linearized model of PLL

Looking a the VCO output, PH_vco(s) and the data input, PH_data(s), we have

$$PH\_vco(s) = \frac{Kvco}{s} \cdot LPF(s) \cdot Kpd\left(PH\_data(s) - PH\_vco(s)\right) \quad \text{eqn 2.8}$$

the closed loop transfer function, which may be rearranged as:

$$H(s) = \frac{PH\_vco(s)}{PH\_data(s)} = \frac{Kvco \cdot LPF(s) \cdot Kpd}{s + Kvco \cdot LPF(s) \cdot Kpd} \quad \text{eqn 2.9}$$

When the transfer function for LPF(s) is included (eqn 2.5) we obtain

$$H(s) = \frac{s \cdot Kvco \cdot Kpd \cdot \left(\tau_z / \tau_p \right) + \left(Kvco \cdot Kpd / \tau_p \right)}{s^2 + s\left(1 + KvcoKpd \cdot \tau_z\right) \cdot \left(1/\tau_p \right) + \left(Kvco \cdot Kpd / \tau_p \right)} \quad \text{eqn 2.10}$$

where again, $\tau_p = (R1 + R2) \cdot C2$ and $\tau_z = R2 \cdot C2$.

The natural frequency of H(s) , w0, is given by $\varpi_0 = \sqrt{\frac{KvcoKpd}{\tau_p}}$ eqn 2.11

The damping factor, (called zeta), is $\zeta = \frac{\omega_0}{2}\left(\tau_z + \frac{1}{KvcoKpd} \right)$. eqn 2.12

The denominator of H(s) can then be written as[3] D = $s^2 + 2\zeta\omega_0 s + \omega_0^2$. eqn 2.13

The open loop transfer function is

$$TFol(s) = \left(s \cdot \tau_z \cdot Kvco \cdot Kpd + Kvco \cdot Kpd\right) / \left(s^2 \tau_p + s\right) \quad \text{eqn 2.14}$$

Bode plots showing the magnitude and phase of the transfer function Tfol(s) and H(s) can readily be plotted with Matlab. A simple script to do this, with given values of Kvco, Kpd, R1, C1 and R2 is provided in the appendix. Representative plots are shown in fig. 8 and fig. 9 below. For the plot, values of Kvco = 5.0e7 Rad/V, Kpd = 1.0e-3 V/Rad, C2=1.0e-9, R2=1K and R1=10k were used. In order to maintain the relationship $\tau_z << \tau_p$, R1 is ten times R2. With these component values, w_0 is 70.4Khz and zeta is 0.707. With a phase margin of 65.6 degrees, the plot of fig 8 indicates a stable system.

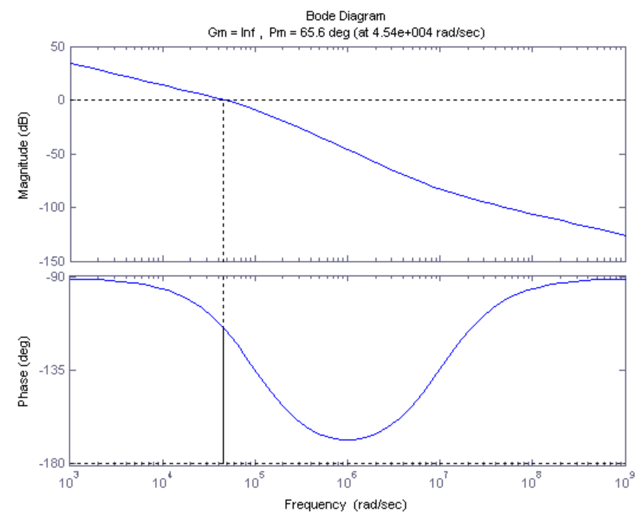

Fig. 8 Bode Plot of open loop transfer function

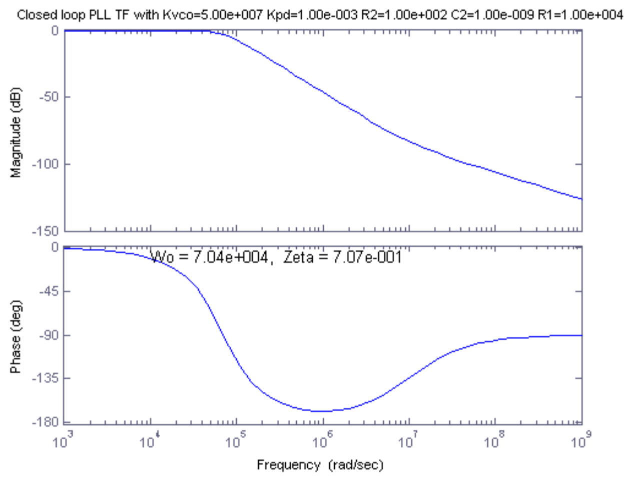

Fig. 9 Bode Plot of closed loop transfer function

In fig.10 is shown the step response of the closed loop transfer function. With a damping factor zeta equal to 0.707, some overshoot is present.

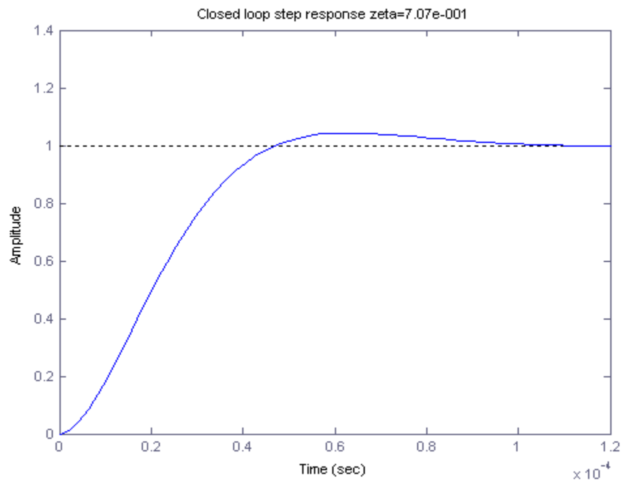

Fig. 10 Closed loop step response

2.5 PLL with Digital Phase Detector and Charge Pump

Most high speed CDR circuits use some type of digital phase detector. Such phase detectors may be classified as linear or bang-bang, with the linear type producing a digital signal that is proportional to the phase difference and the bang-bang type simply producing a bit indicating whether the phase is leading or lagging. In this work a Hogge phase detector, which is a linear type, is used. The Hogge PD and the PLL implemented with the Hogge PD will now be briefly described.

2.5.1 Hogge Phase Detector and Charge Pump

A schematic of the Hogge PD is shown in fig. 11. When the data and clock inputs are in phase, the pulse widths of the UP and DOWN signals are the same and equal to to half the clock period. If the clock is leading the data, the width of the UP pulse becomes proportionally less. Conversley, if the clock is lagging the data, the width of the up pulse becomes proportionally more. In fig. 12 waveforms are shown for the clock in phase and the clock leading conditions. The UP and

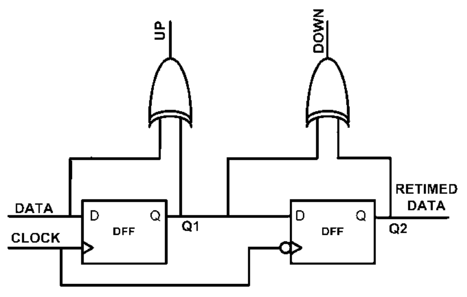

Fig. 11 Hogge Phase Detector

DOWN signals are used to steer current from a current source either into or out of an RC filter. The charge injected or removed from the capacitors of the RC network is directly proportional to the difference in widths of the UP and DOWN pulses, thereby giving the Hogge phase detector its linear property.

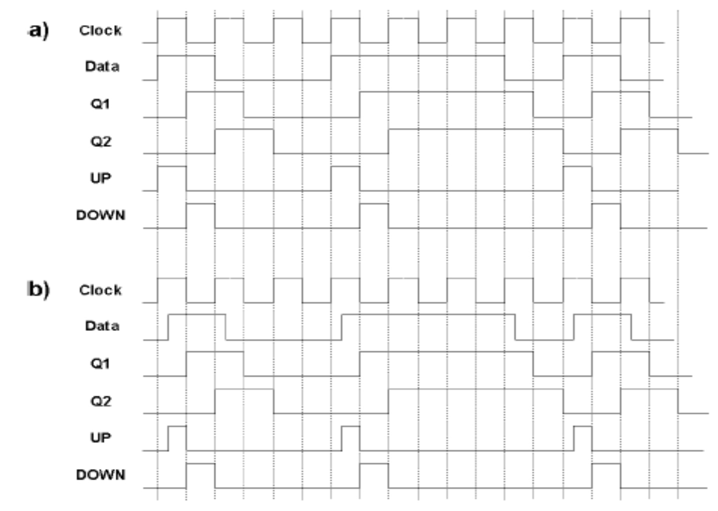

Fig. 12 Hogge Waveforms a) in phase b)clock lagging

A diagrammatic representation of a charge pump and an RC filter is shown in fig.13. Both current sources are matched to provide identical currents. Thus the net charge flowing into the filter is the net difference between the length of the UP pulse and the length of the DOWN pulse. With both current sources matched to deliver current Ich, the average charge flowing into the low pass filter during an interval of 2π radians is given by [4]

$$Iavg = \frac{\Delta\Phi in}{2\pi} \cdot Ich$$ eqn 2.15

Comparing this to an equivalent phase detector system, such as that described by eqn 2.6, whose average current would be given by a constant times the difference in phase of its input signals

$$Iavg = Kpdi \cdot (\Phi in - \Phi osc) = Kpdi \cdot \Delta\Phi in$$ eqn 2.16

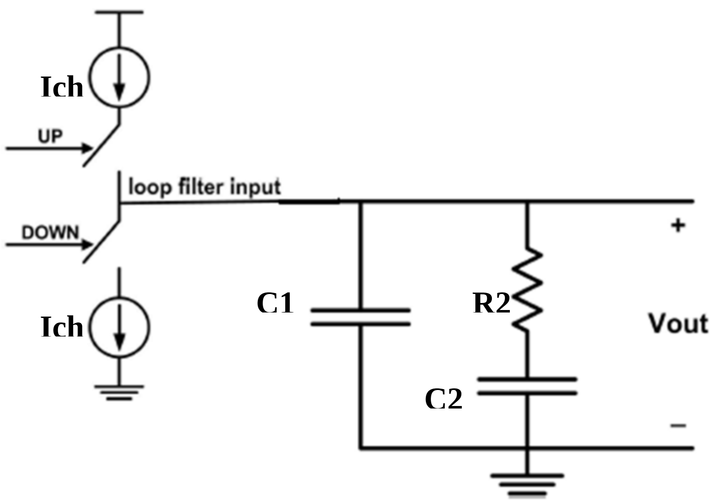

Fig.13 Charge pump and loop filter

and setting 2.16 equal to 2.15 by virtue of their defined equivalnce, we have

$$Kpdi = \frac{Ich}{2\pi}$$ eqn 2.17.

Kpdi provides the effective gain of the phase detector and charge pump. The subscript i can be dropped when it is clear which phase detector is being referenced.

The voltage Vout(s) of the loop filter is given by $Vout(s) = Iavg(s) \cdot Zlf(s)$. The transfer function of the loop filter is its impedance, since $\dfrac{Vout(s)}{Iavg(s)} = Zlf(s)$. eqn 2.18

Also, $Zlf(s) = \dfrac{sR_2C_2 + 1}{s^2R_2C_1C_2 + s(C_1 + C_2)}$ eqn 2.19

which may be written as $Zlf(s) = \dfrac{sR_2C_2 + 1}{s(C_1 + C_2)(sR_2C_P + 1)}$ eqn 2.20

where Cp is the impedance of C$_1$ in series with C$_2$ and $C_p = \dfrac{C_1C_2}{C_1 + C_2}$ eqn 2.21

C$_1$ gives rise to another pole. In this case we want this pole to be much higher than the zero, hence choose C$_1$<<C$_2$.

The open loop transfer function of the PLL with digital phase detector and charge pump, analogous to 2.14 but with 2.20 as the as the filter transfer function and 2.17 for Kpd is $TFol(s) = \dfrac{Ich \cdot (sR_2C_2 + 1) \cdot Kvco}{2\pi \cdot s\left(s^2R_2C_1C_2 + s(C_1 + C_2)\right)} = \dfrac{sR_2C_2 \cdot K_d \cdot Kvco + K_d \cdot Kvco}{s^3R_2C_1C_2 + s^2(C_1 + C_2)}$ eqn 2.22

where Kd= $Ich/2\pi$, and the closed loop transfer function is

$$H(s) = \frac{sR_2C_2K_dKvco + K_dK_{vco}}{s^3R_2C_1C_2 + s^2(C_1 + C_2) + sR_2C_2K_dK_{vco} + K_dK_{vco}} \qquad \text{eqn 2.23}$$

Both the open and closed loop functions can be readily plotted with Matlab, as shown in figs. 14 and 15. The Kvco, Ich, R and C values used to generate figs 14 and 15 come from Yang [6] and the plots are similar to those presented there. Representative values of Kvco and Ich which have been found in the literature are given in table 1.

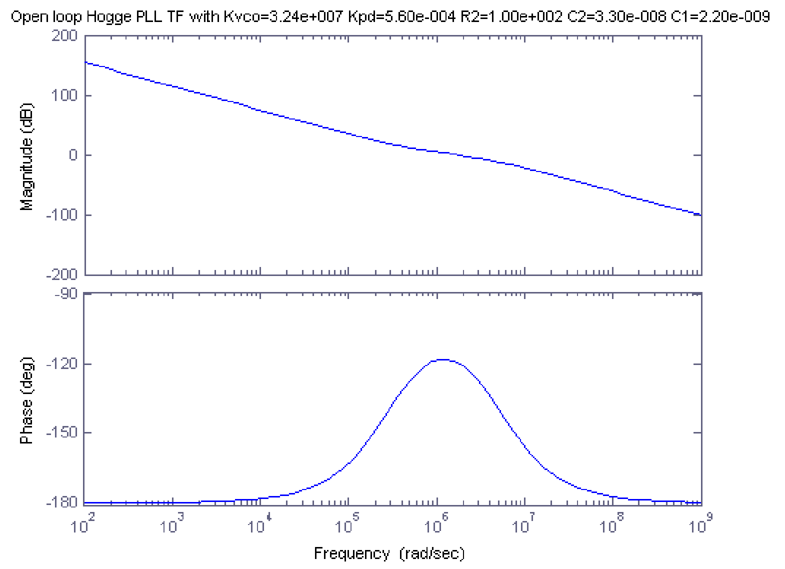

Fig. 14 Hogge PLL open loop bode plot

Ich ua	Kvco MHz/V	Ref.	CMOS Tech.
82.9	550	[5]	0.7u
	450	[7]	0.09
560	32.4	[6]	0.35
5e3		[8]	

Table 1 Ich and Kvco values

3.0 Simulink Model of The Hogge Phase Locked Loop

The PLL was modeled in more detail using the simulink feature of Matlab. The model and simulation are described in this section.

3.1 Simulink Model Description

The top level model and its test setup is shown in fig. 16. The key components of the model are the PFD (phase frequency detector), charge pump and filter, and the VCO. Since the phase detector also detects frequency differences it is referred to here as a PFD. The VCO is the continuous time VCO block which is available in the Communications Blockset from Matlab. The PFD and the charge pump and filter have been built from more elementary components A random data generator which generates random data with jitter has also been created. A reference oscillator is used to initially check the setup. Since the PFD operates best with digital signals, which are more like square waves than sine waves, the sine wave output of the VCO and of the reference oscillator are amplified and passed through saturation blocks to produce square waves. Oscilloscopes are provided so that waveforms can be examined. Details of the PFD and the charge pump with filter subsystems are shown in figs 17 and

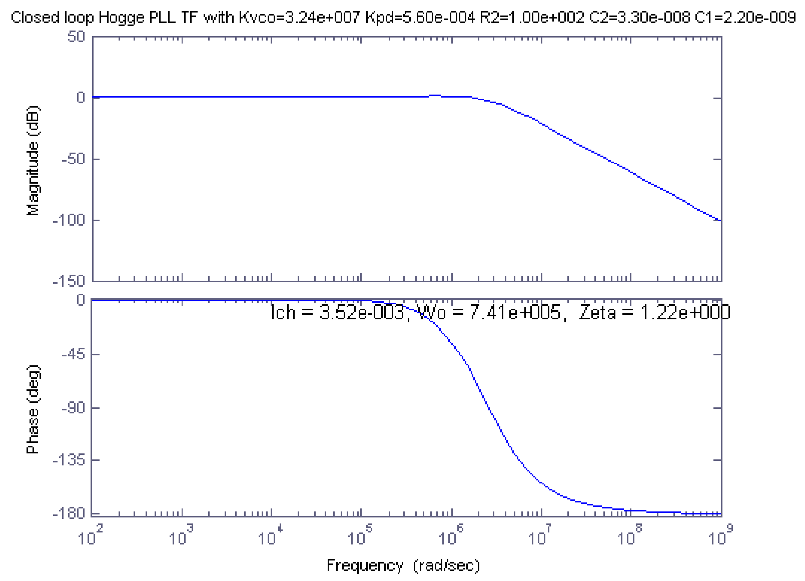

Fig. 15 Hogge closed loop bode plot

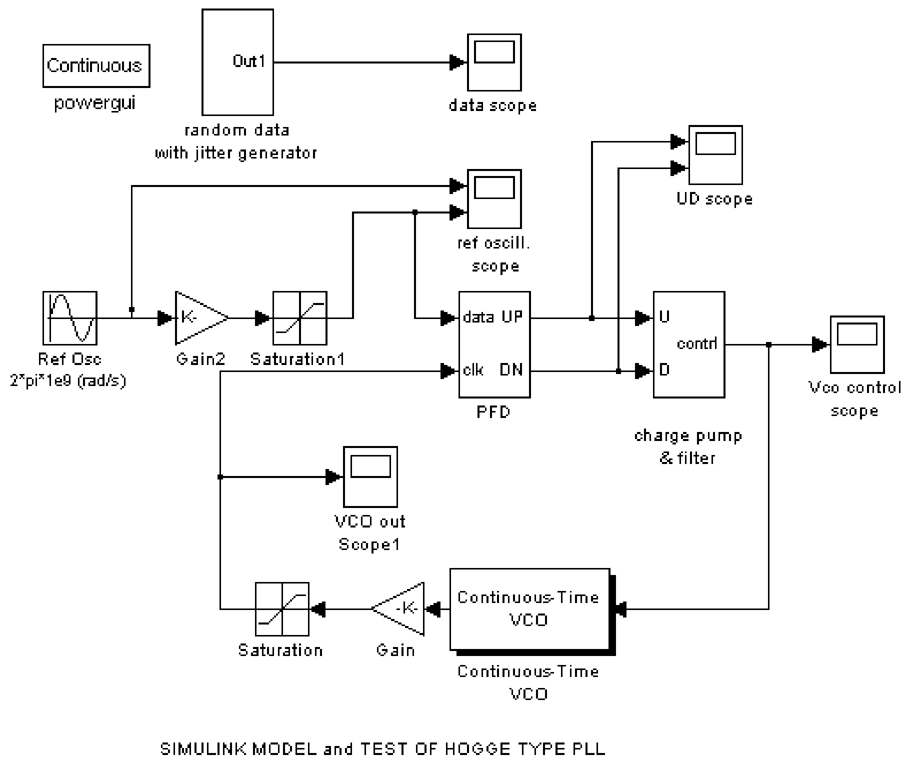

Fig. 16 Top level simulink model and test setup

18.

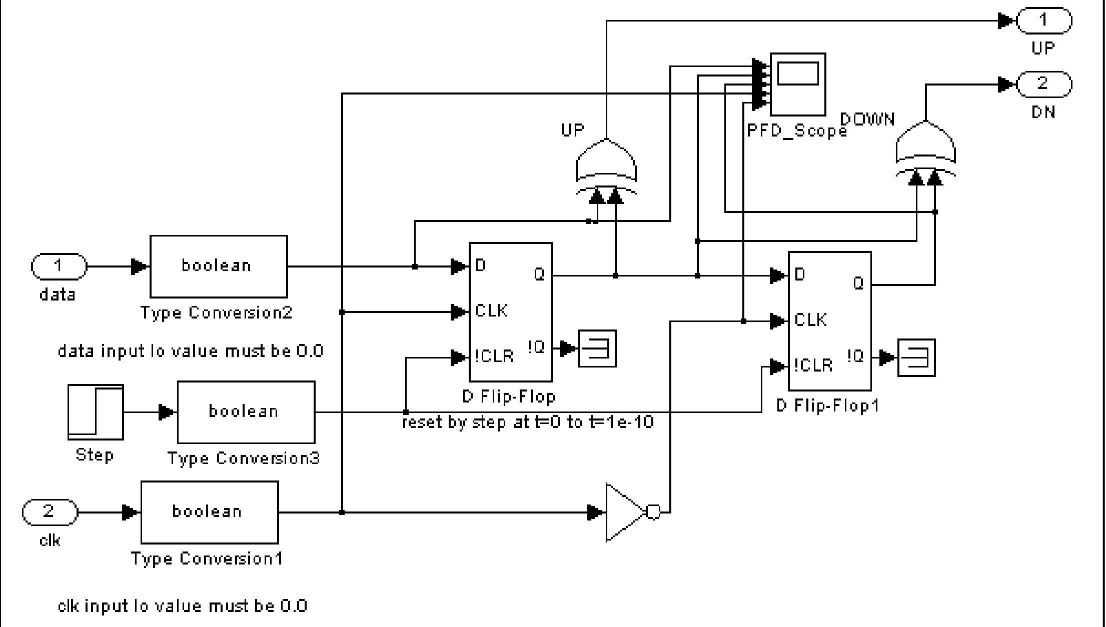

Fig. 17 PFD block, Hogge Phase Detector

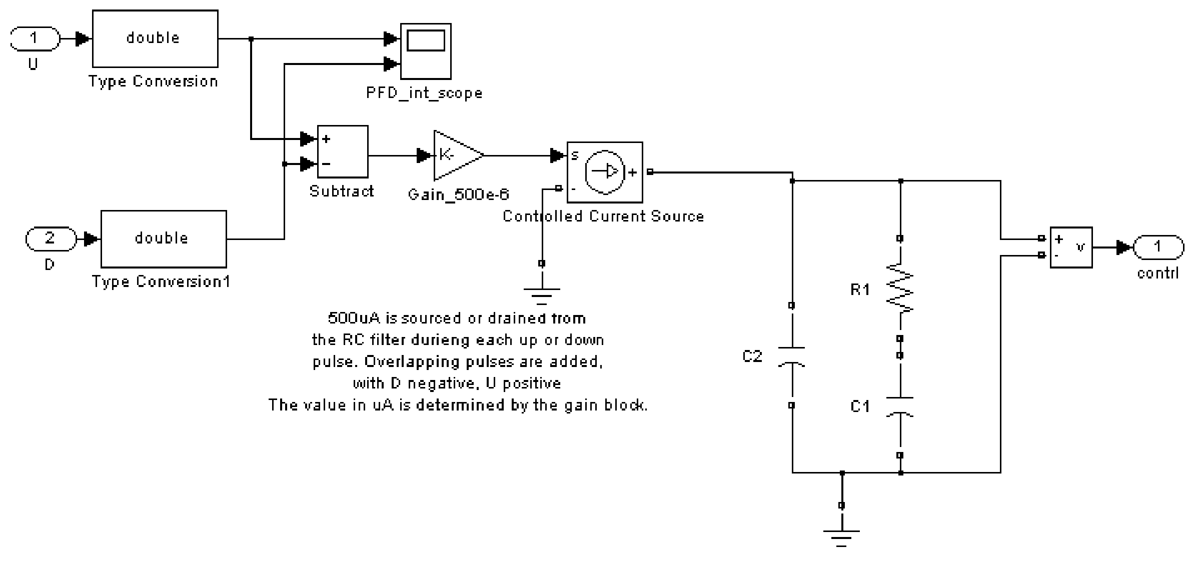
1
U
double
Type Conversion
PFD_int_scope
Subtract
Gain_500e-6
Controlled Current Source
2
D
double
Type Conversion1
500uA is sourced or drained from
the RC filter durieng each up or down
pulse. Overlapping pulses are added,
with D negative, U positive
The value in uA is determined by the gain block.
C2
R1
C1
1
contrl

Fig. 18 Charge Pump and low pass filter

The PFD consists basically of 2 flip flops and 2 exor gates. A step block is provided to initially reset the flip flops. Type conversion blocks are added as needed by simulink. An oscilloscope is provided so that detailed operation may be examined.

The charge pump uses a subtract block to subtract the D (DOWN) input from the U (UP) input and feed the resulting 1 or 0 signal to a gain block. The gain block determines the amount of current (i.e. Ich). This value is fed to a current source whose output sources current into the low pass filter. The values of R1, C1 and C2 need to be set in the workspace prior to running the simulation. Note that the references here of R1, C1, and C2 do not correspond exactly to those of fig. 13, (where the equivalent of R1 is labeled R2, C1 is labeled C2 and C2 is labeled C1).

3.2 Initial Check for Locking

With the reference oscillator connected to the data input of the PFD an initial check is performed to test that the loop achieves lock. The values used are as follows: Ich 500 ua, Kvco 500MHz/V, R1 100 Ohms, C1 1.59 nF, C2 0.1 nF. Component values were derived using the method described in [4]. The frequency of the oscillator is set to 2GHz, with initial phase of 3pi/2 radians. The frequency of the VCO is set to 1.999GHz with initial phase of –1/2pi radians. Simulation time duration (stop time) is set to 200ns. Configuration parameters are set to use the ode23tb solver and a minimum time step of 1.0e-12.

The simulation is run and the waveform of the vco control signal (input to the vco) is shown in fig. 19.

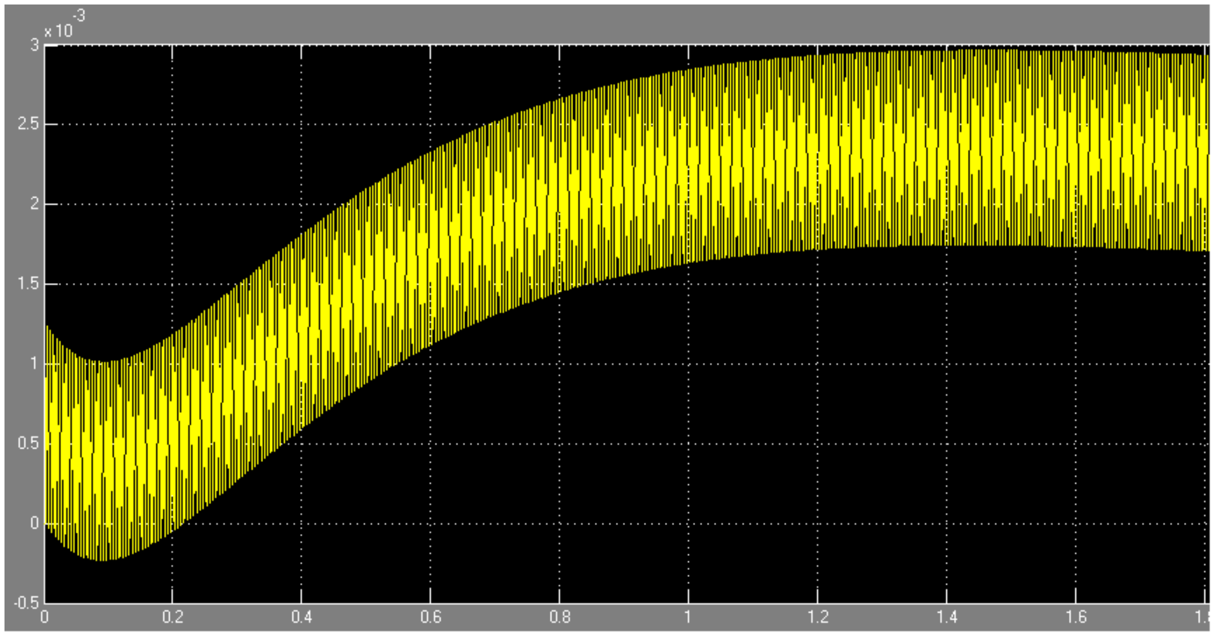

Fig 19 The VCO control input as the loop is in the process of achieving lock

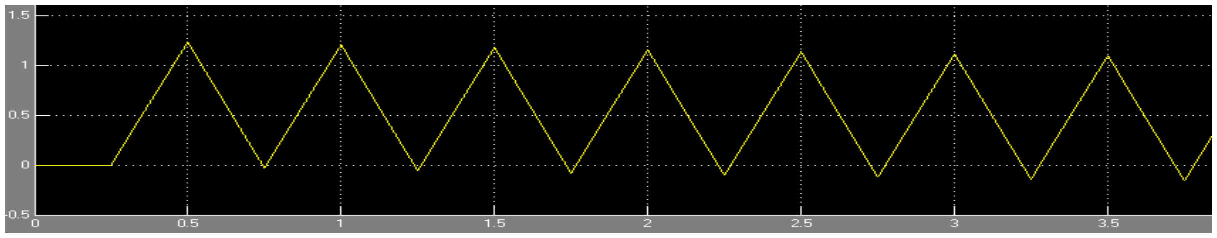

Fig. 20 Detail waveform of VCO control during locking process

From the waveform it can be seen that the loop achieves lock and that there is a slight overshoot.

3.2 Check with random data

The reference oscillator is disconnected from the PFD and the random data with jitter generator is instead connected to it. The non jittered data is also brought out from

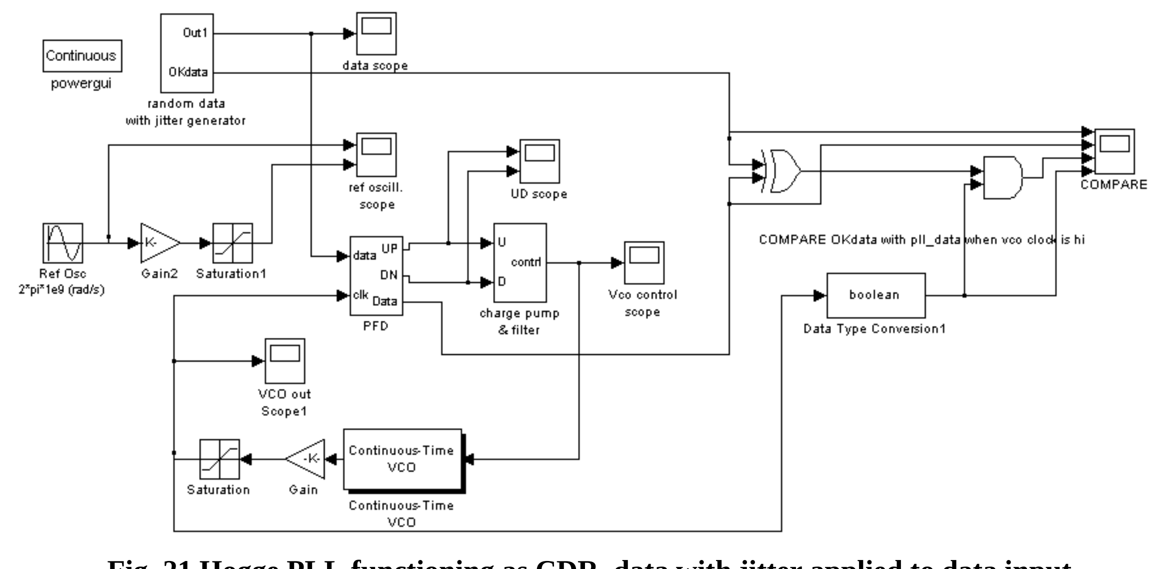

Fig. 21 Hogge PLL functioning as CDR, data with jitter applied to data input

data generator and compare logic is used to compare the recovered data with this original data. The setup is shown in fig. 21. The details of the random data with jitter generator are shown in fig. 22, and the details of the block within that, the random data with jitter source, are shown in fig 23.

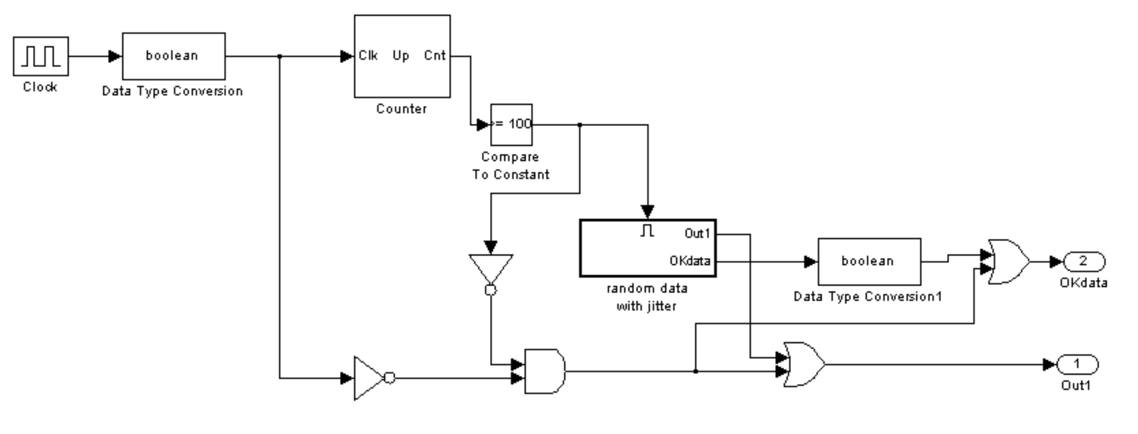

Fig. 22 Random data with jitter generator

The data generator, fig. 22, initially sends a clean data pattern devoid of jitter in order to allow the PLL to achieve lock. This is done because there are no guaranteed transitions in the random data and 8B10B bit stuffing has not yet been incorporated into the model. After a certain count is reached, the data with jitter is sent out. The value of this count is set by the constant block within the data generator. When the count is reached, the enable input of the random data source is set to true. Non jittered data is also sent out, as signal OKdata so that the recovered data can be compared to it..

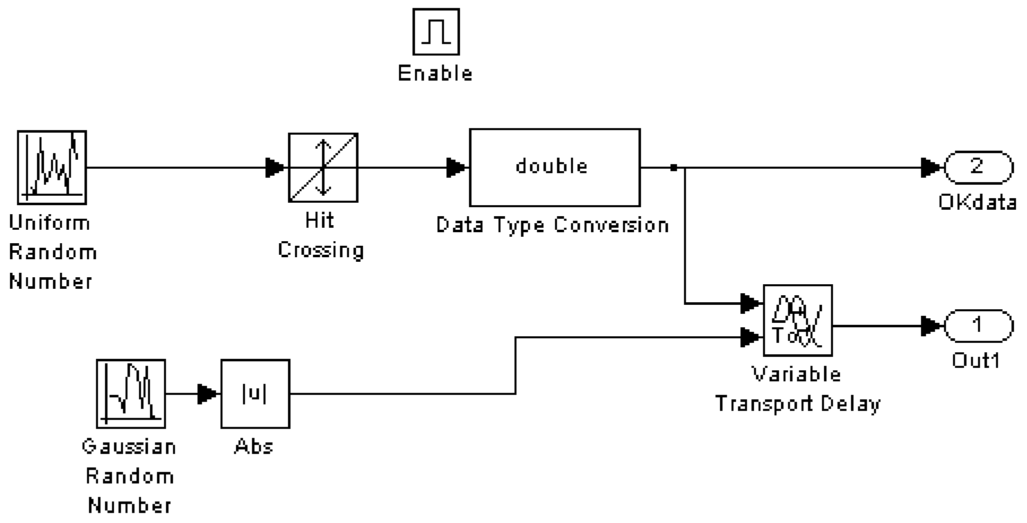

Fig. 23 Random data with jitter source

The random data with jitter source, fig. 23 uses a uniform random number generator to generate random binary bits. A Gaussian random number generator is used to add a variable delay to the edges of the data bits. Since both rising and falling edges are delayed, the delay need only be positive, which it is by virtue of the absolute value block. A transport delay block is used to add the Gaussian random delay to the random binary data stream. This models the jitter in the data. The amount of jitter can be set by setting the variance of the Gaussian random number generator. The mean is always 0. A higher variance means larger excursions from 0, hence more jitter.

According to [9] a bit error rate of 1.0e-10 represents an area which is 12.7 sigma beyond the mean. If this represents 0.55 of a unit interval as used in [10], and our unit interval is 1ns, then 1 sigma is 0.55/12.7 = 0.043ns and the variance is 1.849e-21. The variance was set to 3.0e-21 in the model.

The model was run with the compare constant set to 100, which provides 100 ns worth of non jittered data initially. The resulting waveforms are shown in figs 23 and 24.

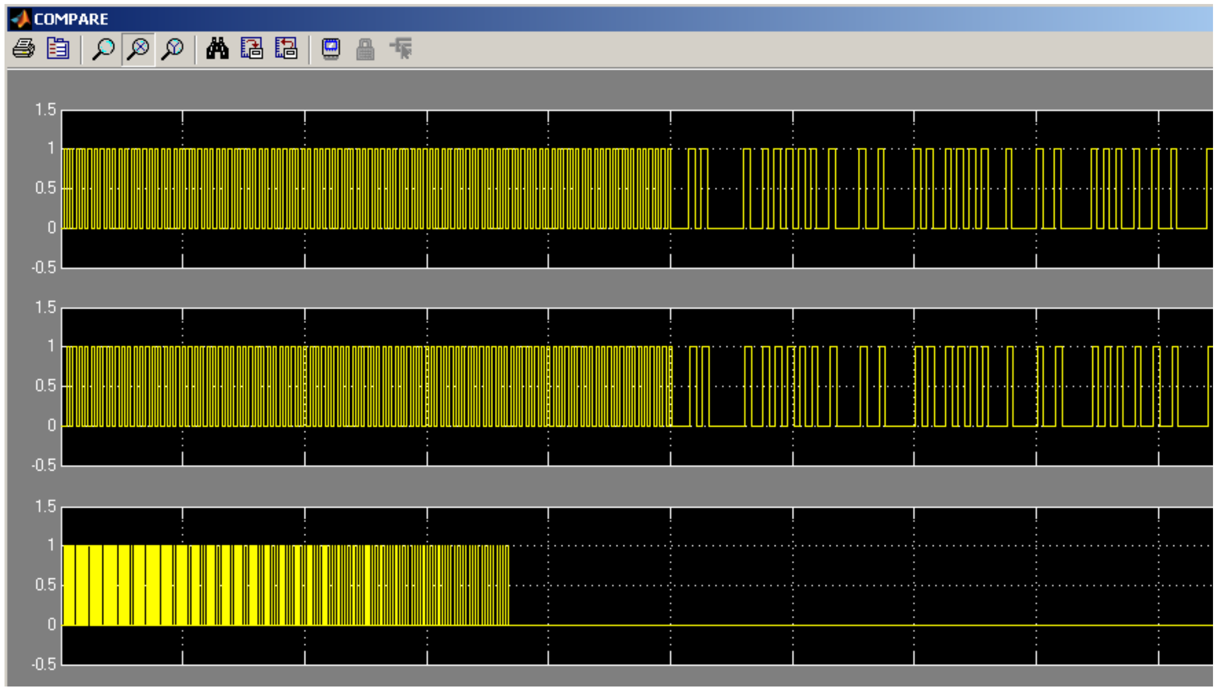

Fig. 24 COMPARE scope a) original data b) recovered data c) compare output

Fig 23 is the control voltage to the VCO and shows how the CDR tracks the jitter data after achieving lock during the first 100 ns. Fig. 24 shows the original data, the recovered data and the data error waveforms. Note that all data errors disappear after the loop achieves lock.

4.0 Summary and Conclusion

A basic model of a clock and data recovery circuit has been created using the simulink feature of Matlab. It can be used to explore quite rapidly various aspects and modes of operation of such a circuit. The results obtained have been compared to published data and the model appears reasonable.

Matlab has also been used to generate line code waveforms and their frequency spectra and also bode plots. It has a rich feature set which provides for quick and efficient

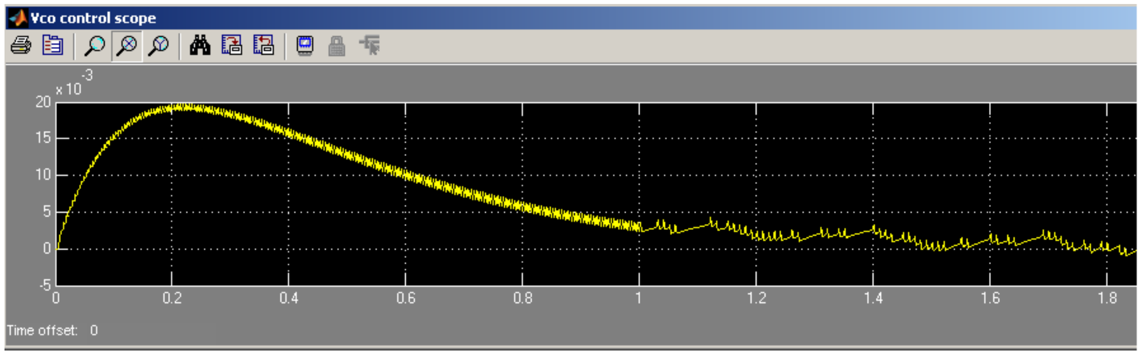

Fig. 23 Hogge PLL as CDR; VCO control initially attains lock, then tracks jittered data

modeling of a variety of systems. Only a few of its features have been used during this work.

The model itself could be further developed to incorporate other aspects of cdr circuits, such as 8b10b bit stuffing and to provide features such as plots of eye diagrams. Additional user features could be added which would allow, for example, quick updates of the variance parameter or setting the jitter at a defined value. Overall, the model has not been thoroughly tested for robusteness.

References

1. Simon Haykin, "Digital Communications", Wiley 1988.
2. Rich Taborek et al., IEEE 802.3ae Task Force Presentation, May 23 2000 on the web at: http://grouper.ieee.org/groups/802/3/ae/public/may00/taborek_1_0500.pdf
3. Roland Best, "Phase Locked Loops", McGraw-Hill 2007.
4. David A. Johns, Ken Martin "Analog Integrated Circuit Design", Wiley 1997
5. M. Burzio, P.Pellegrino, "A high speed 0.7u CMOS PLL circuit for clock/data recovery in interconnection systems", European Solid State Circuits Conference (ESSCIRC), 1996.
6. Ching-Yuan Yang, Shen-Iuan Liu, "Fast-Switching Frequency Synthesizer with a Discriminator-Aided Phase Detector", IEEE Journal of Solid State Circuits, Oct. 2000
7. W. Rhee et al., "A Continuously Tunable LC-VCO PLL with Bandwidth Linearization technique for PCI Express Gen-2 Applications", Journal of Semiconductor Science and Technology, Sept. 2008
8. National Semiconductor, Application Note 1001, July 2001
9. M.H. Perott et al. "A 2.5 Gb/s Multirate 0.25um CDR Utilizing A Hybrid Analog/Digital Loop Filter", IEEE International Solid State Circuits Conference, Feb 2006
10. Ken Kundert "Modeling Jitter in PLL based Frequency Synthesizers", on the web at www.designers-guide.com, August 2006

Appendix

Running the Matlab code and the simulink model.

A list of the files provided on the CD and how to use them is given below. Open Matlab and follow the instructions. The simulink models use a variety of items from different blocksets.

1.) UNRZ_RZ.m
- right click on the file an select run
  the unipolar NRZ and RZ line codes are displayed

2.) power_spectrum_nrz_rz.m
- right click on the file and select run
  the power spectra are displayed

3.) plot_bode_open_and_closed_1.m
- right click on the file and select run
  the open and close loop bode plots of the second order function, as in figs 8 and 9 are displayed

4.) plot_bode_open_and_closed_hogge.m
- right click on the file and select run
  the open and close loop bode plots of the third order function, as in figs 14 and 15 are displayed

5.) pll_hogge_lock.mdl
- this is a simulink model; open the model with simulink and set the configuration parameters to use the ode23tb solver, simulation stop time to 200ns, maximum size of step time to 1.0e-12 and relative tolerance to 1e-4 . Close any oscilloscope windows which pop up. In the workspace window set R1=100, C1=1.59e-9 and C2=0.1e-9. Select the RUN button from simulink to run the model. Execution takes ~1 minute.

  After the run completes, click on the Vco control scope icon to view the waveform.

6.) pll_hogge_cdr.mdl
- this is a simulink model; open the model with simulink and set the configuration parameters to use the ode23tb solver, simulation stop time to 200ns, maximum size of step time to 1.0e-12 and relative tolerance to 1e-4 . Close any oscilloscope windows which pop up. In the workspace window set R1=100, C1=1.59e-9 and C2=0.1e-9. Select the RUN button from simulink to run the model. Execution takes ~1 minute.

  After the run completes, click on the Vco control scope icon and the COMPARE scope icon to view the waveforms.