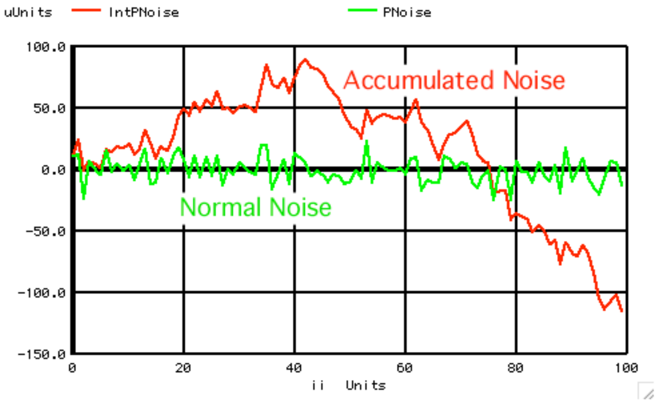
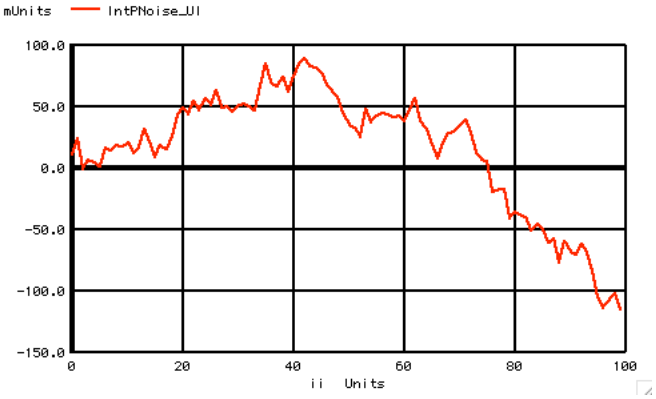
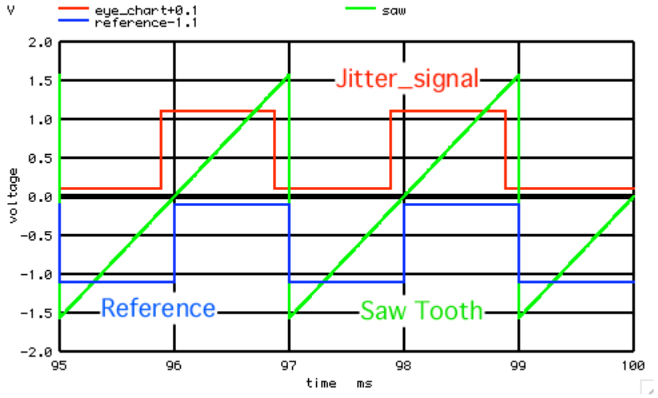
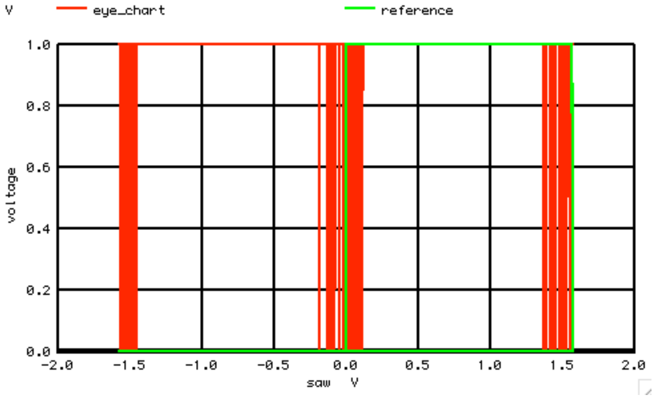
=====================Create/Display_Jitter_In_Spice=====================================

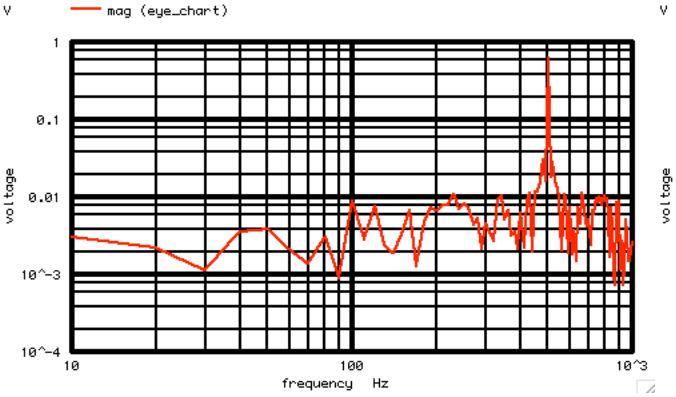
1) **Jitter is an accumulation random process in which each random point is summed with the sum of all previous random points.**
2) **Such randomness can be generated in a array and applied to a voltage sources as a piece wise linear input signal to generate the jitter signal.**
3) **A saw tooth wave form is easy to generate.**
4) **Plotting the jitter signal versus the saw tooth will generate the desired eye pattern.**

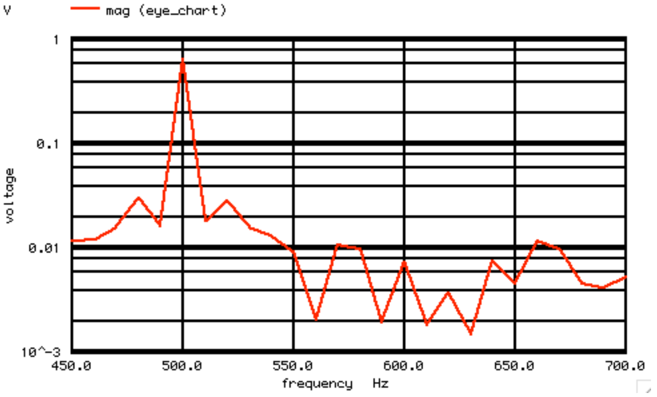












```
Create/Display_Jitter_tests
*================Need_voltage_Sources_to_alter_with_PWL_Data===================="
VT      Vtime      0        dc    0    PWL  ( 0 0 1 1 )
B1      SAW        0        V = atan(tan(3.14159*500*v(Vtime)))
V1      V1         0        dc    0
V2      V2         0        dc    0
V3      V3         0        dc    0
.control
set               pensize = 2
echo              "===================Want_100_1ms_steps======================="
let n =           100
let Nlev =        127
let tstep =       1ms
let Nrnd =        8
let Nbins =       Nlev*Nrnd
echo              "random levels      0-> $&Nlev"
echo             "Numb rnd waveforms $&Nrnd"
echo             "===================Create_PHaseNoise_array==================="
let PNoise =     vector($&n)
let IntPNoise =  vector($&n)
let ii =         vector($&n)
let index =      0
repeat           $&n
let              PNoise[index] = 10u*(rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)+rnd(127)-507.5)/102.879
let index =      index + 1
end
*plot            PNoise  vs ii
let averVal =    mean(PNoise)
let noisAC =     PNoise - averVal
let RmsVal =     sqrt(mean(noisAC* noisAC))
echo             "Average level      $&averVal"
echo             "RMS level          $&RmsVal"
echo             "===================Create_Histogram_Bins====================="
let binsN        = vector($&Nbins)
let binPNoise    = vector($&Nbins)*0
let binIntPNoise = vector($&Nbins)*0

echo             "Number  Bins       0-> $&Nbins"
echo             "===================Histogram_PNoise=========================="
let index =      0
let hist =       0
repeat           $&n
let indexb =     0
let PNoiseH =    PNoise*102.879/10u +507.5
repeat           $&Nbins
let hist =       PNoiseH[index]
if               (hist < indexb +.3    &  hist > indexb -.3)
let              binPNoise[indexb] =  binPNoise[indexb]  + 1
endif
let indexb  =    indexb + 1
end
let index =      index + 1
end
let              binsNScale = 10u/102.879
let              binsNAveScale = 507.5*binsNScale
let PNoise_V =   binsN*binsNScale-binsNAveScale
*plot            binPNoise vs binsN
*echo            "plot                binPNoise vs binsN"
plot             binPNoise vs PNoise_V
echo             "plot               PNoise_bin vs PNoise_V"
echo             "===================Create_Integrated_PHaseNoise_array==========="
let              IntPNoise[0] = PNoise[0]
let index =      1
let nb =         n-1
repeat           $&nb
let              IntPNoise[index] = IntPNoise[index-1] +PNoise[index]
let index =      index + 1
end
let averVal =    mean(IntPNoise)
let noisAC =     IntPNoise - averVal
let RmsVal =     sqrt(mean(noisAC* noisAC))
echo             "Average level      $&averVal"
echo             "RMS level          $&RmsVal"
plot             IntPNoise PNoise vs ii
let IntPNoise_UI = IntPNoise/tstep
plot             IntPNoise_UI vs ii
echo             "===================Create_PWL_arrays==========================="
let pwl_1 =      vector(4*n)*tstep
let pwl_2 =      vector(4*n)*tstep
let pwl_3 =      vector(4*n)*tstep
let n2 =         n/2
echo             "===================Make_the_jitter_PWL_array===================="
let              pwl_1[0] = 0
let              pwl_1[1] = 0
let              pwl_1[2] = 1u
let              pwl_1[3] = 1
let              pwl_1[4] = tstep -1u
let              pwl_1[5] = 1
let              pwl_1[6] = tstep
let              pwl_1[7] = 0
let n2 =         n/2-1
let index =      1
repeat           $&n2
let              pwl_1[0+8*index] = pwl_1[-2+8*index] +tstep -1u +PNoise[2*index-1]
let              pwl_1[1+8*index] = 0
let              pwl_1[2+8*index] = pwl_1[-2+8*index] +tstep +PNoise[2*index-1]
let              pwl_1[3+8*index] = 1
let              pwl_1[4+8*index] = pwl_1[2+8*index] + tstep -1u +PNoise[2*index]
let              pwl_1[5+8*index] = 1
let              pwl_1[6+8*index] = pwl_1[2+8*index] + tstep +PNoise[2*index]
let              pwl_1[7+8*index] = 0
let index =      index + 1
end
echo             "===================Make_a_nonjitter_PWL_array===================="
let              pwl_2[0] = 0
let              pwl_2[1] = 0
let              pwl_2[2] = 1u
let              pwl_2[3] = 1
let              pwl_2[4] = tstep -1u
let              pwl_2[5] = 1
let              pwl_2[6] = tstep
let              pwl_2[7] = 0
let n2 =         n/2-1
let index =      1
repeat           $&n2
let              pwl_2[0+8*index] = pwl_2[-2+8*index] +tstep -1u
let              pwl_2[1+8*index] = 0
let              pwl_2[2+8*index] = pwl_2[-2+8*index] +tstep
let              pwl_2[3+8*index] = 1
let              pwl_2[4+8*index] = pwl_2[2+8*index] + tstep -1u
let              pwl_2[5+8*index] = 1
let              pwl_2[6+8*index] = pwl_2[2+8*index] + tstep
let              pwl_2[7+8*index] = 0
let index =      index + 1
end
echo             "===================Make_a_EdgeError_PWL_array===================="
let index =      0
let n3  =        n2 +1
repeat           $&n3
let              pwl_3[0+8*index] = pwl_2[0+8*index]
let              pwl_3[1+8*index] = pwl_2[0+8*index] -pwl_1[0+8*index]
let              pwl_3[2+8*index] = pwl_2[2+8*index]
let              pwl_3[3+8*index] = pwl_2[2+8*index] -pwl_1[2+8*index]
let              pwl_3[4+8*index] = pwl_2[4+8*index]
let              pwl_3[5+8*index] = pwl_2[4+8*index] -pwl_1[4+8*index]
let              pwl_3[6+8*index] = pwl_2[6+8*index]
let              pwl_3[7+8*index] = pwl_2[6+8*index] -pwl_1[6+8*index]
let index =      index + 1
end
echo             "===================Install_the_PWL_arrays=========================="
alter            @v1[pwl] = pwl_1
alter            @v2[pwl] = pwl_2
alter            @v3[pwl] = pwl_3
echo             "===================Run_and_Plot====================================="
tran             .05m   100m   0  3u
let edge_errorUI = v3/1m
plot             edge_errorUI
let eye_chart    = v1
let reference    = v2
plot             eye_chart+0.1  saw reference-1.1
plot             eye_chart+0.1  saw reference-1.1 xlimit 95m 100m
plot             eye_chart reference vs saw
echo             "plot               eye_chart reference vs saw"
echo             "===================FFT_and_Plot====================================="
linearize
let              FFT_BandWidth_Hz =  1K
let              FFT_resolution_Hz = 10
echo             "FFT_BandWidth_Hz=  $&FFT_BandWidth_Hz"
echo             "FFT_resolution_Hz= $&FFT_resolution_Hz"
*set             specwindow =         "hanning"
set              specwindow =         "rectangular"
spec             $&FFT_resolution_Hz  $&FFT_BandWidth_Hz   $&FFT_resolution_Hz     v(eye_chart)
plot             mag (eye_chart)   loglog
plot             mag (eye_chart)  loglog xlimit 490 700

plot             mag (eye_chart)  ylog xlimit 490 700
echo             "plot               fft eye_chart"
echo             "===================Done============================================="
.endc
.end
```

4.4.11_11.30AM
dsauersanjose@aol.com
Don Sauer