# Analysis of CO $^{60}\beta$-source data. Data set consists of 1805 time stamped waveforms recorded at 145 GSa/s.

In[1]:=
```
SetDirectory["~white/Desktop/4mm^2"];
Drop[FileNames[], 1] // Length;
ntrace = %
```

Out[3]= 1790

In[4]:=
```
Namelist = Drop[FileNames[], 1];
time = Transpose[Import[Namelist[[1]], "Data"]][[1]];
nbins = Dimensions[time][[1]]
```

Out[6]= 15 986

In[7]:=
```
trange = time[[nbins]] - time[[1]];
t[bin_] := time[[1]] + trange * (bin) / nbins
bint[tvar_] := (tvar - time[[1]]) * nbins / trange
```

In[10]:=
```
noisemax = 7500;
Print["Do noise analysis for the first  ",
  (t[noisemax] - t[1]) × 10^9, " nanoseconds"]
front = Transpose[Import[Namelist[[1]], "Data"]][[2]];
back = Transpose[Import[Namelist[[1]], "Data"]][[3]];
offset = Mean[Take[front, noisemax]] // EngineeringForm
```

Do noise analysis for the first  46.8658 nanoseconds

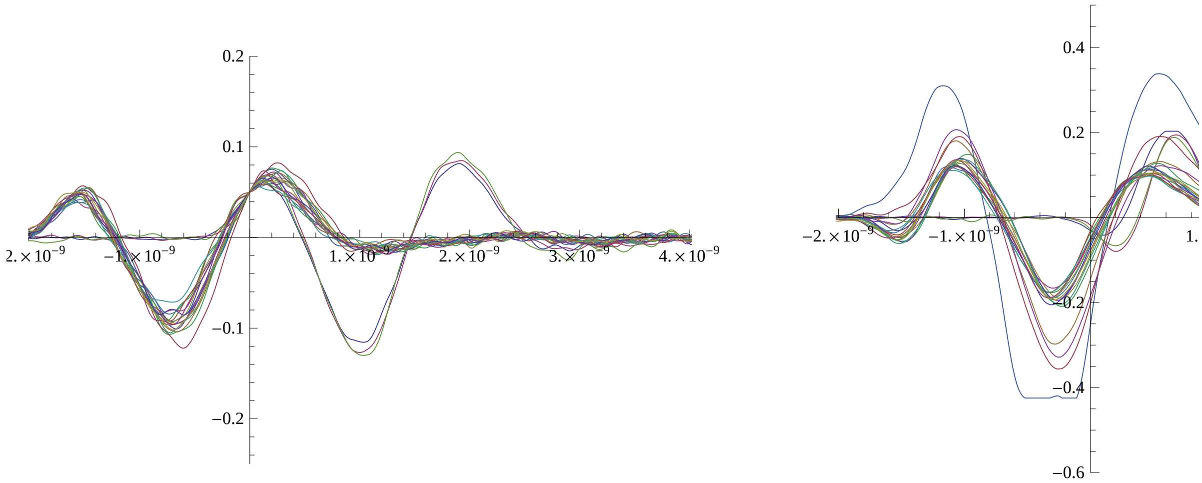
Out[14]//EngineeringForm=

$-428.374 \times 10^{-6}$

Front and Back are the 2 APDs which face eachother. Front is closest to the CO $^{60}$ source. Back triggers the scope.
  The first 47 nanoseconds are used to extract the baseline correction which is typically ~ -.5 mV.

## Inspect some waveforms

In[15]:=
```
GraphicsRow[{ListPlot[Table[Transpose[{Transpose[Import[Namelist[[i]], "Data"]][[1]],
      Transpose[Import[Namelist[[i]], "Data"]][[3]]}], {i, 20}],
   PlotRange → {{-0.000000002, .000000004}, {-.25, .2}}, Joined → True],
  ListPlot[Table[Transpose[{Transpose[Import[Namelist[[i]], "Data"]][[1]],
      Transpose[Import[Namelist[[i]], "Data"]][[2]]}], {i, 20}],
   PlotRange → {{-0.000000002, .000000004}, {-.6, .5}}, Joined → True]}, ImageSize → 600]
```

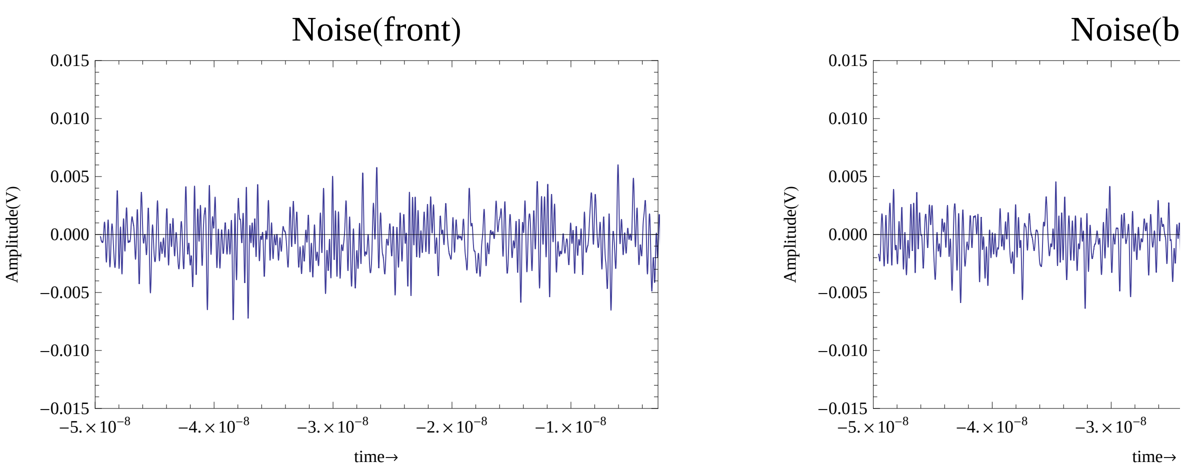


In[16]:=
```
tnoisemax = t[noisemax]
```

Out[16]= $-2.68056 \times 10^{-9}$

In[17]:=
```
GraphicsRow[{ListPlot[Transpose[{time, front}], Joined → True, Frame → True,
    FrameLabel → {{"Amplitude(V)",}, {"time→", "Noise(front)"}},
    PlotRange → {{-0.00000005, tnoisemax}, {-.015, .015}}],
  ListPlot[Transpose[{time, back}], Joined → True, Frame → True,
    FrameLabel → {{"Amplitude(V)",}, {"time→", "Noise(back)"}},
    PlotRange → {{-0.00000005, tnoisemax}, {-.015, .015}}]}, ImageSize → 600]
```

Out[17]=




In[18]:=
```
dt = time[[2]] - time[[1]] // EngineeringForm
f = 1 / (time[[2]] - time[[1]]);
fftrange = f / 2
```

Out[18]//EngineeringForm=

$6.2 \times 10^{-12}$

Out[20]= $8.06452 \times 10^{10}$

# RMS noise and baseline

In[21]:=
```
basef = {}; noisef = {}; baseb = {}; noiseb = {}; front1 = {}; back1 = {};
fbin = {}; bbin = {};
iwave = 1000;
```

In[24]:=
```
Do[
 front1 = Take[Transpose[Import[Namelist[[itrace]], "Data"]][[2]], noisemax];
 back1 = Take[Transpose[Import[Namelist[[itrace]], "Data"]][[3]], noisemax];
 AppendTo[basef, Mean[front1]];
 AppendTo[baseb, Mean[back1]];
 AppendTo[noisef, RootMeanSquare[front1]]; AppendTo[noiseb, RootMeanSquare[back1]];
 , {itrace, iwave}]
```

In[25]:=
```
avenoisef = Mean[noisef]
avenoiseb = Mean[noiseb]
```

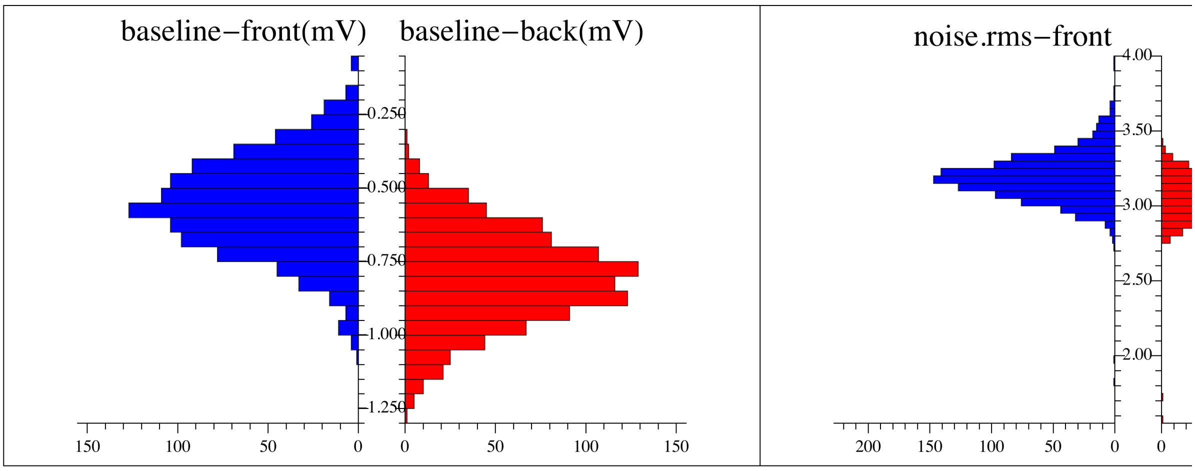
Out[25]= 0.00319179

Out[26]= 0.00305551

```
Dimensions[basef]
GraphicsRow[
 {PairedHistogram[1000 * basef, Style[1000 * baseb, Red], ChartBaseStyle → Blue,
   ChartLabels → {"baseline-front (mV) ", "baseline-back (mV) "}],
  PairedHistogram[1000 * noisef, Style[1000 * noiseb, Red], ChartBaseStyle → Blue,
   ChartLabels → {"noise.rms-front     ", "      -back (mV) "}]},
 Frame → All, ImageSize → 600]
```

Out[27]= {1000}

Out[28]=




In[29]:=
```
front1 = Take[Transpose[Import[Namelist[[1]], "Data"]][[2]], noisemax];
back1 = Take[Transpose[Import[Namelist[[1]], "Data"]][[3]], noisemax];
time1 = Take[Transpose[Import[Namelist[[1]], "Data"]][[1]], noisemax];
```

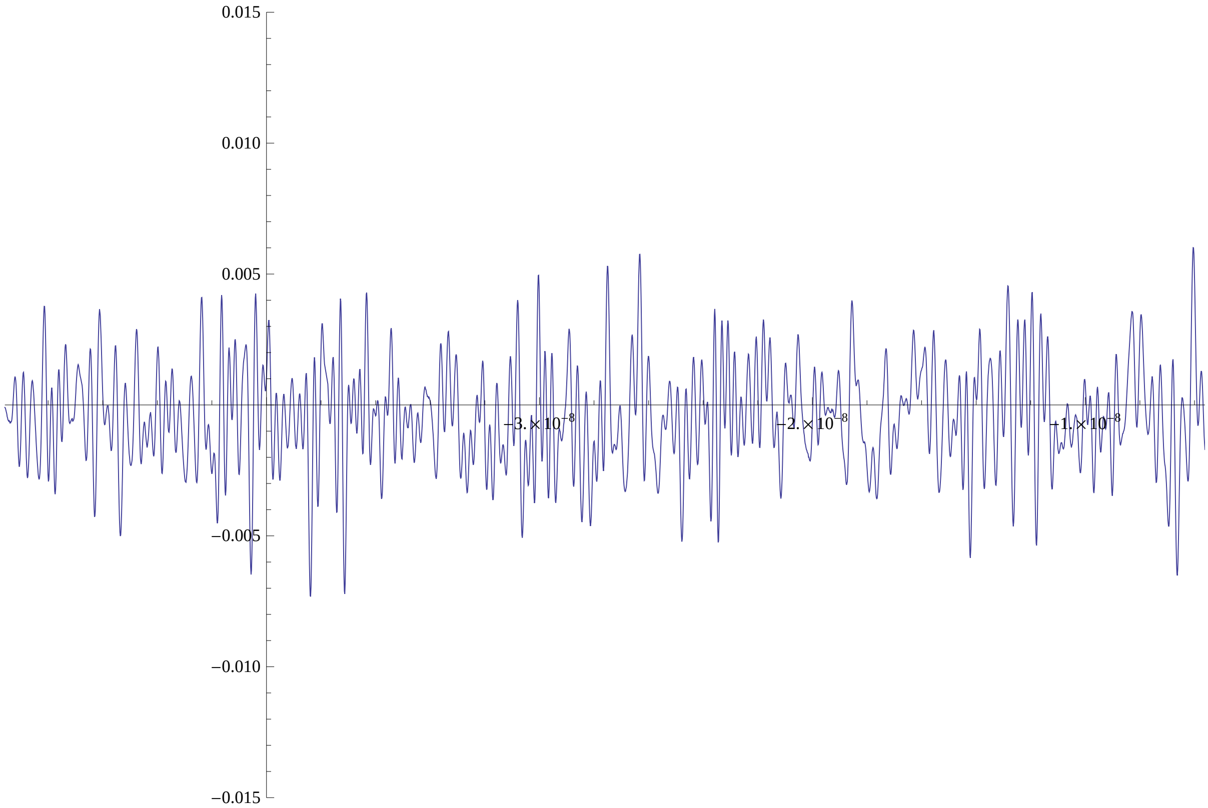
In[32]:=
```
Ff = Interpolation[Transpose[{time1, front1}], Method → "Spline"];
```

In[33]:=
```
Fb  = Interpolation[Transpose[{time1, back1}], Method -> "Spline"]
```

Out[33]= InterpolatingFunction[{{−4.95526 × 10^-8, −2.6839 × 10^-9}}, <>]

In[34]:=
```
Plot[Ff[x], {x, -4.9586 × 10^-8, -3 × 10^-9},
 PlotRange → {{-4.9586 × 10^-8, -3 × 10^-9}, {-.015, .015}}]
```

Out[34]=




In[35]:=
```
fftout1 = Abs[ FourierDCT[front1]];
df = fftrange / noisemax;
```

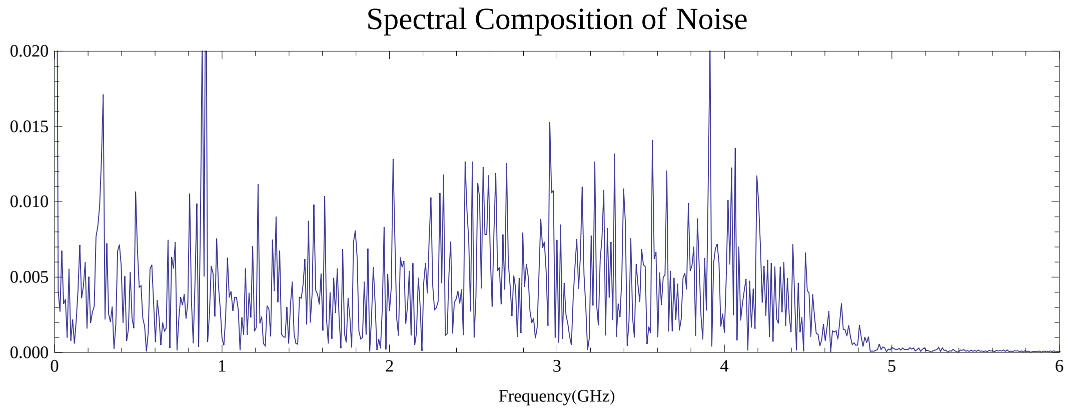
In[37]:=
```
freq = Range[df, fftrange, df] / 1 000 000 000.;
```

In[38]:=
```
fftnew = Transpose[{freq, fftout1}];
```

For this scope the Sampling frequency is much higher than the actual bandwidth. Therefore in the plots below there is a white noise distribution out to a few GHz. Above this (the scope bandwidth) there is an apparent reduction in noise. But in this region data are really reflecting the fact that high sampling frequency is really a figment of ' on - chip " interpolation.

```
In[39]:= ListPlot[fftnew, Joined → True, AspectRatio → 0.3, ImageSize → Large, Frame → True,
  FrameLabel → {{, }, {"Frequency(GHz)", "Spectral Composition of Noise"}},
  PlotRange → {{0, 6}, {0, .02}}]
```

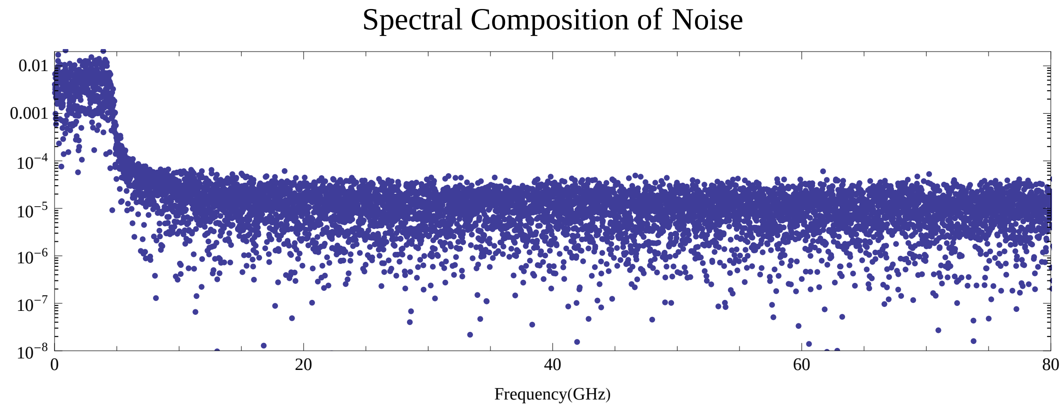
Out[39]=




This has zero frequency in element 1. The 7693- th element corresponds to 1/2 the sampling frequency.After that aliassing takes over and the frequency heads back to zero.

```
In[40]:= ListLogPlot[fftnew, AspectRatio → 0.3, Frame → True,
  FrameLabel → {{, }, {"Frequency(GHz)", "Spectral Composition of Noise"}},
  ImageSize → Large, PlotRange → {{0, 80}, {10^-8, .02}}]
```

Out[40]=



# Initialize useful bin locations for region of interest and location of first peak in the waveform

In[219]:=
```
peakbin = IntegerPart[bint[-1.5 × 10^-9]];
startbin = IntegerPart[bint[-2 × 10^-9]];
endbin = IntegerPart[bint[3 × 10^-9]] - 1;
timestrip = Take[time, {startbin, endbin}];
```

In[223]:=
```
Clear[backstrip]; Clear[frontstrip];
Clear[backs]; Clear[fronts];
frontstrip = {}; backstrip = {}; isaved = 0; backs {}; fronts {};
```

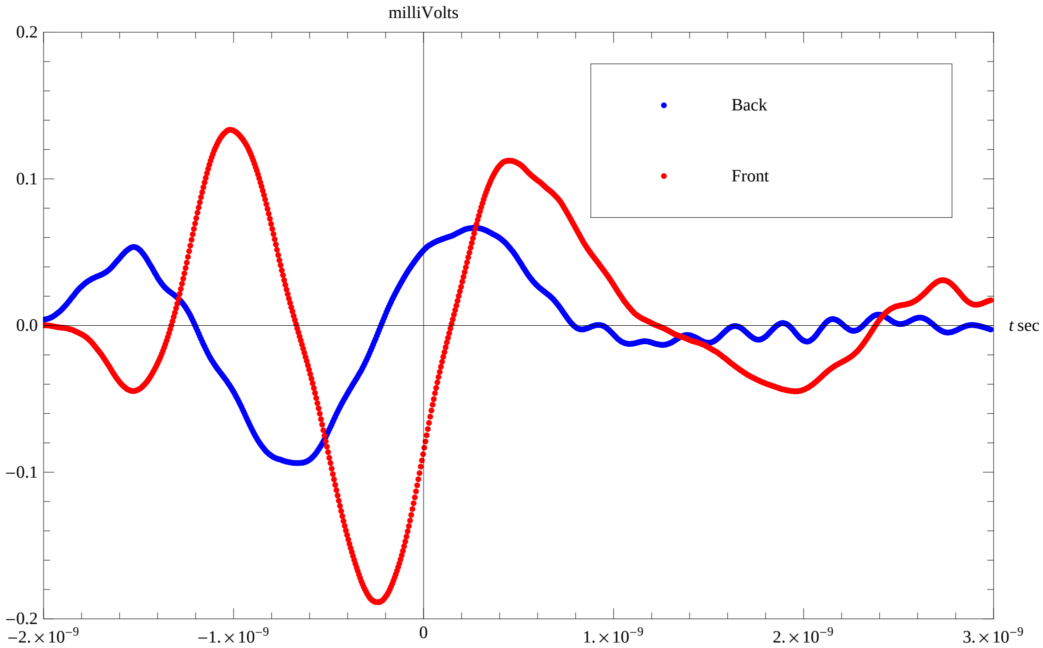
In[226]:=
```
Do[
 frontraw = Transpose[Import[Namelist[[i]], "Data"]][[2]] - basef[[i]];
 (*baseline restoration on waveforms using average level of 1st 46.8 nsec*)
 backraw = Transpose[Import[Namelist[[i]], "Data"]][[3]] - baseb[[i]];
 If[backraw[[peakbin]] < 0.02, Goto["skipwave"], isaved++];
 (*first analyze events which trigger on the 2nd peak*)
 backs = Take[backraw, {startbin, endbin}];
 fronts = Take[frontraw, {startbin, endbin}];
 AppendTo[backstrip, backs];
 AppendTo[frontstrip, fronts];
 Label["skipwave"];
 , {i, iwave}]
```

In[227]:=
```
Dimensions[backstrip]
Dimensions[frontstrip]
```

Out[227]= {653, 800}

Out[228]= {653, 800}

In[217]:=
```
Needs["PlotLegends`"]
ListPlot[{Transpose[{timestrip, backstrip[[1]]}],
   Transpose[{timestrip, frontstrip[[1]]}]}, PlotStyle → {Blue, Red},
 PlotRange → {{-2 × 10^-9, .3 × 10^-8}, {-.2, .2}}, AxesLabel → {t (sec), milliVolts},
 PlotLegend → {"Back", "Front"}, Frame → True, ImageSize → 600]
```




.

In[229]:=
```
Wf = ConstantArray[0, 1000]; Wb = ConstantArray[0, 1000];
```

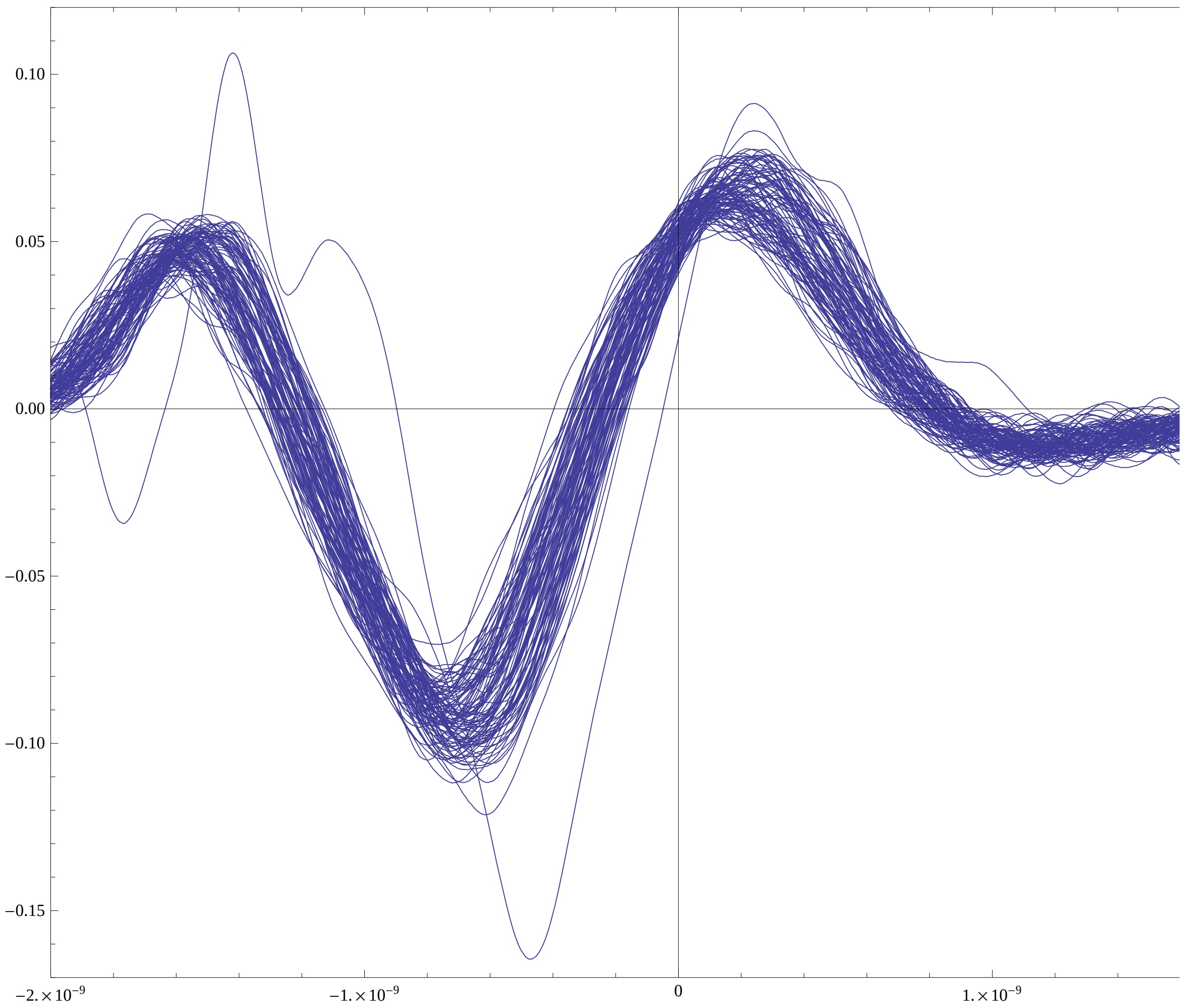
In[230]:=
```
Do[
Wf[[itrace]] =
  Interpolation[Transpose[{timestrip, frontstrip[[itrace]]}], Method → "Spline"];
 Wb[[itrace]] = Interpolation[Transpose[{timestrip, backstrip[[itrace]]}],
   Method → "Spline"];
 , {itrace, isaved}]
```

In[244]:=
```
Clear[Zeroesb]; Clear[Zeroesf];
Zeroesb = x /. FindRoot[Wb[[#]][x], {x, -1.2 × 10^-9}] & /@ Range[isaved];
```

In[246]:=
```
Zeroesf = x /. FindRoot[Wf[[#]][x], {x, -0.7 × 10^-9}] & /@ Range[isaved];
```

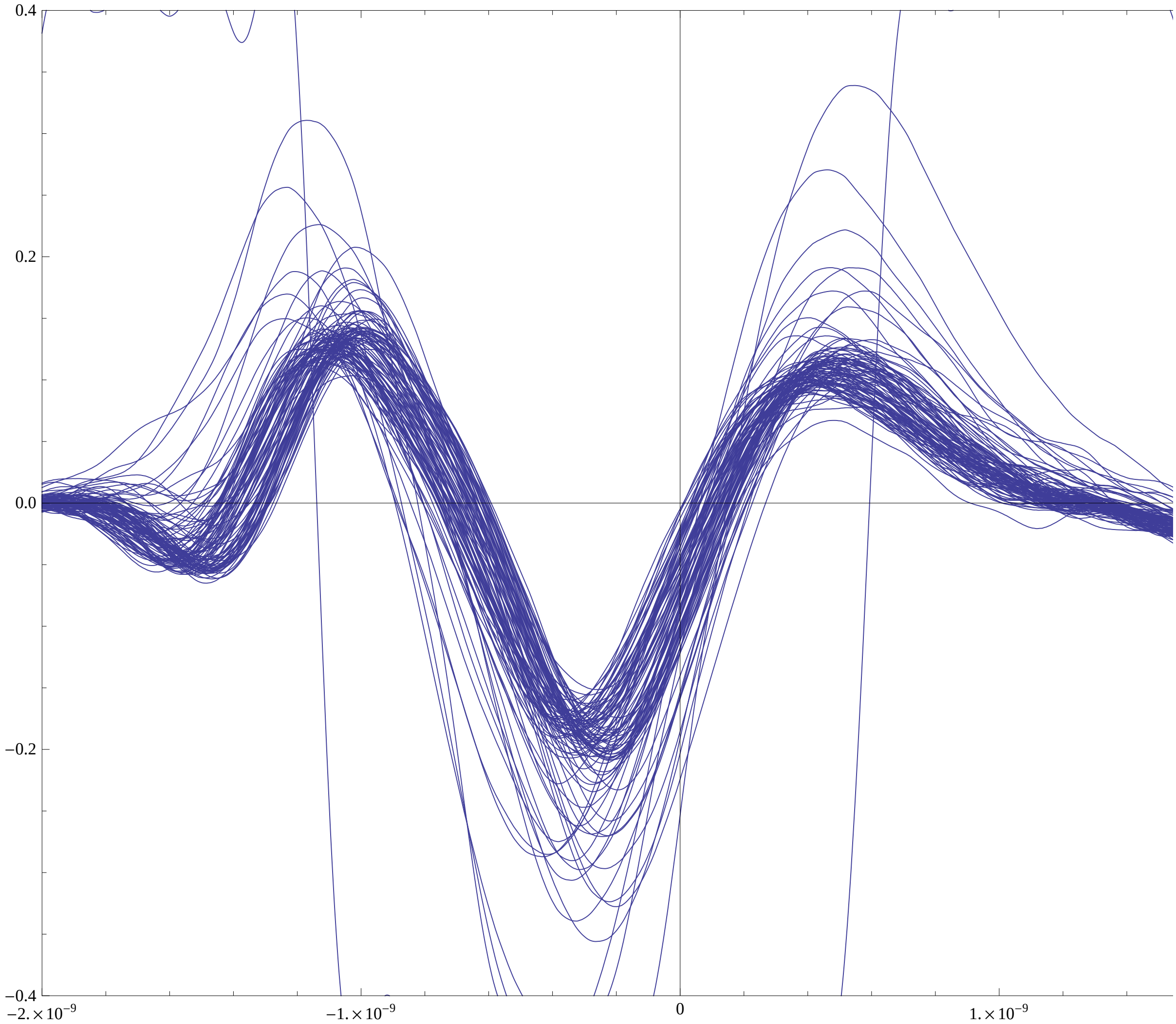
In[256]:= Plot[Table[Wb[[i]][x], {i, 1, 100}], {x, -2 × 10^-9, .3 × 10^-8},
PlotRange → {{-2 × 10^-9, .3 × 10^-8}, {-.17, .12}}, Frame → True]

Out[256]=



In[257]:=
```
Plot[Table[Wf[[i]][x], {i, 1, 100}], {x, -2 × 10^-9, .3 × 10^-8},
 PlotRange → {{-2 × 10^-9, .3 × 10^-8}, {-.4, .4}}, Frame → True]
```

Out[257]=

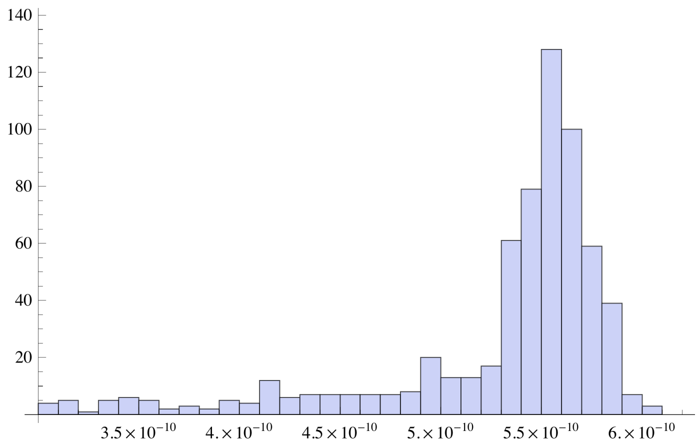



In[258]:=
```
oa = OpenAppend["InterpolatedWaveForms.dat", PageWidth → Infinity];
Do[
 Write[oa, {i, Wb[[i]], Wf[[i]]}];,
 {i, 1, iwave}]
Close[oa];
```

In[247]:=
```
Clear[Differ]
Differ = Table[(Zeroesf[[i]] - Zeroesb[[i]]), {i, 1, isaved}];
```

In[254]:=
```
Histogram [Differ , {0.3 × 10^-9, 0.62 × 10^-9, 0.01 × 10^-9}]
```

Out[254]=



Evaluate the slope of the wave function over 300 psec near the zero - crossing. This can be used to estimate the contribution to time jitter from scope input noise.

```
In[266]:= bslopes = {}; fslopes = {};
Do[
 AppendTo[bslopes, Wb[[i]][-.1 × 10^-9] - Wb[[i]][-.4 × 10^-9]];
 AppendTo[fslopes, Wf[[i]][.2 × 10^-9] - Wf[[i]][-.1 × 10^-9]];,
 {i, 1, isaved}]

In[268]:= meanbslope = Mean[bslopes] / (.3 × 10^-9)
meanfslope = Mean[fslopes] / (.3 × 10^-9)

Out[268]= 2.51672 × 10^8

Out[269]= 6.56288 × 10^8
```

We then derive the expected noise jitter on the time difference measured from the zero - cross method and get 13 picoseconds which is not far from what we are finding!

```
In[270]:= resexpb = avenoiseb / meanbslope
resexpf = avenoisef / meanfslope
dtwidthexp = Sqrt[resexpb^2 + resexpf^2]

Out[270]= 1.21408 × 10^-11

Out[271]= 4.8634 × 10^-12

Out[272]= 1.30787 × 10^-11

ListZb = ConstantArray[0, 1000];
Do[
 ListZb[[i]] := If[InexactNumberQ[Zeroesb[[i]]], Zeroesb[[i]], -10 × 10^-9];
 , {i, isaved}]
```