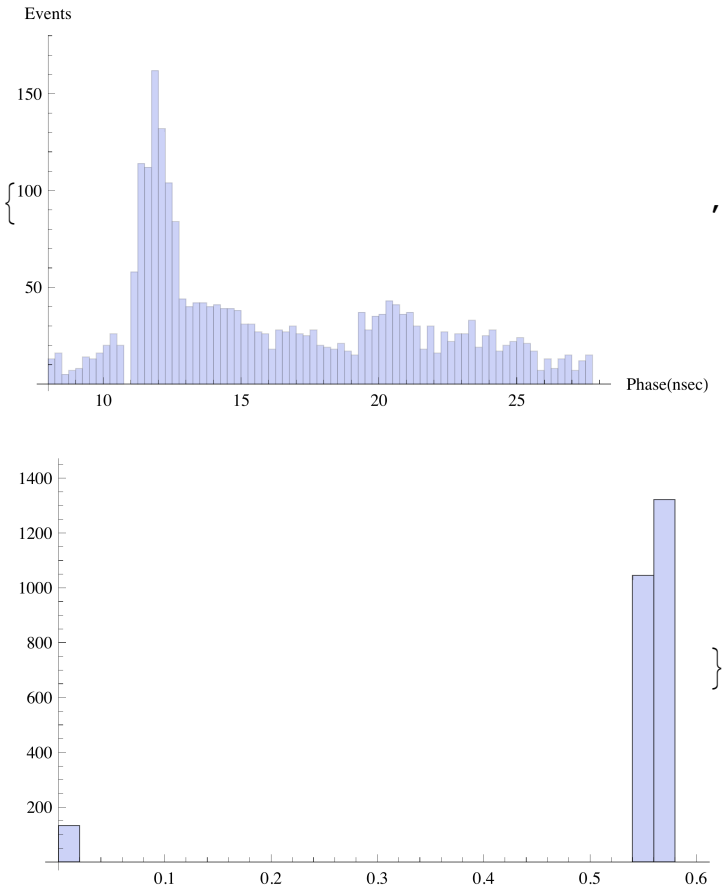
```
nsegment = 10; nloop = 25; nchan = 4; ichan = 4;
Clear[v]; v = ConstantArray[0, {2500, 4002}];
(*Do[cannel index-   but for moment only channel 4*)
Do[(*loop index*)
  Do[(*segemt index*)
    nev = iseg + 10 * (iloop - 1);
    trace = scopedata[[10 * (ichan - 1) + iseg + 40 * (iloop - 1)]];
    v[[nev]] = Drop[trace, 3];
    , {iseg, 1, 10, 1}];
  , {iloop, 1, 250, 1}];

model = a * a Sin[.318 (t - b)];
fitphase = Table[FindFit[Transpose[{time, (v[[iev]] + .4)}], model, {a, b}, t] /.
    Rule → List, {iev, 1, 2500}];

phase = Table[fitphase[[i, 2, 2]], {i, 2500}];
phaseshift = Mod[phase, 19.75, 8.];
ampl = Table[fitphase[[i, 1, 2]], {i, 2500}];
{Histogram[phaseshift, {8, 28, .25},
  AxesLabel → {"Phase(nsec)", "Events"}, ImageSize → Medium],
 Histogram[Abs[ampl], {0, .6, .02}, ImageSize → Medium]}
```



Estimate Distance scale from spread in Time of Flight

```
mu = ParticleData["Muon", "Mass"]; mpi = ParticleData["PiPlus", "Mass"];
me = ParticleData["Electron", "Mass"];
```

```
Solve[250 == mpi * beta / Sqrt[1 - beta * beta], beta] /. Rule → List;
vpi = %[[1, 1, 2]]
```

```
0.87314524
```

```
Solve[250 == me * beta / Sqrt[1 - beta * beta], beta] /. Rule → List;
ve = %[[1, 1, 2]]
```

```
0.99999791
```

```
Solve[250 == mu * beta / Sqrt[1 - beta * beta], beta] /. Rule → List;
vmu = %[[1, 1, 2]]
```

```
0.921113762
```

```
Solve[tofe == 2500 / 30 * (1 / ve - 1 / vpi), tofe]
```

```
{{tofe → -12.10690}}
```

```
Solve[tofmu == 2500 / 30 * (1 / vmu - 1 / vpi), tofmu]
```

```
{{tofmu → -4.97022}}
```

So get a time difference between pions and electrons of order 12 nsec assuming a distance from the production target of about 25 meters.

```
timefun[tbin_] := (tbin - 1) * dt
bin[t_] := (t) / dt + 1
```

```
timefun[1940]
```

```
96.95
```

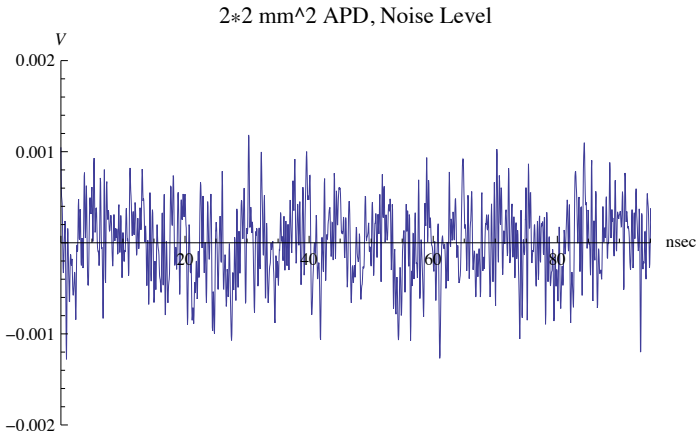
Now examine APD channels small APD (Scope input#3)

```
nsegment = 10; nloop = 25; nchan = 4; ichan = 3;
Clear[v]; v = ConstantArray[0, {2500, 4002}];
(*Do[cannel index-   but for moment only channel 3*)
Do[(*loop index*)
  Do[(*segemt index*)
    nev = iseg + 10 * (iloop - 1);
    trace = scopedata[[10 * (ichan - 1) + iseg + 40 * (iloop - 1)]];
    v[[nev]] = Drop[trace, 3];
    , {iseg, 1, 10, 1}];
  , {iloop, 1, 250, 1}];
```

```
offset3 = Table[Mean[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
v3 = Table[(v[[i]] - offset3[[i]]), {i, 1, 2500, 1}];
noise3 = Table[RootMeanSquare[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
Print[" Average offset=", 1000 * Mean[offset3],
 " mV and average noise= ", 1000 * Mean[noise3], " mV"]
```

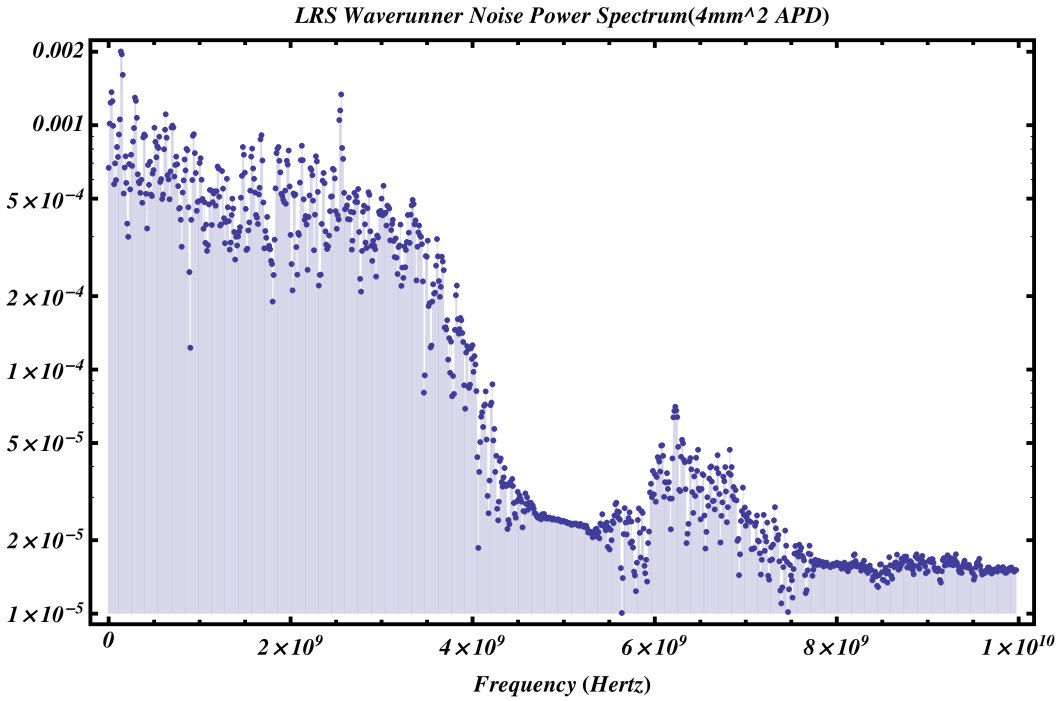
```
 Average offset=0.451717 mV and average noise= 0.772625 mV
```

```
ListPlot[Transpose[{time, v3[[10]]}], Joined → True, AxesLabel → {nsec, V},
 PlotRange → {{0, 95}, {-.002, .002}}, PlotLabel → "2*2 mm^2 APD, Noise Level"]
```

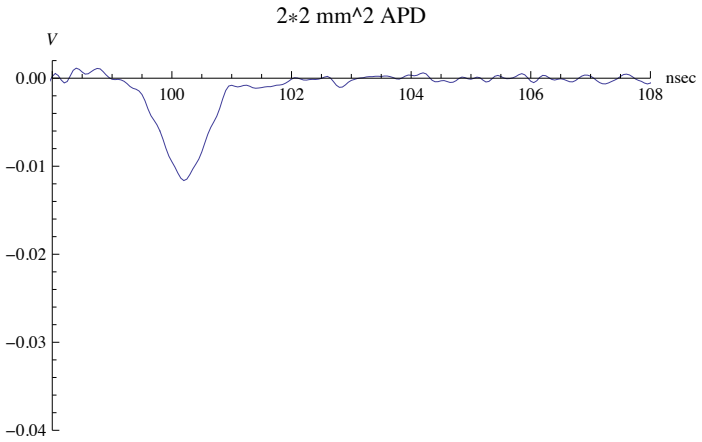


Noise analysis

```
precursor = Take[v3[[10]], 1940];
fftslice = Fourier[precursor];
samplFreq = 1 / dt * 10^9; fspi = (samplFreq / 2) / 970;
fftbins = fspi * Range[0, (970 - 1)];
splitfft = Take[Abs[fftslice], 970];
smoothfft = MovingAverage[splitfft, 3];
ListLogPlot[Transpose[{Take[fftbins, (970 - 2)], smoothfft}],
 Filling → Axis, Frame → True, FrameStyle → Thick, FrameLabel → {{, },
   {"Frequency (Hertz)", "LRS Waverunner Noise Power Spectrum(4mm^2 APD)"}},
 LabelStyle → {Medium, Italic, Bold}, ImageSize → Large]
```



```
ListPlot[Transpose[{time, v3[[10]]}], Joined → True, AxesLabel → {nsec, V},
 PlotRange → {{98, 108}, {-.04, .002}}, PlotLabel → "2*2 mm^2 APD"]
```



Now calculate expected peak pulse height assuming roughly triangular waveform 5 nsec wide at the base. i_peak*1/2*5nsec=Q=qe*ne*APDGain*Ampgain, V_peak=i_peak*50 Ohms

Gains and detector Capacitance dependent deficit from Voltage Amplifier Model

```
qe = -1.60217 * 10^-19;
ne = 6000; APDgain = 600;
defecit4mm = 0.5; defecit64 = 0.09;
NSolve[13 == 20 Log10[vgain], vgain] /. Rule → List;
Ampgain13 = %[[1, 1, 2]]
NSolve[20 == 20 Log10[vgain], vgain] /. Rule → List;
Ampgain20 = %[[1, 1, 2]]
```

```
4.46684
```

```
10.
```

```
NSolve[vpeak == 50 * 1 / (.5 * 5 * 10^-9) * qe * ne * APDgain * Ampgain13, vpeak] /.
 Rule → List
pulsepeakch3 = %[[1, 1, 2]] * defecit4mm
```

```
{{{vpeak, -0.0515277}}}
```

```
-0.0257639
```

```
NSolve[vpeak == 50 * 1 / (.5 * 5 * 10^-9) * qe * ne * APDgain * Ampgain13 * Ampgain20,
  vpeak] /. Rule → List
pulsepeakch1 = %[[1, 1, 2]] * defecit64
```

```
{{{vpeak, -0.515277}}}
```

```
-0.046375
```

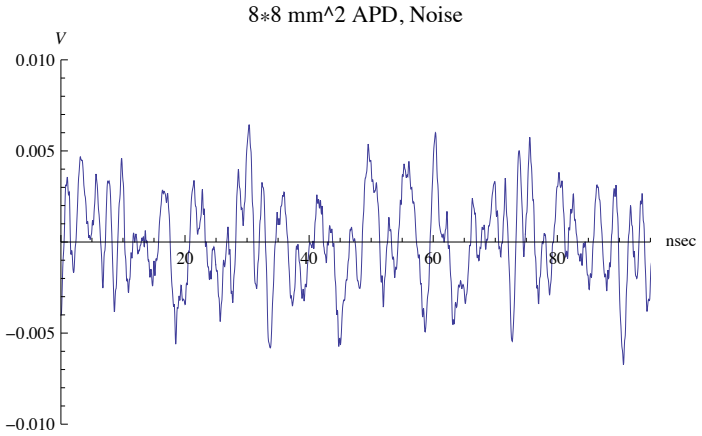
Large APD (Scope input channel#1). ~60 pF detector capacitance, 20 dB and 13dB in cascade

```
nsegment = 10; nloop = 25; nchan = 4; ichan = 1;
Clear[v]; v = ConstantArray[0, {2500, 4002}];
(*Do[cannel index-   but for moment only channel 1*)
Do[(*loop index*)
  Do[(*segemt index*)
    nev = iseg + 10 * (iloop - 1);
    trace = scopedata[[10 * (ichan - 1) + iseg + 40 * (iloop - 1)]];
    v[[nev]] = Drop[trace, 3];
    , {iseg, 1, 10, 1}];
  , {iloop, 1, 250, 1}];
```

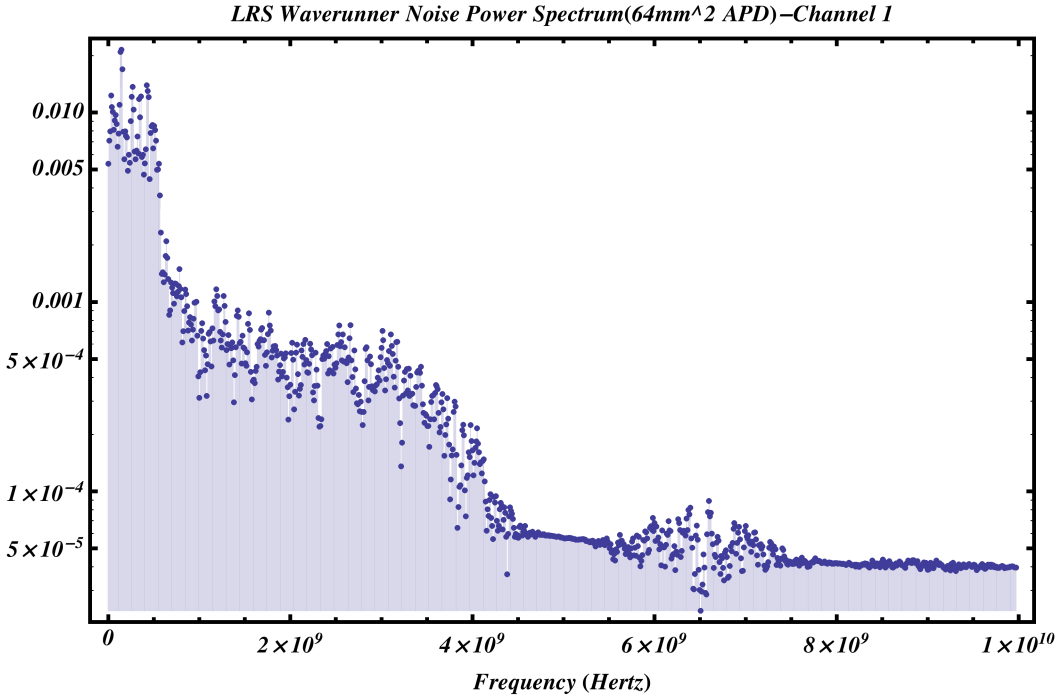
```
offset1 = Table[Mean[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
v1 = Table[(v[[i]] - offset1[[i]]), {i, 1, 2500, 1}];
noise1 = Table[RootMeanSquare[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
Print[" Average offset=", 1000 * Mean[offset1],
 " mV and average noise= ", 1000 * Mean[noise1], " mV"]
```

```
 Average offset=0.862518 mV and average noise= 3.00544 mV
```

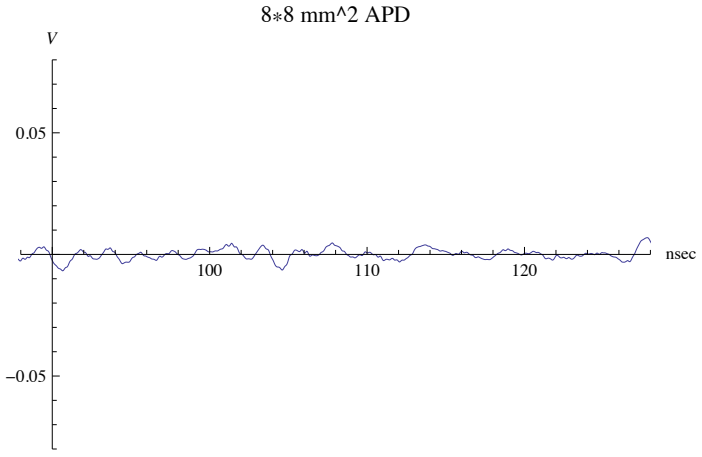
```
ListPlot[Transpose[{time, v1[[7]]}], Joined → True, AxesLabel → {nsec, V},
 PlotRange → {{0, 95}, {-.01, .01}}, PlotLabel → "8*8 mm^2 APD, Noise"]
```



```
precursor = Take[v1[[10]], 1940];
fftslice = Fourier[precursor];
splitfft = Take[Abs[fftslice], 970];
smoothfft = MovingAverage[splitfft, 3];
ListLogPlot[Transpose[{Take[fftbins, (970 - 2)], smoothfft}], Filling → Axis,
 Frame → True, FrameStyle → Thick, FrameLabel → {{, }, {"Frequency (Hertz)",
    "LRS Waverunner Noise Power Spectrum(64mm^2 APD)-Channel 1"}},
 LabelStyle → {Medium, Italic, Bold}, ImageSize → Large]
```




```
ListPlot[Transpose[{time, v1[[7]]}], Joined → True, AxesLabel → {nsec, V},
 PlotRange → {{88, 128}, {-.08, .08}}, PlotLabel → "8*8 mm^2 APD"]
```

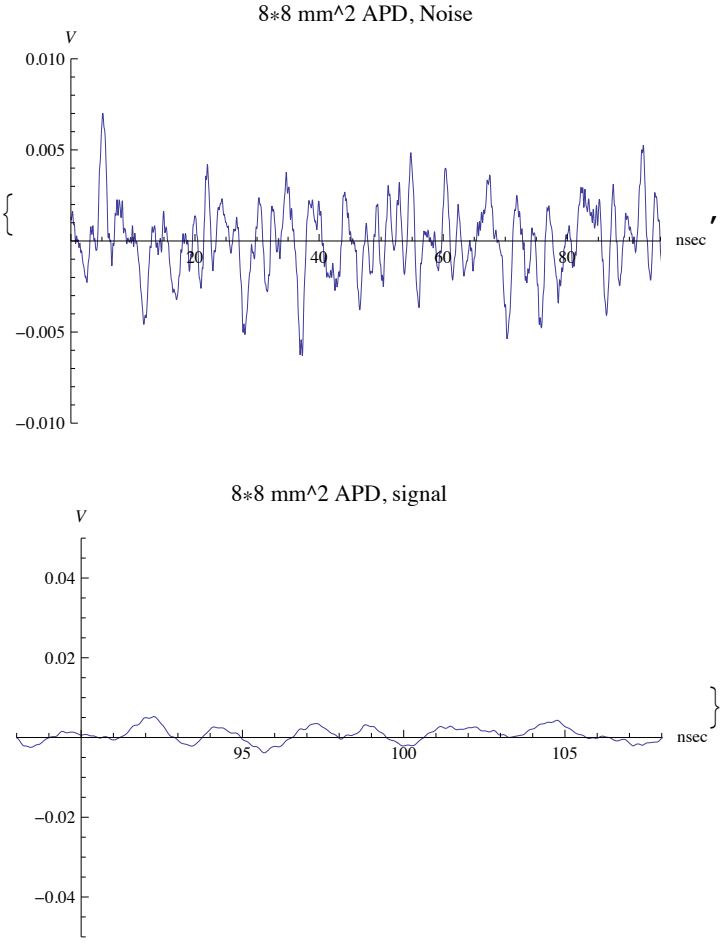


Large APD on scope input #2

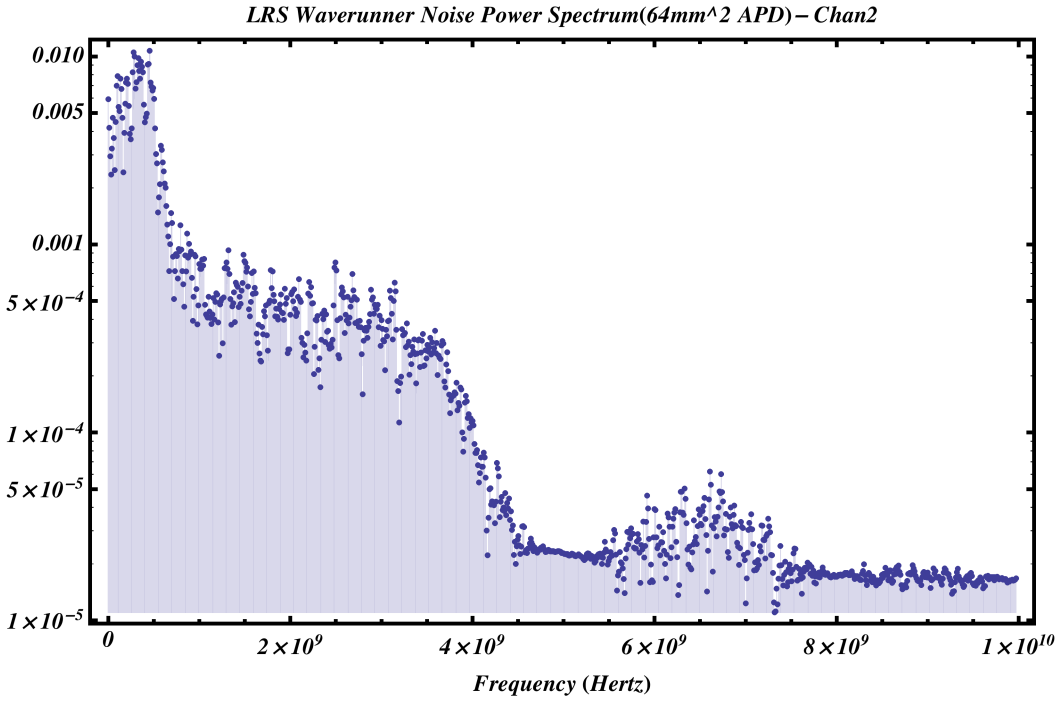
```
nsegment = 10; nloop = 25; nchan = 4; ichan = 2;
Clear[v]; v = ConstantArray[0, {2500, 4002}];
(*Do[cannel index-   but for moment only channel 4*)
Do[(*loop index*)
  Do[(*segemt index*)
    nev = iseg + 10 * (iloop - 1);
    trace = scopedata[[10 * (ichan - 1) + iseg + 40 * (iloop - 1)]];
    v[[nev]] = Drop[trace, 3];
    , {iseg, 1, 10, 1}];
  , {iloop, 1, 250, 1}];
offset2 = Table[Mean[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
v2 = Table[(v[[i]] - offset2[[i]]), {i, 1, 2500, 1}];
noise2 = Table[RootMeanSquare[Take[v[[i]], 1980]], {i, 1, 2500, 1}];
Print[" Average offset=", 1000 * Mean[offset2],
 " mV and average noise= ", 1000 * Mean[noise2], " mV"]
```

```
 Average offset=0.879456 mV and average noise= 2.50954 mV
```

```
{ListPlot[Transpose[{time, v2[[7]]}], Joined → True, AxesLabel → {nsec, V},
  PlotRange → {{0, 95}, {-.01, .01}}, PlotLabel → "8*8 mm^2 APD, Noise",
  ImageSize → Medium], ListPlot[Transpose[{time, v2[[7]]}], Joined → True,
  AxesLabel → {nsec, V}, PlotRange → {{88, 108}, {-.05, .05}},
  PlotLabel → "8*8 mm^2 APD, signal", ImageSize → Medium]}
```



```
precursor = Take[v2[[10]], 1940]; fftslice = Fourier[precursor];
splitfft = Take[Abs[fftslice], 970]; smoothfft = MovingAverage[splitfft, 3];
ListLogPlot[Transpose[{Take[fftbins, (970 - 2)], smoothfft}], Filling → Axis,
 Frame → True, FrameStyle → Thick, FrameLabel → {{, }, {"Frequency (Hertz)",
    "LRS Waverunner Noise Power Spectrum(64mm^2 APD)- Chan2"}},
 LabelStyle → {Medium, Italic, Bold}, ImageSize → Large]
```



Now Extract useful range of data for further analysis. We will save phase information for clock and time info for the time range 90 to 110 nsec.

```
bin[90]
bin[110]
```

```
1801.
```

```
2201.
```

```
Dimensions[v1red]
```

```
{}
```

```
v1red = Table[Take[Drop[v1[[i]], 1800], 400], {i, 1, 2500, 1}];
v2red = Table[Take[Drop[v2[[i]], 1800], 400], {i, 1, 2500, 1}];
v3red = Table[Take[Drop[v3[[i]], 1800], 400], {i, 1, 2500, 1}];
datacompress = Table[
   {phase[[i]], ampl[[i]], v1red[[i]], v2red[[i]], v3red[[i]]}, {i, 1, 2500, 1}];
```

```
Dimensions[datacompress]
```

```
{2500, 5}
```

```
Dimensions[datacompress[[1, 3]]]
```

```
{400}
```

```
Export[ outfile, datacompress, "csv"]
```

```
LeCroy-compress-2013-06-02-049.csv
```