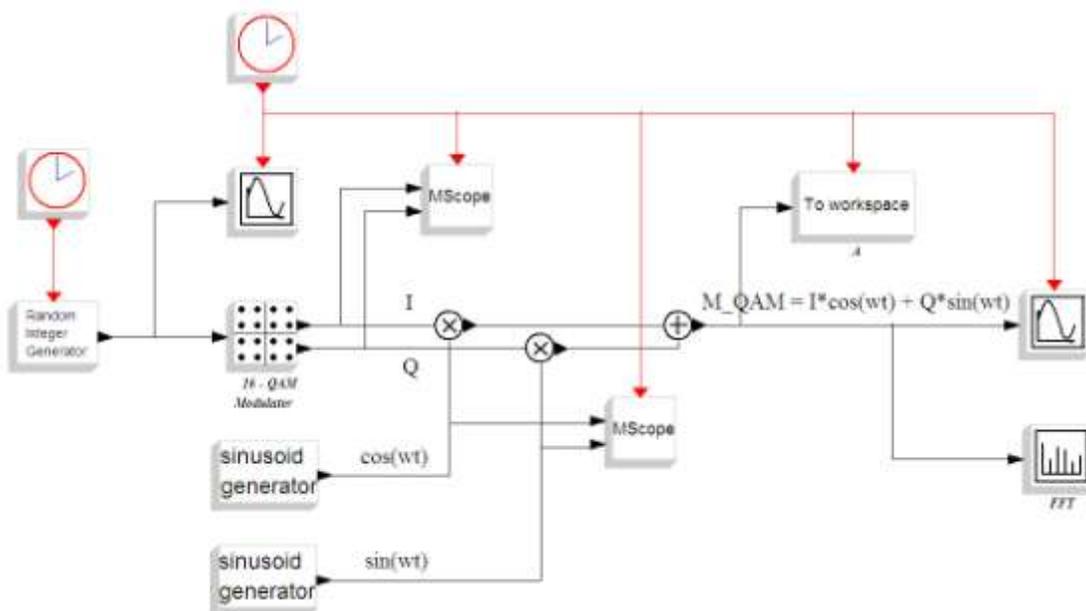


LEARN TELECOMMUNICATIONS BY SIMULATION

Jeremy Clark VE3PKC





ISBN 978-0-9880490-0-0

© Clark Telecommunications/Jeremy Clark June 2012

All rights reserved. No part of this work shall be reproduced, stored in a retrieval system or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the written permission of the author. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the author assumes no responsibility for errors, omissions, inaccuracies or any inconsistency herein. Nor is any liability assumed for damages resulting from the use of the information contained herein.

This work is sold as is, without any warranty of any kind, either express or implied, respecting the contents of this book, including but not limited to implied warranties for the book's quality, performance, merchantability, or fitness for any particular purpose.

Scicoslab & Scicos are trademarks of ©INRIA-ENPC in France. Modnum is a trademark of Alan Layec at INRIA France.

Clark Telecommunications

jclark@clarktelecommunications.com

<https://www.clarktelecommunications.com/simulation.htm>

Contents

- 1 Introduction
- 2 Open Source Tools and Instrumentation. ScicosLab, Modnum Toolbox
- 3 Overview Telecommunications Signals. Wired, Wireless, Baseband and Modulated
- 4 Baseband Analog and Digital Signals
 - 4.1 Baseband Analog - Single Tone
 - 4.2 Baseband Analog - Multiple Tones
 - 4.3 Baseband Analog - WAV
 - 4.4 Baseband Analog - Signal Captures
 - 4.5 Baseband Digital - NRZ Random Data
 - 4.6 Baseband Digital - Manchester Random Data
 - 4.7 Baseband Digital - Signal Captures
 - 4.8 Baseband Analog and Digital Exercises
- 5 The Superheterodyne Process for Analog and Digital Signals
 - 5.1 Superhet Analog - Single Tone
 - 5.2 Superhet Analog - Multiple Tones
 - 5.3 Superhet Analog - WAV
 - 5.4 Superhet Analog - Signal Captures
 - 5.5 Superhet Digital - NRZ Random Data
 - 5.6 Superhet Analog and Digital - Exercises
- 6 AM Amplitude Modulation for Analog and Digital Signals
 - 6.1 AM Analog
 - 6.2 AM Analog - Signal Captures
 - 6.3 AM Digital - ASK Amplitude Shift Keying
 - 6.4 AM Digital - Signal Captures
 - 6.5 AM Analog and Digital - Exercises
- 7 FM Frequency Modulation and PM Phase Modulation for Analog and Digital Signals
 - 7.1 FM Analog – Single Tone
 - 7.2 FM Analog – WAV
 - 7.3 FM Analog Signal Captures
 - 7.4 FM Digital – FSK Frequency Shift Keying

Contents

- 7.5 FM Digital FSK Signal Captures
- 7.6 PM Analog – Single Tone
- 7.7 PM Digital PSK Phase Shift Keying
- 7.8 PM Digital PSK Phase Shift Keying Signal Captures
- 7.9 FM and PM Analog and Digital Exercises

- 8 QAM Quadrature Amplitude Modulation and OFDM Orthogonal Frequency Division Multiplexing
 - 8.1 QAM
 - 8.2 QAM Signal Captures
 - 8.3 OFDM
 - 8.4 OFDM Signal Captures
 - 8.5 QAM and OFDM Exercises

- 9 Channel Modeling for QAM and CDMA Code Division Multiple Access
 - 9.1 QAM Channel
 - 9.2 CDMA Code Division Multiple Access Channel
 - 9.3 QAM and CDMA Channel Model Exercises

- Appendix A Working with ScicosLab & Modnum Toolbox

- Appendix B Glossary

- References

- Index

Introduction

I have always found that learning something is helped by using more than one of your senses if at all possible. Linking several things together can be a powerful way to anchor something in your mind. One of the benefits of being an amateur radio operator is that when you study to get your license you invariably build or work with equipment. So diagrams and equations are not just abstract things, they relate to real things. Computer simulation adds a further dimension to learning. We are able to actually build a system in software and see the same waveforms that we would see on an oscilloscope or spectrum analyzer without having to build or buy the equipment.

In this text I will overview the most common Telecommunications signal types in use nowadays and build models to simulate their operation on a basic level. Wherever possible, I will compress the spectrum into the audio range so that we can not only see but hear the signal as well. To do this I will use freely available open source tools such as ScicosLab and the Modnum Toolbox. I will also include real world signal captures so that we can compare real hardware signals to our simulations.

When we analyze all the commonly used Telecommunications Signals in use today, we will discover that most of the newer ones use common digital modulation techniques such as PSK Phase Shift Keying, QAM Quadrature Amplitude Modulation and OFDM Orthogonal Frequency Division Multiplexing. A separate section is present for each one of these techniques as well as AM Amplitude Modulation, FM Frequency Modulation and PM Phase Modulation.

During my career as a Telecommunication engineer and Professor of Engineering Technology, I have found that keeping things as simple as possible pays off. I have been able to work on design projects and implement them in the field as well. I worked in many countries and learned valuable lessons. I always like doing measurements different ways on different equipment/systems and getting approximately the same answer. I like equipment that is small, rugged, portable and battery powered (many telecom sites only have DC power). Things never quite go as planned, so being flexible and organized is essential.

Building Telecom models in software also provides a bridge to DSP Digital Signal Processing. DSP is used in all branches of Telecommunications. Once a signal is described mathematically in a discrete fashion, then DSP algorithms can be applied. Young people these days love using computers, so tying math and computers together is a win-win process.

This text has been designed as an e-book. In order to see a diagram, just mouse over the figure and the hyperlink will open up the appropriate drawing. Links have been included to descriptive videos and code files. The video [intro.mp4](#) explains how to use the text. Appendix A and video [scicos.mp4](#) describes the installation and use of ScicosLab and the Modnum Toolbox. Read this section first!

Open Source Tools & Instrumentation. Scilab, ScicosLab.

In order to build the various models and scripts used in this text, I will use ScicosLab or Scicos with the Modnum Toolbox. This powerful software is Open Source and freely available for download:

ScicosLab Latest Version 4.4.1 as of Apr 2012:

www.scicoslab.org

Modnum Toolbox Version 4.2.2 as of Apr 2012:

www.scicos.org/ScicosModNum/modnum_web/web/eng/eng.htm

Appendix A and video [scicos.mp4](#) contain detailed instructions how to download, install and use this software. Most examples in the text are built two ways. The first way is using a script program in ScicosLab and the second way is using the graphical blockset from the Modnum Toolbox within Scicos.

For real world signal captures, I will use either the RFSPACE SDR-IQ (0 - 30MHz) Software Defined Receiver or the Signal Hound SA44B Software Defined Spectrum Analyzer (0 - 4.4GHz).

RFSPACE SDR-IQ: www.rfspace.com/RFSPACE/SDR-IQ.html

Signal Hound: www.signalhound.com/SA44B.htm

Overview of Telecommunications Signals

Let us consider some of the most popular Telecommunications signals in use nowadays. **Fig 3.1** Lists the various systems. There are various ways to classify these signals:

- Personal, Commercial, Industrial, or Military
- Broadcast
- Baseband or Modulated
- Wired (Cable, Fiber Optics, Power Line Carrier) or Wireless
- Terrestrial or Satellite

Popular Telecommunications Signals	Description
Cellular Telephony	Personal, Wireless, PSK or CDMA
Internet	Personal, Wired, OFDM
WiFi	Personal, Wireless, OFDM
HDTV	Broadcast, Wireless, ASK
FM Radio	Broadcast, Wireless, FM
Satellite Radio	Broadcast, Wireless, PSK
AM Radio	Broadcast, Wireless, AM
Telephony Landline	Personal, Wired, Baseband Analog

Fig 3.1 Popular Telecommunications Signals

For example consider Cellular Telephony. Cellular Telephony uses Wireless to communicate between the subscriber cell phone and the cellular base station. Various modulation techniques are used. For instance GSM and CDMA are two widely used techniques. GSM uses a type of digital phase modulation or PSK. Another popular signal is analog FM radio. FM radio has been around since the 1950's. It allows for stereo transmission with excellent sound quality and very simple transmission equipment.

When we consider all the various signals, we see that they all contain some combination of the basic building blocks that we will cover in the next chapters:

- Baseband analog and digital
- AM amplitude modulation analog and digital
- FM Frequency modulation analog and digital
- PM Phase modulation analog and digital
- QAM Quadrature Amplitude Modulation, combination of digital AM & PM
- OFDM Orthogonal Frequency Division Multiplexing, many PSK/QAM signals
- CDMA Code Division Multiple Access

Digital Telecommunications systems are often analyzed according to the OSI or TCP/IP models, **Fig 3.2**. In this text, we will be considering the so called Physical 1 Layer. Upper layers are concerned more with software concepts. The Physical 1 Layer is concerned with the actual signal voltage that is placed on the cable or wireless link.

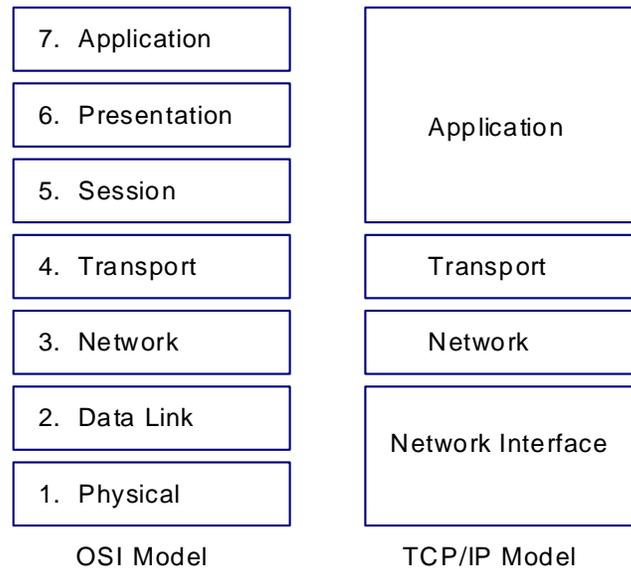


Fig 3.2 OSI & TCP/IP Models

The video [over_sigs.mp4](#) discusses these concepts further.

Baseband Analog and Digital Signals

4.1 Baseband Analog - Single Tone

Baseband is a relative term and refers to frequencies that are close to DC and have not been modulated or shifted by a carrier. Human speech as monitored from a microphone contains frequencies from about 20 Hz to 7KHz. Human hearing extends to about 20KHz max. An analog video signal contains frequencies from approx. DC to several MHz. A baseband data signal contains frequencies from approx. DC to the data rate and beyond. The video [bb.mp4](#) introduces these topics.

The simplest of all baseband signals is a single tone. **Fig 4.1** [bb_tone.sce](#) is a ScicosLab program to generate a single tone at 1000Hz and plot the waveform in time and plot its spectrum in the frequency domain. **Fig 4.2** [bb_tone.cos](#) does exactly the same thing but uses the Scicos graphical block structure. Both files are explained in the demo file [bb_analog.mp4](#). **Fig 4.3** shows the time domain and spectral display of the single tone.

Note that a single tone has a single line spectrum at the tone frequency. Conversely, a line in the spectrum of a signal indicates that the signal contains a tone component at that frequency.

```

1 // This file generates a single element tone
2 // fa = tone amplitude in volts peak
3 // fs = sampling frequency
4 // Ts = sampling frequency inverse
5 // Date April 200-2012
6
7 // Enter the tone parameters
8 fa=1
9 fs=10000
10
11 // Calculate the tone waveform
12 n=length(fs)
13 t=0:n-1;
14 Ts=1/fs;
15 t=t*Ts;
16 signal=fa*cos(2*pi*t);
17 plot(t,signal);
18
19 // Save the tone waveform
20 // And/or open graphic window to
21 save('bb_tone.sce');
22
23 // Calculate the spectrum with FFT
24 // And/or open graphic window to
25 plot(f,fft);
26
27 // Calculate the spectrum with FFT
28 // And/or open graphic window to
29 plot(f,abs(fft));
30
31 // Save the spectrum
32 save('bb_tone.cos');
33
34 // Plot the spectrum
35 plot(f,abs(fft));
36
37 // Save the spectrum
38 save('bb_tone.cos');
39
40 // Plot the spectrum
41 plot(f,abs(fft));
42
43 // Save the spectrum
44 save('bb_tone.cos');
45
46 // Plot the spectrum
47 plot(f,abs(fft));
48
49 // Save the spectrum
50 save('bb_tone.cos');
51
52 // Plot the spectrum
53 plot(f,abs(fft));
54
55 // Save the spectrum
56 save('bb_tone.cos');
57
58 // Plot the spectrum
59 plot(f,abs(fft));
60
61 // Save the spectrum
62 save('bb_tone.cos');
63
64 // Plot the spectrum
65 plot(f,abs(fft));
66
67 // Save the spectrum
68 save('bb_tone.cos');
69
70 // Plot the spectrum
71 plot(f,abs(fft));
72
73 // Save the spectrum
74 save('bb_tone.cos');
75
76 // Plot the spectrum
77 plot(f,abs(fft));
78
79 // Save the spectrum
80 save('bb_tone.cos');
81
82 // Plot the spectrum
83 plot(f,abs(fft));
84
85 // Save the spectrum
86 save('bb_tone.cos');
87
88 // Plot the spectrum
89 plot(f,abs(fft));
90
91 // Save the spectrum
92 save('bb_tone.cos');
93
94 // Plot the spectrum
95 plot(f,abs(fft));
96
97 // Save the spectrum
98 save('bb_tone.cos');
99
100 // Plot the spectrum
101 plot(f,abs(fft));
102
103 // Save the spectrum
104 save('bb_tone.cos');
105
106 // Plot the spectrum
107 plot(f,abs(fft));
108
109 // Save the spectrum
110 save('bb_tone.cos');
111
112 // Plot the spectrum
113 plot(f,abs(fft));
114
115 // Save the spectrum
116 save('bb_tone.cos');
117
118 // Plot the spectrum
119 plot(f,abs(fft));
120
121 // Save the spectrum
122 save('bb_tone.cos');
123
124 // Plot the spectrum
125 plot(f,abs(fft));
126
127 // Save the spectrum
128 save('bb_tone.cos');
129
130 // Plot the spectrum
131 plot(f,abs(fft));
132
133 // Save the spectrum
134 save('bb_tone.cos');
135
136 // Plot the spectrum
137 plot(f,abs(fft));
138
139 // Save the spectrum
140 save('bb_tone.cos');
141
142 // Plot the spectrum
143 plot(f,abs(fft));
144
145 // Save the spectrum
146 save('bb_tone.cos');
147
148 // Plot the spectrum
149 plot(f,abs(fft));
150
151 // Save the spectrum
152 save('bb_tone.cos');
153
154 // Plot the spectrum
155 plot(f,abs(fft));
156
157 // Save the spectrum
158 save('bb_tone.cos');
159
160 // Plot the spectrum
161 plot(f,abs(fft));
162
163 // Save the spectrum
164 save('bb_tone.cos');
165
166 // Plot the spectrum
167 plot(f,abs(fft));
168
169 // Save the spectrum
170 save('bb_tone.cos');
171
172 // Plot the spectrum
173 plot(f,abs(fft));
174
175 // Save the spectrum
176 save('bb_tone.cos');
177
178 // Plot the spectrum
179 plot(f,abs(fft));
180
181 // Save the spectrum
182 save('bb_tone.cos');
183
184 // Plot the spectrum
185 plot(f,abs(fft));
186
187 // Save the spectrum
188 save('bb_tone.cos');
189
190 // Plot the spectrum
191 plot(f,abs(fft));
192
193 // Save the spectrum
194 save('bb_tone.cos');
195
196 // Plot the spectrum
197 plot(f,abs(fft));
198
199 // Save the spectrum
200 save('bb_tone.cos');
201
202 // Plot the spectrum
203 plot(f,abs(fft));
204
205 // Save the spectrum
206 save('bb_tone.cos');
207
208 // Plot the spectrum
209 plot(f,abs(fft));
210
211 // Save the spectrum
212 save('bb_tone.cos');
213
214 // Plot the spectrum
215 plot(f,abs(fft));
216
217 // Save the spectrum
218 save('bb_tone.cos');
219
220 // Plot the spectrum
221 plot(f,abs(fft));
222
223 // Save the spectrum
224 save('bb_tone.cos');
225
226 // Plot the spectrum
227 plot(f,abs(fft));
228
229 // Save the spectrum
230 save('bb_tone.cos');
231
232 // Plot the spectrum
233 plot(f,abs(fft));
234
235 // Save the spectrum
236 save('bb_tone.cos');
237
238 // Plot the spectrum
239 plot(f,abs(fft));
240
241 // Save the spectrum
242 save('bb_tone.cos');
243
244 // Plot the spectrum
245 plot(f,abs(fft));
246
247 // Save the spectrum
248 save('bb_tone.cos');
249
250 // Plot the spectrum
251 plot(f,abs(fft));
252
253 // Save the spectrum
254 save('bb_tone.cos');
255
256 // Plot the spectrum
257 plot(f,abs(fft));
258
259 // Save the spectrum
260 save('bb_tone.cos');
261
262 // Plot the spectrum
263 plot(f,abs(fft));
264
265 // Save the spectrum
266 save('bb_tone.cos');
267
268 // Plot the spectrum
269 plot(f,abs(fft));
270
271 // Save the spectrum
272 save('bb_tone.cos');
273
274 // Plot the spectrum
275 plot(f,abs(fft));
276
277 // Save the spectrum
278 save('bb_tone.cos');
279
280 // Plot the spectrum
281 plot(f,abs(fft));
282
283 // Save the spectrum
284 save('bb_tone.cos');
285
286 // Plot the spectrum
287 plot(f,abs(fft));
288
289 // Save the spectrum
290 save('bb_tone.cos');
291
292 // Plot the spectrum
293 plot(f,abs(fft));
294
295 // Save the spectrum
296 save('bb_tone.cos');
297
298 // Plot the spectrum
299 plot(f,abs(fft));
300
301 // Save the spectrum
302 save('bb_tone.cos');
303
304 // Plot the spectrum
305 plot(f,abs(fft));
306
307 // Save the spectrum
308 save('bb_tone.cos');
309
310 // Plot the spectrum
311 plot(f,abs(fft));
312
313 // Save the spectrum
314 save('bb_tone.cos');
315
316 // Plot the spectrum
317 plot(f,abs(fft));
318
319 // Save the spectrum
320 save('bb_tone.cos');
321
322 // Plot the spectrum
323 plot(f,abs(fft));
324
325 // Save the spectrum
326 save('bb_tone.cos');
327
328 // Plot the spectrum
329 plot(f,abs(fft));
330
331 // Save the spectrum
332 save('bb_tone.cos');
333
334 // Plot the spectrum
335 plot(f,abs(fft));
336
337 // Save the spectrum
338 save('bb_tone.cos');
339
340 // Plot the spectrum
341 plot(f,abs(fft));
342
343 // Save the spectrum
344 save('bb_tone.cos');
345
346 // Plot the spectrum
347 plot(f,abs(fft));
348
349 // Save the spectrum
350 save('bb_tone.cos');
351
352 // Plot the spectrum
353 plot(f,abs(fft));
354
355 // Save the spectrum
356 save('bb_tone.cos');
357
358 // Plot the spectrum
359 plot(f,abs(fft));
360
361 // Save the spectrum
362 save('bb_tone.cos');
363
364 // Plot the spectrum
365 plot(f,abs(fft));
366
367 // Save the spectrum
368 save('bb_tone.cos');
369
370 // Plot the spectrum
371 plot(f,abs(fft));
372
373 // Save the spectrum
374 save('bb_tone.cos');
375
376 // Plot the spectrum
377 plot(f,abs(fft));
378
379 // Save the spectrum
380 save('bb_tone.cos');
381
382 // Plot the spectrum
383 plot(f,abs(fft));
384
385 // Save the spectrum
386 save('bb_tone.cos');
387
388 // Plot the spectrum
389 plot(f,abs(fft));
390
391 // Save the spectrum
392 save('bb_tone.cos');
393
394 // Plot the spectrum
395 plot(f,abs(fft));
396
397 // Save the spectrum
398 save('bb_tone.cos');
399
400 // Plot the spectrum
401 plot(f,abs(fft));
402
403 // Save the spectrum
404 save('bb_tone.cos');
405
406 // Plot the spectrum
407 plot(f,abs(fft));
408
409 // Save the spectrum
410 save('bb_tone.cos');
411
412 // Plot the spectrum
413 plot(f,abs(fft));
414
415 // Save the spectrum
416 save('bb_tone.cos');
417
418 // Plot the spectrum
419 plot(f,abs(fft));
420
421 // Save the spectrum
422 save('bb_tone.cos');
423
424 // Plot the spectrum
425 plot(f,abs(fft));
426
427 // Save the spectrum
428 save('bb_tone.cos');
429
430 // Plot the spectrum
431 plot(f,abs(fft));
432
433 // Save the spectrum
434 save('bb_tone.cos');
435
436 // Plot the spectrum
437 plot(f,abs(fft));
438
439 // Save the spectrum
440 save('bb_tone.cos');
441
442 // Plot the spectrum
443 plot(f,abs(fft));
444
445 // Save the spectrum
446 save('bb_tone.cos');
447
448 // Plot the spectrum
449 plot(f,abs(fft));
450
451 // Save the spectrum
452 save('bb_tone.cos');
453
454 // Plot the spectrum
455 plot(f,abs(fft));
456
457 // Save the spectrum
458 save('bb_tone.cos');
459
460 // Plot the spectrum
461 plot(f,abs(fft));
462
463 // Save the spectrum
464 save('bb_tone.cos');
465
466 // Plot the spectrum
467 plot(f,abs(fft));
468
469 // Save the spectrum
470 save('bb_tone.cos');
471
472 // Plot the spectrum
473 plot(f,abs(fft));
474
475 // Save the spectrum
476 save('bb_tone.cos');
477
478 // Plot the spectrum
479 plot(f,abs(fft));
480
481 // Save the spectrum
482 save('bb_tone.cos');
483
484 // Plot the spectrum
485 plot(f,abs(fft));
486
487 // Save the spectrum
488 save('bb_tone.cos');
489
490 // Plot the spectrum
491 plot(f,abs(fft));
492
493 // Save the spectrum
494 save('bb_tone.cos');
495
496 // Plot the spectrum
497 plot(f,abs(fft));
498
499 // Save the spectrum
500 save('bb_tone.cos');
501
502 // Plot the spectrum
503 plot(f,abs(fft));
504
505 // Save the spectrum
506 save('bb_tone.cos');
507
508 // Plot the spectrum
509 plot(f,abs(fft));
510
511 // Save the spectrum
512 save('bb_tone.cos');
513
514 // Plot the spectrum
515 plot(f,abs(fft));
516
517 // Save the spectrum
518 save('bb_tone.cos');
519
520 // Plot the spectrum
521 plot(f,abs(fft));
522
523 // Save the spectrum
524 save('bb_tone.cos');
525
526 // Plot the spectrum
527 plot(f,abs(fft));
528
529 // Save the spectrum
530 save('bb_tone.cos');
531
532 // Plot the spectrum
533 plot(f,abs(fft));
534
535 // Save the spectrum
536 save('bb_tone.cos');
537
538 // Plot the spectrum
539 plot(f,abs(fft));
540
541 // Save the spectrum
542 save('bb_tone.cos');
543
544 // Plot the spectrum
545 plot(f,abs(fft));
546
547 // Save the spectrum
548 save('bb_tone.cos');
549
550 // Plot the spectrum
551 plot(f,abs(fft));
552
553 // Save the spectrum
554 save('bb_tone.cos');
555
556 // Plot the spectrum
557 plot(f,abs(fft));
558
559 // Save the spectrum
560 save('bb_tone.cos');
561
562 // Plot the spectrum
563 plot(f,abs(fft));
564
565 // Save the spectrum
566 save('bb_tone.cos');
567
568 // Plot the spectrum
569 plot(f,abs(fft));
570
571 // Save the spectrum
572 save('bb_tone.cos');
573
574 // Plot the spectrum
575 plot(f,abs(fft));
576
577 // Save the spectrum
578 save('bb_tone.cos');
579
580 // Plot the spectrum
581 plot(f,abs(fft));
582
583 // Save the spectrum
584 save('bb_tone.cos');
585
586 // Plot the spectrum
587 plot(f,abs(fft));
588
589 // Save the spectrum
590 save('bb_tone.cos');
591
592 // Plot the spectrum
593 plot(f,abs(fft));
594
595 // Save the spectrum
596 save('bb_tone.cos');
597
598 // Plot the spectrum
599 plot(f,abs(fft));
600
601 // Save the spectrum
602 save('bb_tone.cos');
603
604 // Plot the spectrum
605 plot(f,abs(fft));
606
607 // Save the spectrum
608 save('bb_tone.cos');
609
610 // Plot the spectrum
611 plot(f,abs(fft));
612
613 // Save the spectrum
614 save('bb_tone.cos');
615
616 // Plot the spectrum
617 plot(f,abs(fft));
618
619 // Save the spectrum
620 save('bb_tone.cos');
621
622 // Plot the spectrum
623 plot(f,abs(fft));
624
625 // Save the spectrum
626 save('bb_tone.cos');
627
628 // Plot the spectrum
629 plot(f,abs(fft));
630
631 // Save the spectrum
632 save('bb_tone.cos');
633
634 // Plot the spectrum
635 plot(f,abs(fft));
636
637 // Save the spectrum
638 save('bb_tone.cos');
639
640 // Plot the spectrum
641 plot(f,abs(fft));
642
643 // Save the spectrum
644 save('bb_tone.cos');
645
646 // Plot the spectrum
647 plot(f,abs(fft));
648
649 // Save the spectrum
650 save('bb_tone.cos');
651
652 // Plot the spectrum
653 plot(f,abs(fft));
654
655 // Save the spectrum
656 save('bb_tone.cos');
657
658 // Plot the spectrum
659 plot(f,abs(fft));
660
661 // Save the spectrum
662 save('bb_tone.cos');
663
664 // Plot the spectrum
665 plot(f,abs(fft));
666
667 // Save the spectrum
668 save('bb_tone.cos');
669
670 // Plot the spectrum
671 plot(f,abs(fft));
672
673 // Save the spectrum
674 save('bb_tone.cos');
675
676 // Plot the spectrum
677 plot(f,abs(fft));
678
679 // Save the spectrum
680 save('bb_tone.cos');
681
682 // Plot the spectrum
683 plot(f,abs(fft));
684
685 // Save the spectrum
686 save('bb_tone.cos');
687
688 // Plot the spectrum
689 plot(f,abs(fft));
690
691 // Save the spectrum
692 save('bb_tone.cos');
693
694 // Plot the spectrum
695 plot(f,abs(fft));
696
697 // Save the spectrum
698 save('bb_tone.cos');
699
700 // Plot the spectrum
701 plot(f,abs(fft));
702
703 // Save the spectrum
704 save('bb_tone.cos');
705
706 // Plot the spectrum
707 plot(f,abs(fft));
708
709 // Save the spectrum
710 save('bb_tone.cos');
711
712 // Plot the spectrum
713 plot(f,abs(fft));
714
715 // Save the spectrum
716 save('bb_tone.cos');
717
718 // Plot the spectrum
719 plot(f,abs(fft));
720
721 // Save the spectrum
722 save('bb_tone.cos');
723
724 // Plot the spectrum
725 plot(f,abs(fft));
726
727 // Save the spectrum
728 save('bb_tone.cos');
729
730 // Plot the spectrum
731 plot(f,abs(fft));
732
733 // Save the spectrum
734 save('bb_tone.cos');
735
736 // Plot the spectrum
737 plot(f,abs(fft));
738
739 // Save the spectrum
740 save('bb_tone.cos');
741
742 // Plot the spectrum
743 plot(f,abs(fft));
744
745 // Save the spectrum
746 save('bb_tone.cos');
747
748 // Plot the spectrum
749 plot(f,abs(fft));
750
751 // Save the spectrum
752 save('bb_tone.cos');
753
754 // Plot the spectrum
755 plot(f,abs(fft));
756
757 // Save the spectrum
758 save('bb_tone.cos');
759
760 // Plot the spectrum
761 plot(f,abs(fft));
762
763 // Save the spectrum
764 save('bb_tone.cos');
765
766 // Plot the spectrum
767 plot(f,abs(fft));
768
769 // Save the spectrum
770 save('bb_tone.cos');
771
772 // Plot the spectrum
773 plot(f,abs(fft));
774
775 // Save the spectrum
776 save('bb_tone.cos');
777
778 // Plot the spectrum
779 plot(f,abs(fft));
780
781 // Save the spectrum
782 save('bb_tone.cos');
783
784 // Plot the spectrum
785 plot(f,abs(fft));
786
787 // Save the spectrum
788 save('bb_tone.cos');
789
790 // Plot the spectrum
791 plot(f,abs(fft));
792
793 // Save the spectrum
794 save('bb_tone.cos');
795
796 // Plot the spectrum
797 plot(f,abs(fft));
798
799 // Save the spectrum
800 save('bb_tone.cos');
801
802 // Plot the spectrum
803 plot(f,abs(fft));
804
805 // Save the spectrum
806 save('bb_tone.cos');
807
808 // Plot the spectrum
809 plot(f,abs(fft));
810
811 // Save the spectrum
812 save('bb_tone.cos');
813
814 // Plot the spectrum
815 plot(f,abs(fft));
816
817 // Save the spectrum
818 save('bb_tone.cos');
819
820 // Plot the spectrum
821 plot(f,abs(fft));
822
823 // Save the spectrum
824 save('bb_tone.cos');
825
826 // Plot the spectrum
827 plot(f,abs(fft));
828
829 // Save the spectrum
830 save('bb_tone.cos');
831
832 // Plot the spectrum
833 plot(f,abs(fft));
834
835 // Save the spectrum
836 save('bb_tone.cos');
837
838 // Plot the spectrum
839 plot(f,abs(fft));
840
841 // Save the spectrum
842 save('bb_tone.cos');
843
844 // Plot the spectrum
845 plot(f,abs(fft));
846
847 // Save the spectrum
848 save('bb_tone.cos');
849
850 // Plot the spectrum
851 plot(f,abs(fft));
852
853 // Save the spectrum
854 save('bb_tone.cos');
855
856 // Plot the spectrum
857 plot(f,abs(fft));
858
859 // Save the spectrum
860 save('bb_tone.cos');
861
862 // Plot the spectrum
863 plot(f,abs(fft));
864
865 // Save the spectrum
866 save('bb_tone.cos');
867
868 // Plot the spectrum
869 plot(f,abs(fft));
870
871 // Save the spectrum
872 save('bb_tone.cos');
873
874 // Plot the spectrum
875 plot(f,abs(fft));
876
877 // Save the spectrum
878 save('bb_tone.cos');
879
880 // Plot the spectrum
881 plot(f,abs(fft));
882
883 // Save the spectrum
884 save('bb_tone.cos');
885
886 // Plot the spectrum
887 plot(f,abs(fft));
888
889 // Save the spectrum
890 save('bb_tone.cos');
891
892 // Plot the spectrum
893 plot(f,abs(fft));
894
895 // Save the spectrum
896 save('bb_tone.cos');
897
898 // Plot the spectrum
899 plot(f,abs(fft));
900
901 // Save the spectrum
902 save('bb_tone.cos');
903
904 // Plot the spectrum
905 plot(f,abs(fft));
906
907 // Save the spectrum
908 save('bb_tone.cos');
909
910 // Plot the spectrum
911 plot(f,abs(fft));
912
913 // Save the spectrum
914 save('bb_tone.cos');
915
916 // Plot the spectrum
917 plot(f,abs(fft));
918
919 // Save the spectrum
920 save('bb_tone.cos');
921
922 // Plot the spectrum
923 plot(f,abs(fft));
924
925 // Save the spectrum
926 save('bb_tone.cos');
927
928 // Plot the spectrum
929 plot(f,abs(fft));
930
931 // Save the spectrum
932 save('bb_tone.cos');
933
934 // Plot the spectrum
935 plot(f,abs(fft));
936
937 // Save the spectrum
938 save('bb_tone.cos');
939
940 // Plot the spectrum
941 plot(f,abs(fft));
942
943 // Save the spectrum
944 save('bb_tone.cos');
945
946 // Plot the spectrum
947 plot(f,abs(fft));
948
949 // Save the spectrum
950 save('bb_tone.cos');
951
952 // Plot the spectrum
953 plot(f,abs(fft));
954
955 // Save the spectrum
956 save('bb_tone.cos');
957
958 // Plot the spectrum
959 plot(f,abs(fft));
960
961 // Save the spectrum
962 save('bb_tone.cos');
963
964 // Plot the spectrum
965 plot(f,abs(fft));
966
967 // Save the spectrum
968 save('bb_tone.cos');
969
970 // Plot the spectrum
971 plot(f,abs(fft));
972
973 // Save the spectrum
974 save('bb_tone.cos');
975
976 // Plot the spectrum
977 plot(f,abs(fft));
978
979 // Save the spectrum
980 save('bb_tone.cos');
981
982 // Plot the spectrum
983 plot(f,abs(fft));
984
985 // Save the spectrum
986 save('bb_tone.cos');
987
988 // Plot the spectrum
989 plot(f,abs(fft));
990
991 // Save the spectrum
992 save('bb_tone.cos');
993
994 // Plot the spectrum
995 plot(f,abs(fft));
996
997 // Save the spectrum
998 save('bb_tone.cos');
999
1000 // Plot the spectrum
1001 plot(f,abs(fft));
1002
1003 // Save the spectrum
1004 save('bb_tone.cos');
1005
1006 // Plot the spectrum
1007 plot(f,abs(fft));
1008
1009 // Save the spectrum
1010 save('bb_tone.cos');
1011
1012 // Plot the spectrum
1013 plot(f,abs(fft));
1014
1015 // Save the spectrum
1016 save('bb_tone.cos');
1017
1018 // Plot the spectrum
1019 plot(f,abs(fft));
1020
1021 // Save the spectrum
1022 save('bb_tone.cos');
1023
1024 // Plot the spectrum
1025 plot(f,abs(fft));
1026
1027 // Save the spectrum
1028 save('bb_tone.cos');
1029
1030 // Plot the spectrum
1031 plot(f,abs(fft));
1032
1033 // Save the spectrum
1034 save('bb_tone.cos');
1035
1036 // Plot the spectrum
1037 plot(f,abs(fft));
1038
1039 // Save the spectrum
1040 save('bb_tone.cos');
1041
1042 // Plot the spectrum
1043 plot(f,abs(fft));
1044
1045 // Save the spectrum
1046 save('bb_tone.cos');
1047
1048 // Plot the spectrum
1049 plot(f,abs(fft));
1050
1051 // Save the spectrum
1052 save('bb_tone.cos');
1053
1054 // Plot the spectrum
1055 plot(f,abs(fft));
1056
1057 // Save the spectrum
1058 save('bb_tone.cos');
1059
1060 // Plot the spectrum
1061 plot(f,abs(fft));
1062
1063 // Save the spectrum
1064 save('bb_tone.cos');
1065
1066 // Plot the spectrum
1067 plot(f,abs(fft));
1068
1069 // Save the spectrum
1070 save('bb_tone.cos');
1071
1072 // Plot the spectrum
1073 plot(f,abs(fft));
1074
1075 // Save the spectrum
1076 save('bb_tone.cos');
1077
1078 // Plot the spectrum
1079 plot(f,abs(fft));
1080
1081 // Save the spectrum
1082 save('bb_tone.cos');
1083
1084 // Plot the spectrum
1085 plot(f,abs(fft));
1086
1087 // Save the spectrum
1088 save('bb_tone.cos');
1089
1090 // Plot the spectrum
1091 plot(f,abs(fft));
1092
1093 // Save the spectrum
1094 save('bb_tone.cos');
1095
1096 // Plot the spectrum
1097 plot(f,abs(fft));
1098
1099 // Save the spectrum
1100 save('bb_tone.cos');
1101
1102 // Plot the spectrum
1103 plot(f,abs(fft));
1104
1105 // Save the spectrum
1106 save('bb_tone.cos');
1107
1108 // Plot the spectrum
1109 plot(f,abs(fft));
1110
1111 // Save the spectrum
1112 save('bb_tone.cos');
1113
1114 // Plot the spectrum
1115 plot(f,abs(fft));
1116
1117 // Save the spectrum
1118 save('bb_tone.cos');
1119
1120 // Plot the spectrum
1121 plot(f,abs(fft));
1122
1123 // Save the spectrum
1124 save('bb_tone.cos');
1125
1126 // Plot the spectrum
1127 plot(f,abs(fft));
1128
1129 // Save the spectrum
1130 save('bb_tone.cos');
1131
1132 // Plot the spectrum
1133 plot(f,abs(fft));
1134
1135 // Save the spectrum
1136 save('bb_tone.cos');
1137
1138 // Plot the spectrum
1139 plot(f,abs(fft));
1140
1141 // Save the spectrum
1142 save('bb_tone.cos');
1143
1144 // Plot the spectrum
1145 plot(f,abs(fft));
1146
1147 // Save the spectrum
1148 save('bb_tone.cos');
1149
1150 // Plot the spectrum
1151 plot(f,abs(fft));
1152
1153 // Save the spectrum
1154 save('bb_tone.cos');
1155
1156 // Plot the spectrum
1157 plot(f,abs(fft));
1158
1159 // Save the spectrum
1160 save('bb_tone.cos');
1161
1162 // Plot the spectrum
1163 plot(f,abs(fft));
1164
1165 // Save the spectrum
1166 save('bb_tone.cos');
1167
1168 // Plot the spectrum
1169 plot(f,abs(fft));
1170
1171 // Save the spectrum
1172 save('bb_tone.cos');
1173
1174 // Plot the spectrum
1175 plot(f,abs(fft));
1176
1177 // Save the spectrum
1178 save('bb_tone.cos');
1179
1180 // Plot the spectrum
1181 plot(f,abs(fft));
1182
1183 // Save the spectrum
1184 save('bb_tone.cos');
1185
1186 // Plot the spectrum
1187 plot(f,abs(fft));
1188
1189 // Save the spectrum
1190 save('bb_tone.cos');
1191
1192 // Plot the spectrum
1193 plot(f,abs(fft));
1194
1195 // Save the spectrum
1196 save('bb_tone.cos');
1197
1198 // Plot the spectrum
1199 plot(f,abs(fft));
1200
1201 // Save the spectrum
1202 save('bb_tone.cos');
1203
1204 // Plot the spectrum
1205 plot(f,abs(fft));
1206
1207 // Save the spectrum
1208 save('bb_tone.cos');
1209
1210 // Plot the spectrum
1211 plot(f,abs(fft));
1212
1213 // Save the spectrum
1214 save('bb_tone.cos');
1215
1216 // Plot the spectrum
1217 plot(f,abs(fft));
1218
1219 // Save the spectrum
1220 save('bb_tone.cos');
1221
1222 // Plot the spectrum
1223 plot(f,abs(fft));
1224
1225 // Save the spectrum
1226 save('bb_tone.cos');
1227
1228 // Plot the spectrum
1229 plot(f,abs(fft));
1230
1231 // Save the spectrum
1232 save('bb_tone.cos');
1233
1234 // Plot the spectrum
1235 plot(f,abs(fft));
1236
1237 // Save the spectrum
1238 save('bb_tone.cos');
1239
1240 // Plot the spectrum
1241 plot(f,abs(fft));
1242
1243 // Save the spectrum
1244 save('bb_tone.cos');
1245
1246 // Plot the spectrum
1247 plot(f,abs(fft));
1248
1249 // Save the spectrum
1250 save('bb_tone.cos');
1251
1252 // Plot the spectrum
1253 plot(f,abs(fft));
1254
1255 // Save the spectrum
1256 save('bb_tone.cos');
1257
1258 // Plot the spectrum
1259 plot(f,abs(fft));
1260
1261 // Save the spectrum
1262 save('bb_tone.cos');
1263
1264 // Plot the spectrum
1265 plot(f,abs(fft));
1266
1267 // Save the spectrum
1268 save('bb_tone.cos');
1269
1270 // Plot the spectrum
1271 plot(f,abs(fft));
1272
1273 // Save the spectrum
1274 save('bb_tone.cos');
1275
1276 // Plot the spectrum
1277 plot(f,abs(fft));
1278
1279 // Save the spectrum
1280 save('bb_tone.cos');
1281
1282 // Plot the spectrum
1283 plot(f,abs(fft));
1284
1285 // Save the spectrum
1286 save('bb_tone.cos');
1287
1288 // Plot the spectrum
1289 plot(f,abs(fft));
1290
1291 // Save the spectrum
1292 save('bb_tone.cos');
1293
1294 // Plot the spectrum
1295 plot(f,abs(fft));
1296
1297 // Save the spectrum
1298 save('bb_tone.cos');
1299
1300 // Plot the spectrum
1301 plot(f,abs(fft));
1302
1303 // Save the spectrum
1304 save('bb_tone.cos');
1305
1306 // Plot the spectrum
1307 plot(f,abs(fft));
1308
1309 // Save the spectrum
1310 save('bb_tone.cos');
1311
1312 // Plot the spectrum
1313 plot(f,abs(fft));
1314
1315 // Save the spectrum
1316 save('bb_tone.cos');
1317
1318 // Plot the spectrum
1319 plot(f,abs(fft));
1320
1321 // Save the spectrum
1322 save('bb_tone.cos');
1323
1324 // Plot the spectrum
1325 plot(f,abs(fft));
1326
1327 // Save the spectrum
1328 save('bb_tone.cos');
1329
1330 // Plot the spectrum
1331 plot(f,abs(fft));
1332
1333 // Save the spectrum
1334 save('bb_tone.cos');
1335
1336 // Plot the spectrum
1337 plot(f,abs(fft));
1338
1339 // Save the spectrum
1340 save('bb_tone.cos');
1341
1342 // Plot the spectrum
1343 plot(f,abs(fft));
1344
1345 // Save the spectrum
1346 save('bb_tone.cos');
1347
1348 // Plot the spectrum
1349 plot(f,abs(fft));
1350
1351 // Save the spectrum
1352 save('bb_tone.cos');
1353
1354 // Plot the spectrum
1355 plot(f,abs(fft));
1356
1357 // Save the spectrum
1358 save('bb_tone.cos');
1359
1360 // Plot the spectrum
1361 plot(f,abs(fft));
1362
1363 // Save the spectrum
1364 save('bb_tone.cos');
1365
1366 // Plot the spectrum
1367 plot(f,abs(fft));
1368
1369 // Save the spectrum
1370 save('bb_tone.cos');
1371
1372 // Plot the spectrum
1373 plot(f,abs(fft));
1374
1375 // Save the spectrum
1376 save('bb_tone.cos');
1377
1378 // Plot the spectrum
1379 plot(f,abs(fft));
1380
1381 // Save the spectrum
1382 save('bb_tone.cos');
1383
1384 // Plot the spectrum
1385 plot(f,abs(fft));
1386
1387 // Save the spectrum
1388 save('bb_tone.cos');
1389
1390 // Plot the spectrum
1391 plot(f,abs(fft));
1392
1393 // Save the spectrum
1394 save('bb_tone.cos');
1395
1396 // Plot the spectrum
1397 plot(f,abs(fft));
1398
1399 // Save the spectrum
1400 save('bb_tone.cos');
1401
1402 // Plot the spectrum
1403 plot(f,abs(fft));
1404
1405 // Save the spectrum
1406 save('bb_tone.cos');
1407
1408 // Plot the spectrum
1409 plot(f,abs(fft));
1410
1411 // Save the spectrum
1412 save('bb_tone.cos');
1413
1414 // Plot the spectrum
1415 plot(f,abs(fft));
1416
1417 // Save the spectrum
1418 save('bb_tone.cos');
1419
1420 // Plot the spectrum
1421 plot(f,abs(fft));
1422
1423 // Save the spectrum
1424 save('bb_tone.cos');
1425
1426 // Plot the spectrum
1427 plot(f,abs(fft));
1428
1429 // Save the spectrum
1430 save('bb_tone.cos');
1431
1432 // Plot the spectrum
1433 plot(f,abs(fft));
1434
1435 // Save the spectrum
1436 save('bb_tone.cos');
1437
1438 // Plot the spectrum
1439 plot(f,abs(fft));
1440
1441 // Save the spectrum
1442 save('bb_tone.cos');
1443
1444 // Plot the spectrum
1445 plot(f,abs(fft));
1446
1447 // Save the spectrum
1448 save('bb_tone.cos');
1449
1450 // Plot the spectrum
1451 plot(f,abs(fft));
1452
1453 // Save the spectrum
1454 save('bb_tone.cos');
1455
1456 // Plot the spectrum
1457 plot(f,abs(fft));
1458
1459 // Save the spectrum
1460 save('bb_tone.cos');
1461
1462 // Plot the spectrum
1463 plot(f,abs(fft));
1464
1465 // Save the spectrum
1466 save('bb_tone.cos');
1467
1468 // Plot the spectrum
1469 plot(f,abs(fft));
1470
1471 // Save the spectrum
1472 save('bb_tone.cos');
1473
1474 // Plot the spectrum
1475 plot(f,abs(fft));
1476
1477 // Save the spectrum
1478 save('bb_tone.cos');
1479
1480 // Plot the spectrum
1481 plot(f,abs(fft));
1482
1483 // Save the spectrum
1484 save('bb_tone.cos');
1485
1486 // Plot the spectrum
1487 plot(f,abs(
```

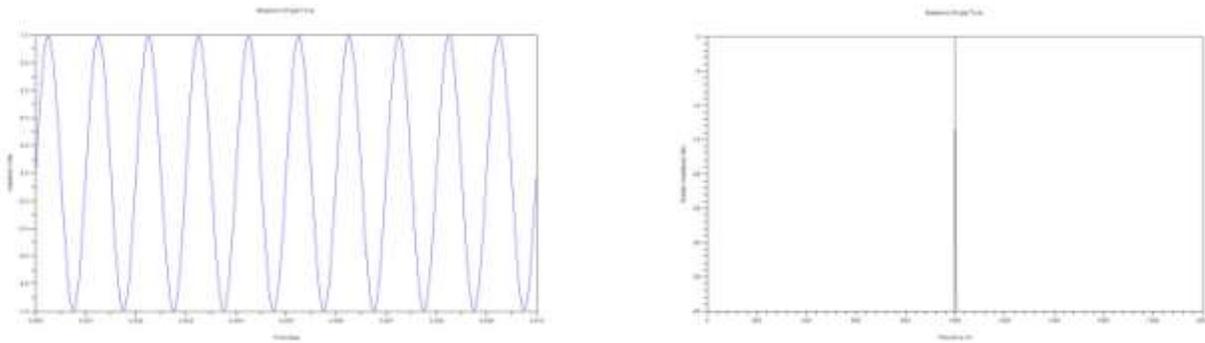


Fig 4.3 Baseband Analog Single Tone [bb_tone.sce](#) Scope & Spectral Display

4.2 Analog Baseband - Multiple Tones

In most cases the baseband will consist of a more complicated signal consisting of many tones. Speech for instance consists of a multiple tones and music consists of hundreds of tones. To get a feeling for this, **Fig 4.4** [bb_tones.sce](#) & **Fig 4.5** [bb_tones.cos](#) show the time and frequency domain displays of a baseband signal consisting of 3 tones with different frequencies and amplitudes:

- Tone 1 = 1KHz @ 3 Volt
- Tone 2 = 2KHz @ 2 Volt
- Tone 3 = 3KHz @ 1 Volt

```

1: Open the workspace
2: Set the tone parameters
3: Set the tone parameters
4: Set the tone parameters
5: Set the tone parameters
6: Set the tone parameters
7: Set the tone parameters
8: Set the tone parameters
9: Set the tone parameters
10: Set the tone parameters
11: Set the tone parameters
12: Set the tone parameters
13: Set the tone parameters
14: Set the tone parameters
15: Set the tone parameters
16: Set the tone parameters
17: Set the tone parameters
18: Set the tone parameters
19: Set the tone parameters
20: Set the tone parameters
21: Set the tone parameters
22: Set the tone parameters
23: Set the tone parameters
24: Set the tone parameters
25: Set the tone parameters
26: Set the tone parameters
27: Set the tone parameters
28: Set the tone parameters
29: Set the tone parameters
30: Set the tone parameters
31: Set the tone parameters
32: Set the tone parameters
33: Set the tone parameters
34: Set the tone parameters
35: Set the tone parameters
36: Set the tone parameters
37: Set the tone parameters
38: Set the tone parameters
39: Set the tone parameters
40: Set the tone parameters
41: Set the tone parameters
42: Set the tone parameters
43: Set the tone parameters
44: Set the tone parameters
45: Set the tone parameters
46: Set the tone parameters
47: Set the tone parameters
48: Set the tone parameters
49: Set the tone parameters
50: Set the tone parameters
51: Set the tone parameters
52: Set the tone parameters
53: Set the tone parameters
54: Set the tone parameters
55: Set the tone parameters
56: Set the tone parameters
57: Set the tone parameters
58: Set the tone parameters
59: Set the tone parameters
60: Set the tone parameters
61: Set the tone parameters
62: Set the tone parameters
63: Set the tone parameters
64: Set the tone parameters
65: Set the tone parameters
66: Set the tone parameters
67: Set the tone parameters
68: Set the tone parameters
69: Set the tone parameters
70: Set the tone parameters
71: Set the tone parameters
72: Set the tone parameters
73: Set the tone parameters
74: Set the tone parameters
75: Set the tone parameters
76: Set the tone parameters
77: Set the tone parameters
78: Set the tone parameters
79: Set the tone parameters
80: Set the tone parameters
81: Set the tone parameters
82: Set the tone parameters
83: Set the tone parameters
84: Set the tone parameters
85: Set the tone parameters
86: Set the tone parameters
87: Set the tone parameters
88: Set the tone parameters
89: Set the tone parameters
90: Set the tone parameters
91: Set the tone parameters
92: Set the tone parameters
93: Set the tone parameters
94: Set the tone parameters
95: Set the tone parameters
96: Set the tone parameters
97: Set the tone parameters
98: Set the tone parameters
99: Set the tone parameters
100: Set the tone parameters

```

Fig 4.4 Baseband Analog Tones [bb_tones.sce](#)

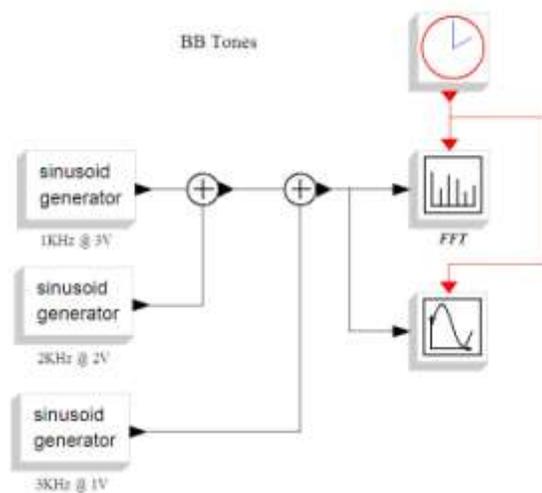


Fig 4.5 Baseband Analog Tones [bb_tones.cos](#)

In general, the sum of many sinusoids of different amplitudes and phases but the same frequency will give a sinusoid at that frequency with a different amplitude and phase given by the vector addition of each component. However, the addition of sinusoids of different frequencies, amplitudes and phases will give rise to a non-sinusoidal periodic signal. This is further explained in [bb_analog.mp4](#).

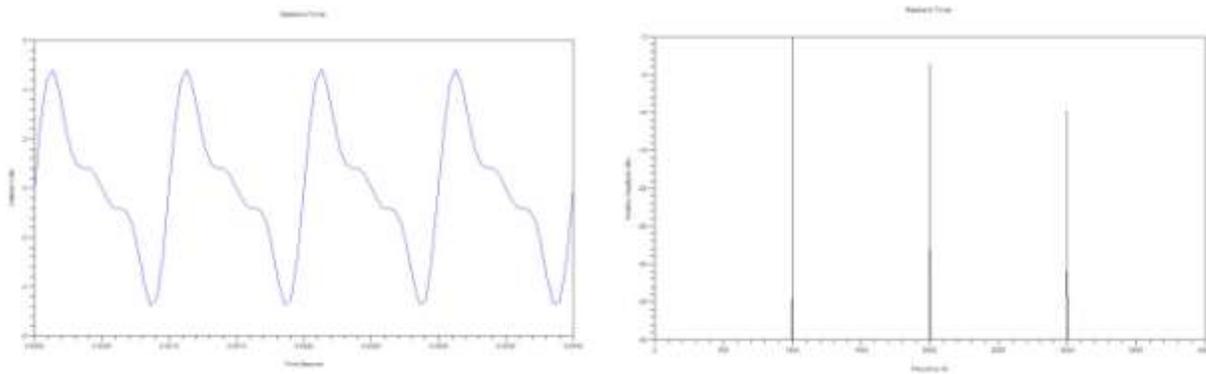


Fig 4.6 Baseband Analog Tones [bb_tones.sce](#) Scope & Spectral Display

From **Fig 4.6** we see that the addition of the 1KHz, 2KHz and 3KHz tones gives rise to a non-sinusoidal periodic wave of 1KHz.

Consider the spectrum of the baseband tones in **Fig 4.6**. Note that there is a discrete line for each component frequency. We can convert the ratio of the voltage of each component and see if the relative spectrum amplitude indicates the same thing:

Component	Relative Amplitude	$\text{dB}=20\log(\text{V}/\text{Vref})$	Fig 4.6 Spectrum
1KHz	$3/3 = 1$ reference	0 dBc	0 dBc
2KHz	$2/3$	-3.5dBc	-3.5dBc
3KHz	$1/3$	-9.5dBc	-9.5dBc

4.3 Baseband Analog - WAV

Now consider what happens when we have a real speech signal consisting of energy at many frequencies. **Fig 4.7** [bb_wav.sce](#) & **Fig 4.8** [bb_wav.cos](#) examine the speech file 'demo.wav' to examine its waveform and spectrum.

In order to process the signal, we need to know the sample rate and duration. By opening 'demo.wav' in Windows Sound Recorder, the sample rate is 22050Hz and time 36.75 seconds. In **Fig 4.7**, the file is read in using the 'wavread' routine. The file is then limited to 4 seconds = $22050 \times 4 = 88,200$ samples. An equivalent time vector is created to match this data. The waveform is then plotted and spectrum calculated.

In **Fig 4.8**, a similar process is used. A script file [wav_read.sce](#) is used to read in the sound data and associate it with a structure variable **V**. **V** is then read into the block file. **V** contains both the sample data and time data.

Note the nature of a speech sound signal shown in **Fig 4.9**. It is bursty in nature when the vocal chords are active, with relatively long periods of zero activity between bursts. The spectrum indicates that for this speech sample, the majority of signal energy lies between DC and about 3500Hz, with a peak energy around 300Hz.

```
1 //This file simulates a WAV baseband
2 //A file 'demo.wav' is read into the workspace
3 //File is 3.74 seconds long
4 //Fs = sampling frequency 22050Hz
5 //J.Clark April2nd-2012
6
7
8 //Read in 'demo.wav' & truncate to 4 secs
9 //Truncate to 4secs
10 V=wavread('demo.wav');
11 V=V(1:88201);
12 Fs=22050;
13 Ts=1/Fs;
14 n=length(V);
15 h=ceil(n/2);
16 cF(h);
17 plot(1:n);
18
19
20 //Graph the tone waveform
21 //set(0) open graphic window #0
22 set(0);
23 plot(1:n);
24 write('Baseband WAV', 'Time Seconds', 'Relative Amplitude')
25
26
27 //Calculate the spectrum with fft
28 //set(1) open graphic window #1
29 set(1);
30 y=fft(V);
31 sample_rate=1/Ts;
32 f=sample_rate*(0:(h/2))/h;
33 n=length(f);
34 m=max(abs(y(1:n)));
35 plot2(f,20*log10(abs(y(1:n))/m));
36 xtitle('Baseband WAV', 'Frequency Hz', 'Relative Amplitude dB')
37
```

Fig 4.7 Baseband Analog WAV
[bb_wav.sce](#)

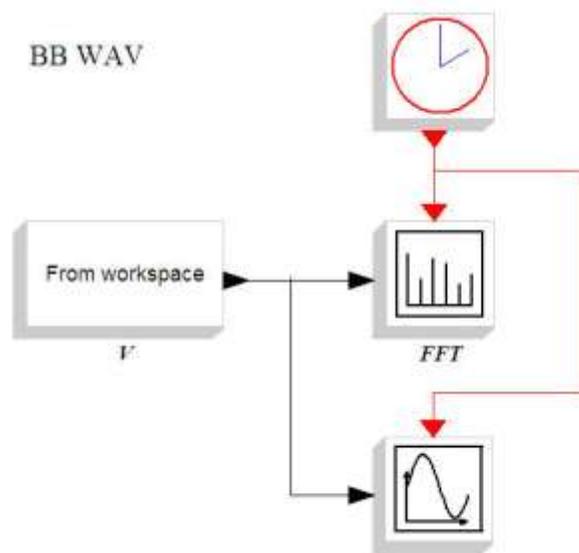


Fig 4.8 Baseband Analog WAV
[bb_wav.cos](#)

Note: Copy [demo.wav](#) from the /chapter_bb/wav to the desktop for ScicosLab to work

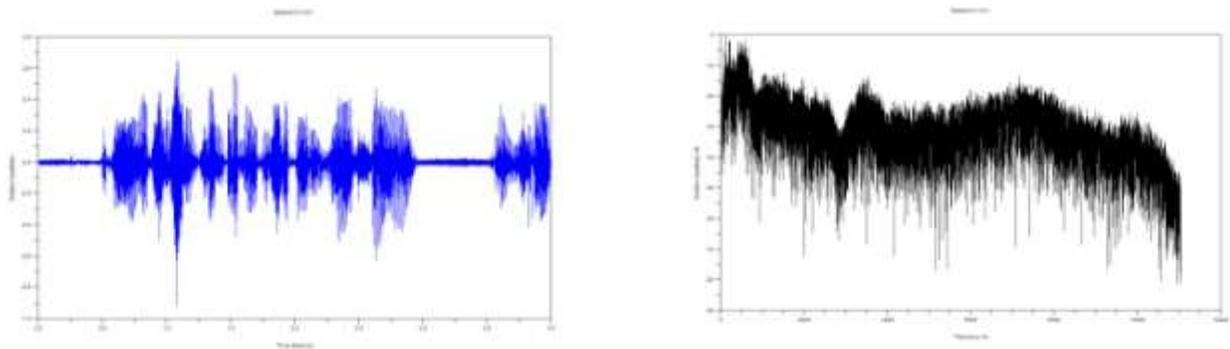


Fig 4.9 Baseband Analog WAV [bb_wav.sce](#) Scope & Spectral Display

4.4 Baseband Analog Signal Captures

Now let's look at some real signals to see how they compare to the simulations. The first signal is an audio tone of 1KHz. The second signal is a microphone signal. The Video [bb_an_sigcap.mp4](#) describes the signal captures.

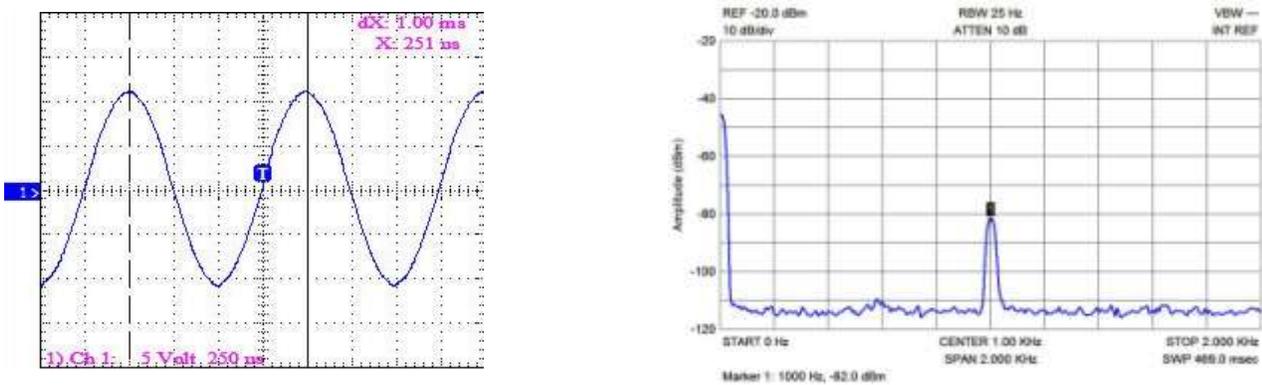


Fig 4.10 Baseband Analog Signal Capture Tone Scope & Spectral Display

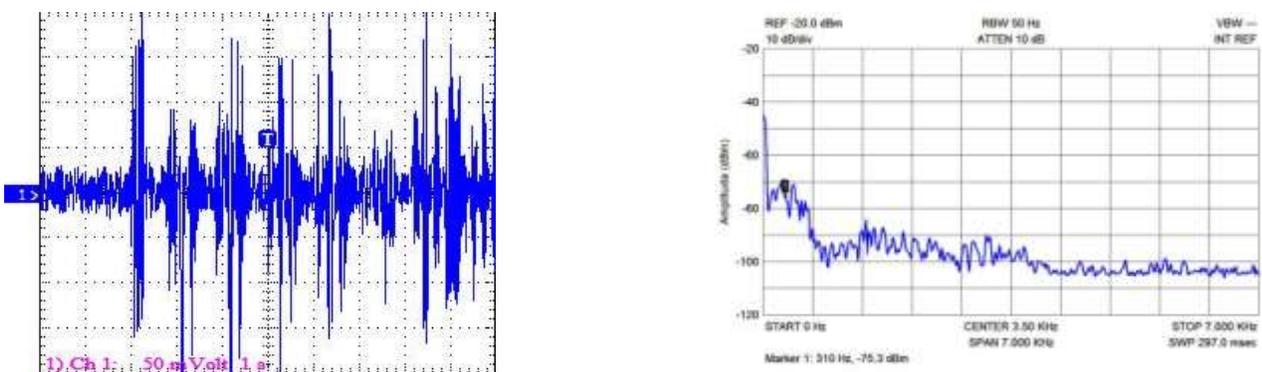


Fig 4.11 Baseband Analog Signal Capture WAV Scope & Spectral Display

4.5 Baseband Digital - NRZ Random Data

So far we have considered a baseband consisting of analog signals. Let's look at a digital data signal. **Fig 4.12** & **Fig 4.13** generate the digital baseband for a NRZ random data signal. The video [bb_digital.mp4](#) describes this.

```
%% NRZ NRZ Random Data Signal
% Parameters
fs = 1000; % Sampling rate in Hz
N = 10000; % Number of samples
% Generate random data signal
randData = rand(1, N);
% Convert to NRZ signal
NRZData = (randData > 0.5) * 5;
% Plot the signal
plot(NRZData);
title('NRZ Random Data Signal');
xlabel('Sample Index');
ylabel('Amplitude (Volts)');
axis([0 N 0 5]);
```

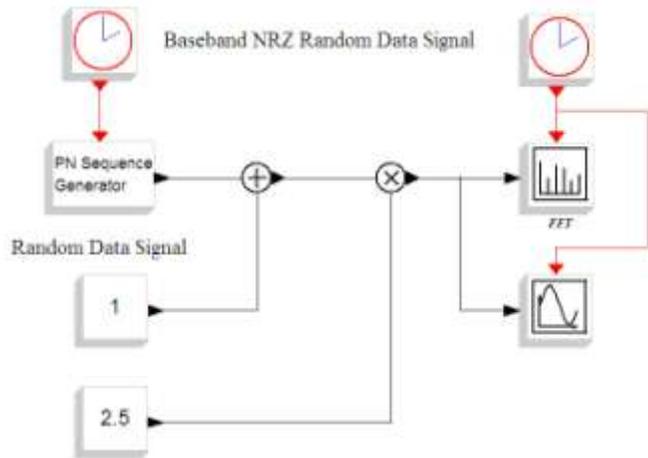


Fig 4.12 Baseband Digital NRZ Data
[bb_rnd_data.sce](#)

Fig 4.13 Baseband Digital NRZ Data
[bb_rnd_data.cos](#)

In **Fig 4.12**, random data is generated by the 'rand' function. This generates a random number between 0 & 1. This is declared to be 5 volts if ≥ 0.5 , or 0 volts if < 0.5 . In **Fig 4.13**, the random data is generated using a PN Sequence Generator. This generates a random data pattern that repeats after $2^N - 1$ bits. This generates a slightly different random pattern with longer sequences of 0's and 1's.

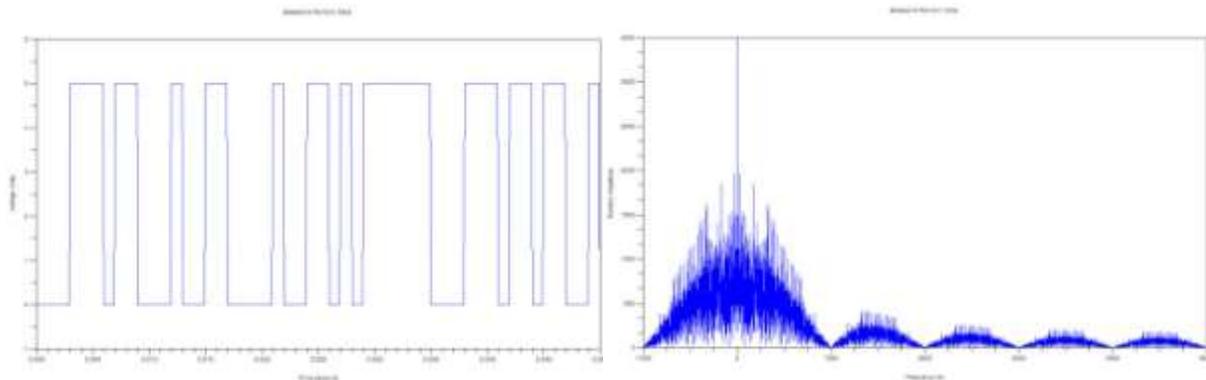


Fig 4.14 Baseband Digital NRZ Data [bb_rnd_data.sce](#) Scope & Spectral Display

Note the defining characteristics for NRZ random data. First of all the spectrum contains no discrete lines except at 0Hz. For an NRZ signal, there is a DC component of approx 2.5volts since the signal is on average half way between 0 volts and 5 volts.

The spectrum has a so called $(\sin x/x)$ shape **Fig 4.15**, with nulls at exactly the data rate and multiples thereof. The data rate is 1Kbps, so the nulls are at 1KHz, 2KHz etc. Note that the data rate null at 1KHz = $1/(\text{pulse width for a bit}) = 1/1\text{msec} = 1\text{KHz}$. In summary:

- DC Component for NRZ data
- 1st null at data rate and at integer multiples thereof
- Smooth shape, no spectral lines
- Nulls decreasing amplitude



Fig 4.15 Shape = $\sin x/x$

4.6 Baseband Digital - Manchester Random Data

NRZ is the basic data type generated by computing devices. However, it has several disadvantages for transmission on wire pairs etc. Manchester coded data is very popular for cable transmission in Ethernet. In Manchester coding, a '1' is encoded with a +V to -V transition and a '0' is encoded by a -V to +V transition. This ensures that for every data bit, there is equal positive and negative voltage. This means the overall DC voltage is 0 volts. The effective bit time is halved, so the bandwidth or required spectrum is doubled.

```

% Fig 4.16 Baseband Digital Manchester Data
% bb_rnd_data_man.sce
%
% Parameters
% -----
% Data rate (bits/sec)
data_rate = 1000;
% Manchester encoding period (sec)
Tm = 1/data_rate;
%
% Random Data Signal
% -----
% Generate a random data signal of length 1000 bits
random_data = randi(2, 1, 1000);
%
% Manchester Encoding
% -----
% Generate a Manchester Data Signal
manchester_data = manchester_encode(random_data, Tm);
%
% Plotting
% -----
% Plot the random data signal
plot(1:length(random_data), random_data);
title('Random Data Signal');
%
% Plot the Manchester Data Signal
plot(1:length(manchester_data), manchester_data);
title('Manchester Data Signal');
%
% FFT
% -----
% Compute the FFT of the Manchester Data Signal
fft_result = fft(manchester_data);
%
% Plot the magnitude spectrum
plot(abs(fft_result)/length(fft_result));
title('Magnitude Spectrum');
%
% Plot the time-domain signal
plot(1:length(manchester_data), manchester_data);
title('Manchester Data Signal (Time Domain)');
%
% Plot the frequency-domain signal
plot(abs(fft_result)/length(fft_result));
title('Manchester Data Signal (Frequency Domain)');
%
% End of script

```

Fig 4.16 Baseband Digital Manchester Data
[bb_rnd_data_man.sce](#)

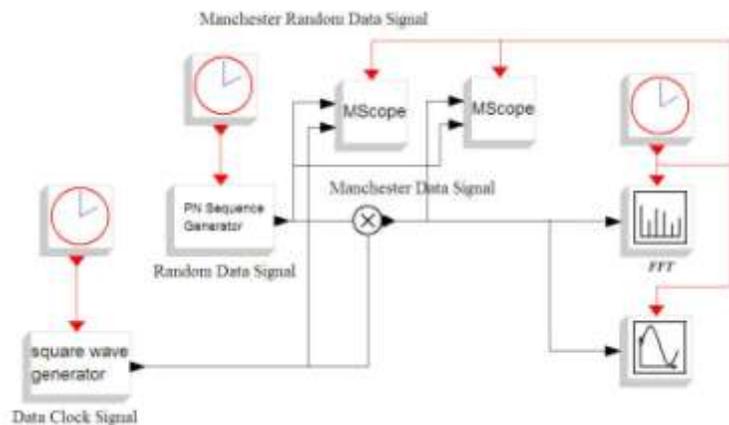


Fig 4.17 Baseband Digital Man.Data
[bb_rnd_data_man.cos](#)

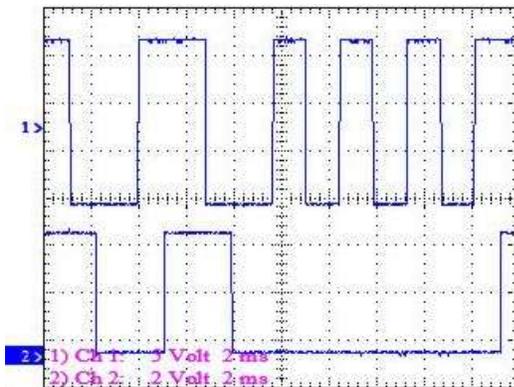


Fig 4.20 Baseband Digital Sigcap. NRZ & Man Random Data Scope Display

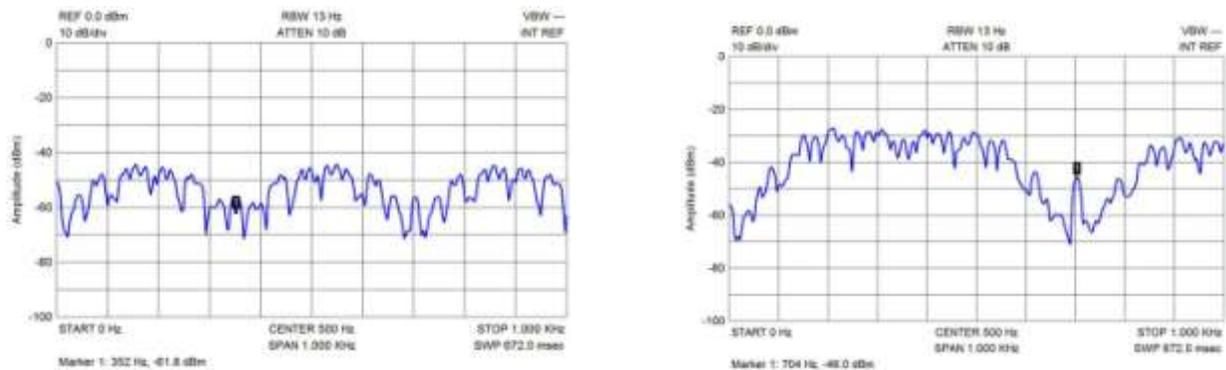


Fig 4.21 Baseband Digital Sigcap. NRZ & Man Random Data Spectral Displays

In **Fig 4.20** The NRZ data is shown on the bottom trace and the Manchester data is shown on the upper trace. For an NRZ “1” at 5 volts, the Manchester data goes from +12 volts to - 12 volts. For an NRZ “0” at 0 volts, the Manchester data goes from -12 volts to + 12 volts. In **Fig 4.21** The NRZ spectrum has a data null at approx. 352Hz and the Manchester spectrum null is at 704Hz or twice this amount. Note that these nulls represent the bit rate and twice the bit rate.

-Bit Time = 2.8msec (**Fig 4.20**)

-Data Rate = 1/Bit Time = 1/2.8msec = 357bps

Note that there is no DC component on the NRZ data. The spectrum analyzer cannot support DC voltage on the input, so there is a DC blocking capacitor inserted for instrument protection.

4.8 Baseband Analog & Digital - Exercises

1.

The spectrum of an unknown signal contains a spectral line at 32.768KHz? What can we say about this signal?

2.

An unknown signal has a totally noise like spectrum. What can we say about this signal?

3.

Construct a signal consisting of the following components:

$$V1(t)=4\sin(\omega_1 t) \quad F1=1\text{KHz}$$

$$V2(t)=2\sin(\omega_2 t) \quad F2=8\text{KHz}$$

$$V3(t)=5\sin(\omega_3 t) \quad F3=15\text{KHz}$$

$$V(t)=V1(t)+V2(t)+V3(t) \quad \omega_n=2\pi F_n$$

Plot the time and frequency displays. How would you classify this signal?

4.

Using Windows Sound Recorder or a similar program, record approx 4 seconds of your voice at a sampling rate of 44.1KHz. Modify the ScicosLab files to plot your voice in the time and frequency domain (Adjust either .sce/.cos files to accommodate different sampling rate of 44.1KHz vs 22.05KHz). Where is your voice spectrum most powerful? For your voice display, what proportion of the 4 seconds is the voltage approx = 0 volts?

5.

A random data stream of NRZ data is produced by a microcontroller. The '1' state is represented by 3.3 VDC and the '0' state by 0 VDC. What is the average voltage of the waveform or the DC value?

6.

An NRZ data stream has a bit rate of 1.544Mbps. Describe the waveform in the frequency domain.

7.

Why does Manchester coded data have twice the bandwidth of NRZ?

8.

What particular advantage does Manchester coding have over NRZ?