

Lab Manual:  
Communications Principles  
  
Using the EMONA Communications board for NI ELVIS III



Lab 13: Introduction to DSSS   
(Spread Spectrum)

List of Updates

|  |  |
| --- | --- |
| **Date** | **Details** |
| 4/1/2018 | Completed final document |
| 7/8/2018 | Improved Imagery, Corrected Errors |
|  |  |

**© 2018 EMONA Instruments Pty Ltd**

All Emona TIMS/ETT-Series/DxIQ user manuals, experiment manuals and supplied software   
are (C) Copyright to Emona Instruments Pty Ltd and its related entities. All rights reserved.

LIMITED AUTHORITY TO COPY TIMS MANUALS

This License Agreement grants a limited authority only to those educational institutions who have purchased the Emona TIMS/ETT/DxIQ laboratory learning equipment, to reproduce (in whole or in part,), and/or to give away copies of any of Emona Instrument’s published TIMS/ETT/DxIQ User Manuals and Experiment Manuals for the exclusive use of their own enrolled students.

No licensing fees are payable to Emona under this limited Authority.

Emona Instruments Pty Ltd retains the copyright of any edited and/or derivative documents.

**SOFTWARE**

EMONA Instruments Pty Ltd respects the intellectual property of others, and we ask our readers to do the same. This resource is protected by copyright and other intellectual property laws.

LabVIEW and National Instruments are trademarks of National Instruments.

All other trademarks or product names are the property of their respective owners.

**ADDITIONAL DISCLAIMERS**

The reader assumes all risk of use of this resource and of all information, theories, and programs contained or described in it. This resource may contain technical inaccuracies, typographical errors, other errors and omissions, and out-of-date information. Neither the author nor the publisher assumes any responsibility or liability for any errors or omissions of any kind, to update any information, or for any infringement of any patent or other intellectual property right.

Neither the author nor the publisher makes any warranties of any kind, including without limitation any warranty as to the sufficiency of the resource or of any information, theories, or programs contained or described in it, and any warranty that use of any information, theories, or programs contained or described in the resource will not infringe any patent or other intellectual property right. THIS RESOURCE IS PROVIDED “AS IS.” ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY AND ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS, ARE DISCLAIMED.

No right or license is granted by publisher or author under any patent or other intellectual property right, expressly, or by implication or estoppel.

IN NO EVENT SHALL THE PUBLISHER OR THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, COVER, ECONOMIC, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS RESOURCE OR ANY INFORMATION, THEORIES, OR PROGRAMS CONTAINED OR DESCRIBED IN IT, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, AND EVEN IF CAUSED OR CONTRIBUTED TO BY THE NEGLIGENCE OF THE PUBLISHER, THE AUTHOR, OR OTHERS. Applicable law may not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

Table of Contents

[Lab 13: Introduction to DSSS (Spread Spectrum) 5](#_Toc519683601)

[Learning Objectives 5](#_Toc519683602)

[Prerequisites 5](#_Toc519683603)

[Required Tools and Technology 6](#_Toc519683604)

[Expected Deliverables 7](#_Toc519683605)

[Section 1: DSSS Modulation 8](#_Toc519683606)

[1.1 Theory and Background 8](#_Toc519683607)

[1.2 Implement: Generating a DSSS signal using a simple message 10](#_Toc519683608)

[1.3 Implement: Observations of DSSS signals in the frequency domain 14](#_Toc519683609)

[1.4 Implement: Using the product detector to recover the message 18](#_Toc519683610)

[Section 2: DSSS and deliberate interference (jamming) 20](#_Toc519683611)

[2.1 Implement: DSSS and deliberate interference (jamming) 21](#_Toc519683612)

[Section 3: Expansion activities 27](#_Toc519683613)

[Figure 1: Patching for DSSS with analog message 12](#_Toc518826016)

[Figure 2: Block diagram for DSSS with analog message 12](#_Toc518826017)

[Figure 3: Patching for DSSS with square wave 15](#_Toc518826018)

[Figure 4: Block diagram for DSSS with square wave 16](#_Toc518826019)

[Figure 5: Patching for product demodulation of DSSS 18](#_Toc518826020)

[Figure 6: Block diagram for product demodulation 19](#_Toc518826021)

[Figure 7: Complete block diagram for product demodulation 19](#_Toc518826022)

[Figure 8: Demodulation with incorrect sequence 20](#_Toc518826023)

[Figure 9: Patching for DSSS jamming with a single frequency 22](#_Toc518826024)

[Figure 10: Block diagram for DSSS jamming with single frequency 23](#_Toc518826025)

[Figure 11: Patching for DSSS with FM jamming signal 24](#_Toc518826026)

[Figure 12: Spectrum of DSSS signal with FM jamming signal 25](#_Toc518826027)

[Figure 13: Patching for DSSS jamming with noise signal 26](#_Toc518826028)

# Lab 13: Introduction to DSSS (Spread Spectrum)

In this lab you will create a DSSS signal with a simple message, explore its characteristics in the frequency domain, understand how DSSS is recovered with a unique key and why it is resistant to jamming and interference.

## Learning Objectives

After completing this lab, you should be able to complete the following activities.

1. Generate a real DSSS signal using a simple message
2. Describe the spectrum of a DSSS signal
3. Discuss the recovery of a DSSS signal: what works, what doesn’t
4. Discuss the resistance of DSSS to jamming signals

## Prerequisites

You should have completed Lab 1 and Lab 2 and be familiar with the equipment, its use and the handling precautions for the equipment.

## Required Tools and Technology

|  |  |
| --- | --- |
| Platform: NI ELVIS III Instruments used in this lab:   * Oscilloscope-Time * Oscilloscope-FFT * Function Generator | * Install Instruments: [http://www.ni.com/documentation/en/ni-elvis-iii/latest/getting-started/installing-the-soft-front-panel/](http://www-preview.ni.com/documentation/en/ni-elvis-iii/1.0/getting-started/installing-the-soft-front-panel/) * Access instruments <https://measurementslive.ni.com> * View User Manual <http://www.ni.com/en-us/support/model.ni-elvis-iii.html>   View tutorials <https://www.youtube.com/playlist?list=PLvcPIuVaUMIWm8ziaSxv0gwtshBA2dh_M> |
| Hardware: Emona Communications Board Components used in this lab:   * Four BNC to 2mm banana-plug leads * Assorted 2mm banana-plug patch leads * Set of headphones or earbuds | * View User Manual <http://www.ni.com/en-us/support/model.emona-communications-board-for-ni-elvis-iii.html> |
| Software: NI ELVIS III Function Generator File used in this lab (available in lab folder):   * ECB\_120k-noise.csv | * Access instrument <https://measurementslive.ni.com> |

## 

## Expected Deliverables

In this lab, you will collect the following deliverables:

* Calculations
* Data from measurements
* Observations

Your instructor may expect you complete a lab report. Refer to your instructor for specific requirements or templates.

## Section 1: DSSS Modulation

## Theory and Background

Recall that when a sinusoidal carrier is DSBSC modulated by a message, the two signals are multiplied together. Recall also that the resulting DSBSC signal consists of two sets of sidebands but no carrier (refer to the preliminary discussion of Experiment 6 for a discussion).

When the DSBSC signal is demodulated using product detection, both sidebands are multiplied with a local carrier that must be synchronized to the transmitter’s carrier (that is, it has the same frequency and phase). Doing so produces two messages that are in-phase with each other and so add to form a single bigger message (refer to the preliminary discussion of Experiment 9 for a discussion).

*Direct sequence spread spectrum* (DSSS or often just “spread spectrum”) is a variation of the DSBSC modulation scheme with a pulse train (called a *pseudo-noise* sequence or just PN sequence) for the carrier instead of a simple sinewave. This may sound radical until you remember that pulse trains are actually made up of a theoretically infinite number of sinewaves (the *fundamental* and *harmonics*). That being the case, spread spectrum is really the DSBSC modulation of a theoretically infinite number of sinusoidal carrier signals. The result is a theoretically infinite number of pairs of tiny sidebands about a suppressed carrier.

In practice, not all of these sidebands have any energy of significance. However, the fact that the message information is distributed across so many of them makes spread spectrum signals difficult to deliberately interfere with or “jam”. To do so, you would have to upset a significant number of the sidebands which is difficult considering their number.

Spread spectrum signals are demodulated in the same way as DSBSC signals using a product detector. Importantly, the product detector’s local carrier signal must contain all the sinewaves that make up transmitter’s pulse train at the same frequency and phase. If this is not done, the tiny demodulated signals will be at the wrong frequency and phase and so they won’t add up to reproduce the original message. Instead, they’ll produce a garbage signal that looks like noise.

The only way for the receiver to generate the right number of sinewaves at the right frequency is to use a pulse train with an identical sequence to that used by the transmitter. Moreover, it must be synchronized. This issue gives spread spectrum another of its advantages over other modulation schemes. The transmitted signal is effectively encrypted.

Of course, with trial and error it’s possible for an unauthorized person to guess the correct PN sequence to use for their receiver. However, this can be made difficult by making the sequence longer before it repeats itself (that is, by making it consist of more bits or *chips*). Longer sequences can produce more combinations of unique codes which would take longer to guess using a trial and error approach. To illustrate this point, an 8-bit code has 256 combinations while a 20-bit code has 1,048,575 combinations. A 256-bit code has 1.1579×1077 combinations. That’s 11579 with 73 zeros after it!

Increasing the sequence’s chip-length has another advantage. To explain, the total energy in a spread spectrum signal is distributed between all of the tiny DSBSC that make it up (though not evenly because not all of the sinewaves that make up the carrier’s pulse train are the same amplitude). A mathematical technique called *Fourier Analysis* shows that the greater the number of chips in a sequence before repeating, the greater the number of sinewaves of significance needed to make it.

That being the case, using more chips in the transmitter’s PN sequence produces more DSBSC signals and so the signal’s total energy is distributed more thinly between them. This in turn means that the individual signals are many and extremely small. In fact, if the PN sequence is long enough, all of these DSBSC signals are smaller than the background electrical noise that’s always present in free-space. This fact gives spread spectrum yet another important advantage. The signal is difficult to detect.

Spread spectrum finds use in several digital applications including: CDMA mobile phone technology, cordless phones, the global positioning system (GPS) and two of the 802.11 wi-fi standards.

## 1.2 Implement: Generating a DSSS signal using a simple message

For this experiment you’ll use the EMONA Communications board to generate a DSSS signal by implementing its mathematical model. You’ll then use a product detector (with a stolen carrier) to reproduce the message. Once done, you’ll examine the importance of using the correct PN sequence for the local carrier and the difficulty of jamming DSSS signals.

It should take you about 50 minutes to complete this experiment.

**Powering up the ELVIS III + EMONA Communications Board**

|  |  |
| --- | --- |
| 1. | Ensure that the NI ELVIS III Application Board power button at the top left corner of the unit is OFF (not illuminated). |

|  |  |
| --- | --- |
| 2. | Carefully plug the Emona Communications board (ECB) into the NI ELVIS III ensuring that it is fully engaged both front and back. |

|  |  |
| --- | --- |
| 3. | Ensure that you have connected the NI ELVIS III to the PC using the USB cable and that the PC is turned on. |

|  |  |
| --- | --- |
| 4. | Turn on the Application Board *Power* button by pressing it once and confirm that it is illuminated. The LEDs on the ECB should also be illuminated. If they are not, then switch the unit off immediately and check for connection or insertion errors. |

|  |  |
| --- | --- |
| 5. | Open the Instrument Launcher software in your browser and select the required instruments. |

Table 1 Scope Configuration

|  |  |
| --- | --- |
| Channel Voltage Range | 2 V/div |
| Horizontal Timebase | 100us/div |
| Trigger | Analog Edge, Channel 1, Rising |
| Probe Attenuation | 1x |

As DSSS is basically just DSBSC with a PN sequence for the carrier instead of a simple sinusoid, it can be generated by implementing the mathematical model for DSBSC.

|  |  |
| --- | --- |
| 6. | Connect the set-up shown in Figure 1.  **Note:** Insert the black plugs of the oscilloscope leads into a ground (*GND*) socket. |

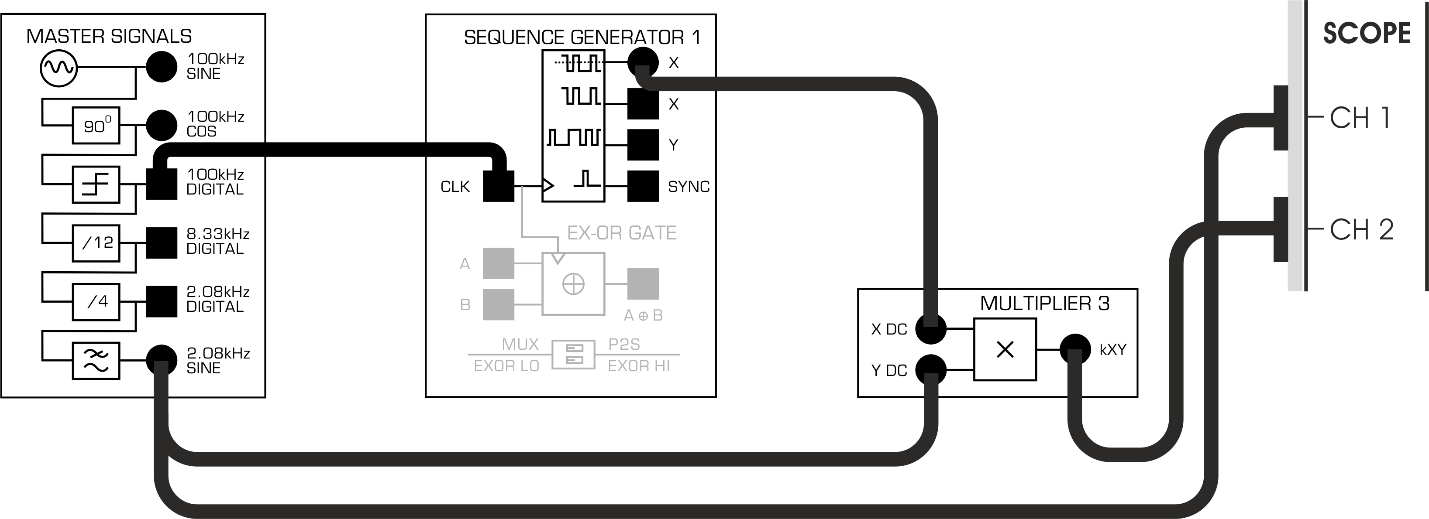


Figure 1: Patching for DSSS with analog message

This set-up can be represented by the block diagram in Figure 2. It multiplies the 2.08kHz sinewave message with a PN sequence modelled by the Sequence Generator’s bipolar 31-bit pulse train output *X*.

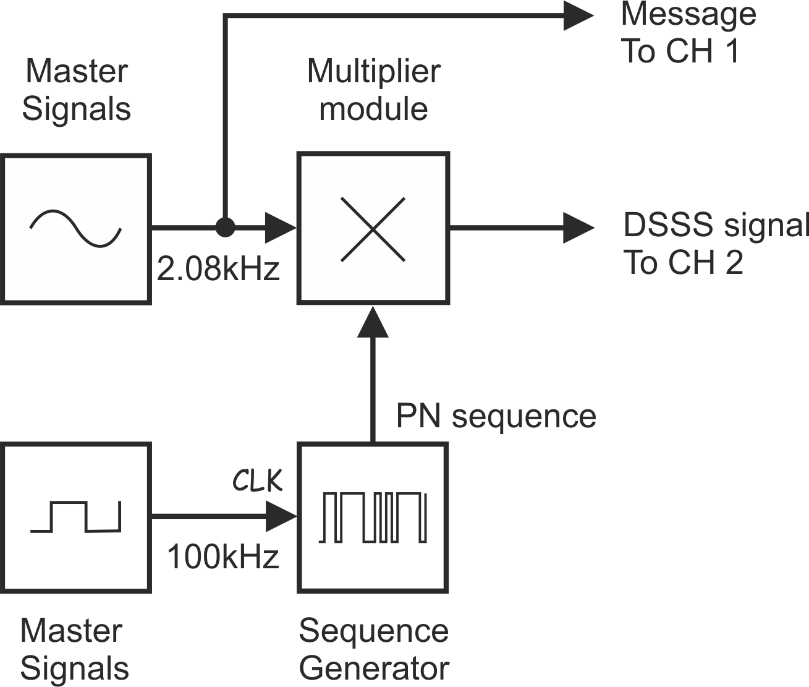


Figure 2: Block diagram for DSSS with analog message

|  |  |
| --- | --- |
| 7. | Launch and run the NI ELVIS III Oscilloscope. |

|  |  |
| --- | --- |
| 8. | Activate the scope’s *Channel 1* and *Channel 2* inputs to observe both the DSSS signal out of the Multiplier module as well as the 2.08kHz message signal. |

|  |  |
| --- | --- |
| 9. | Capture a screenshot of the scope and append to your report. Annotate your report appropriately so as to identify the waveforms captured. Use the cursors to highlight important levels and transition points in the waveform if necessary. |

1-1 What feature of the Multiplier module’s output suggests that it’s basically a DSBSC signal? **Tip:** If you’re not sure, read the preliminary discussion for Experiment 6.

|  |
| --- |
|  |
|  |
|  |

1-2 Why is the DSSS signal so large when it’s supposed to be small and indistinguishable from noise? **Tip:** If you’re not sure, see the preliminary discussion for this experiment.

|  |
| --- |
|  |
|  |
|  |

## 1.3 Implement: Observations of DSSS signals in the frequency domain

One of the features of DSSS is that it produces a theoretically infinite number of pairs of tiny sidebands with each pair straddling a suppressed carrier. This part of the experiment lets you examine this.

|  |  |
| --- | --- |
| 1. | Launch and run the NI ELVIS III Function Generator instrument. |

Table 1 Function Generator Configuration

|  |  |
| --- | --- |
| Waveform | Square |
| Frequency | 30kHz |
| Amplitude | 2Vpk |
| DC Offset | 0V |

|  |  |
| --- | --- |
| 2. | Disconnect the plug to the Sequence Generator module’s *analog X* output and modify the set-up as shown in Figure 3. |

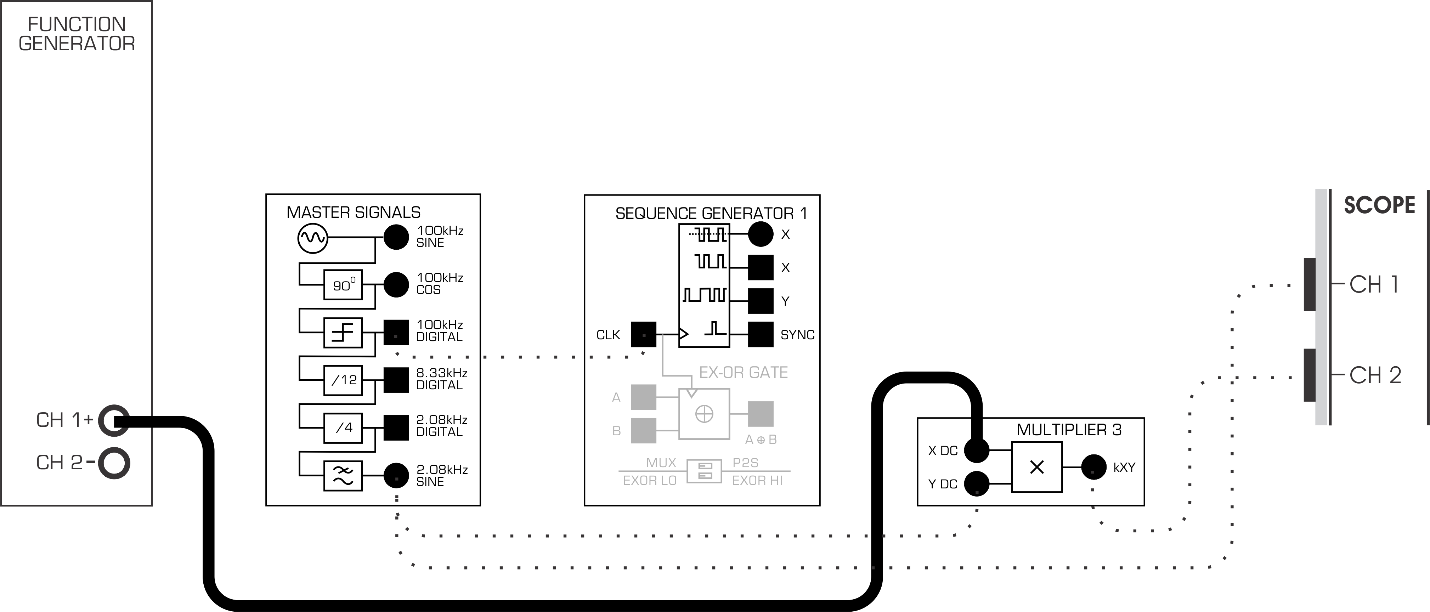


Figure 3: Patching for DSSS with square wave

|  |  |
| --- | --- |
| 3. | Examine the new DSSS signal on the scope.  **Note:** You should notice that it looks similar to the DSSS signal you obtained earlier. That said, it’ll be different in that the spacing between the carrier’s transitions are regular. |

The set-up in Figure 3 can be represented by the block diagram in Figure 4. Notice that the carrier signal is a 30kHz square wave. This is significantly different than what you have encountered in previous modulation schemes.

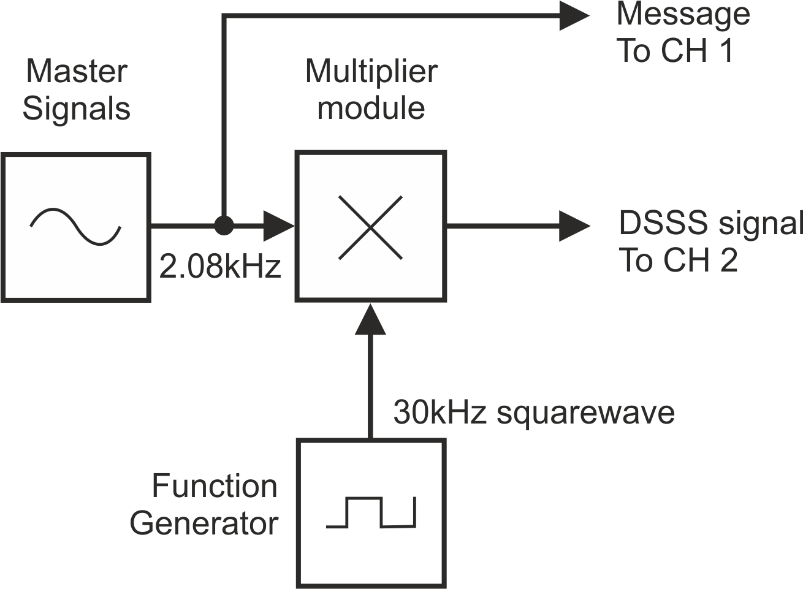


Figure 4: Block diagram for DSSS with square wave

Recall that a square wave consists of a fundamental at the same frequency as the square wave itself and a theoretically infinite number of odd harmonics (each with proportionally smaller amplitude to the amplitude of the frequency before it). So, our 30kHz square wave carrier consists of sinewaves at 30kHz, 90kHz, 150kHz, 210kHz and so on.

Theoretically then, the DSSS signal consists of a 30kHz suppressed carrier with 27.92kHz and 32.08kHz lower and upper sidebands, a 90kHz suppressed carrier with 87.92kHz and 92.08kHz lower and upper sidebands, a 150kHz suppressed carrier with 147.92kHz and 152.08kHz lower and upper sidebands, and so on. Let’s examine these using the NI ELVIS III scope.

|  |  |
| --- | --- |
| 4. | Set the scope Timebase to 500uS/div. |

|  |  |
| --- | --- |
| 5. | Enable the FFT mode on the scope and set the frequency span from 0kHz to 250kHz. Select the FFT Source to be the DSSS signal channel. |

The display should now show more than four pairs of what appear to be significant sinewaves. This is deceptive as you’ll see.

|  |  |
| --- | --- |
| 6. | Activate the FFT display’s cursors. |

|  |  |
| --- | --- |
| 7. | Use an FFT display’s cursor to measure the frequency in the middle of each pair of the sinewaves.  **Note:** You’ll find that the signal consists of pairs of sidebands about a suppressed carrier at frequencies listed on the previous page.  You’ll also find that it consists of sidebands about suppressed carriers at other frequencies. However, although these signals are present, the display is a little misleading because the vertical axis is logarithmic (i.e. non-linear). |

|  |  |
| --- | --- |
| 8. | Change the FFT’s vertical display units from *dB* to *Linear*.  **Note:** This display shows you the linear relationship between the sinewaves’ amplitude. |

|  |  |
| --- | --- |
| 9. | Use the FFT display cursor to measure the frequency of these significant sinewaves.  **Note:** The frequencies should be identical to those listed on the previous page. |

|  |  |
| --- | --- |
| 10. | Return the FFT’s vertical display units to the *dB* position. |

|  |  |
| --- | --- |
| 11. | Disconnect the patch lead from the function generator’s output and return it to the Sequence Generator module’s bipolar analog *X* output.  **Note:** This returns the set-up to that shown in Figures 1 and 2 with a PN Sequence for the carrier instead of a square wave. |

|  |  |
| --- | --- |
| 12. | Examine the spectral composition of the original DSSS signal with the FFT’s vertical display units set to both *dB* and *Linear*. |

|  |  |
| --- | --- |
| 13. | Capture a screenshot of the FFT and append to your report. Annotate your report appropriately so as to identify the waveforms captured. Use the cursors to highlight important levels and transition points in the waveform if necessary. |

1-3 Why is the spectral composition of the DSSS signal much more complex when the carrier is a PN Sequence instead of a square wave?

|  |
| --- |
|  |
|  |
|  |

## 1.4 Implement: Using the product detector to recover the message

|  |  |
| --- | --- |
| 1. | Locate the RC LPF module on the board. Note that this filter is simply a first order RC stage. |
| 2. | Modify the set-up as shown in Figure 5.  **Note:** Pay close attention that the leads connect to the Multiplier 2 module’s *AC* inputs and not its *DC* inputs. |

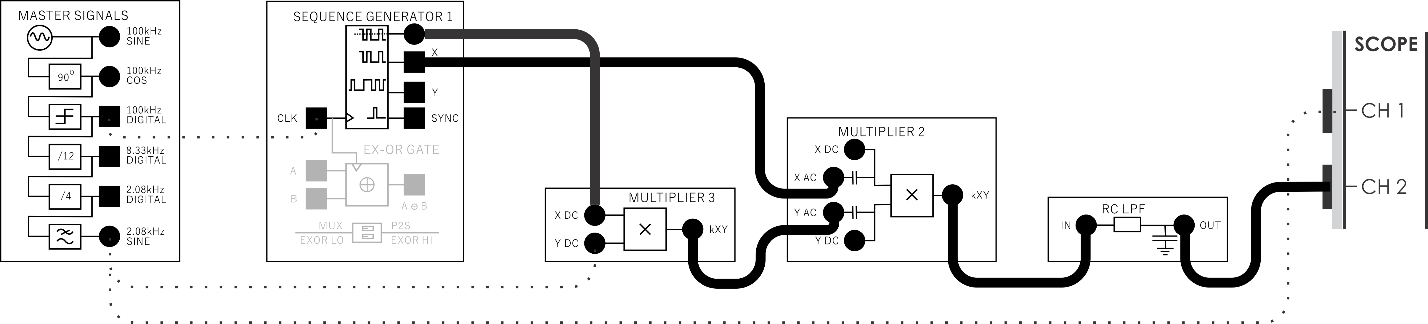


Figure 5: Patching for product demodulation of DSSS

The additions to the set-up in Figure 5 can be represented by the block diagram in Figure 6. The Multiplier module and the RC LPF module implement a product detector which recovers the original message from the DSSS signal. To facilitate this, the PN sequence used for the modulator’s carrier is “stolen” for the product detector’s local carrier (though it’s stolen from the module’s *X* output but the bit pattern is the same).

You can also replace the RC LPF with a higher order filter i.e.: Tuneable LPF or RRC LPF. Try this and document the difference it makes.

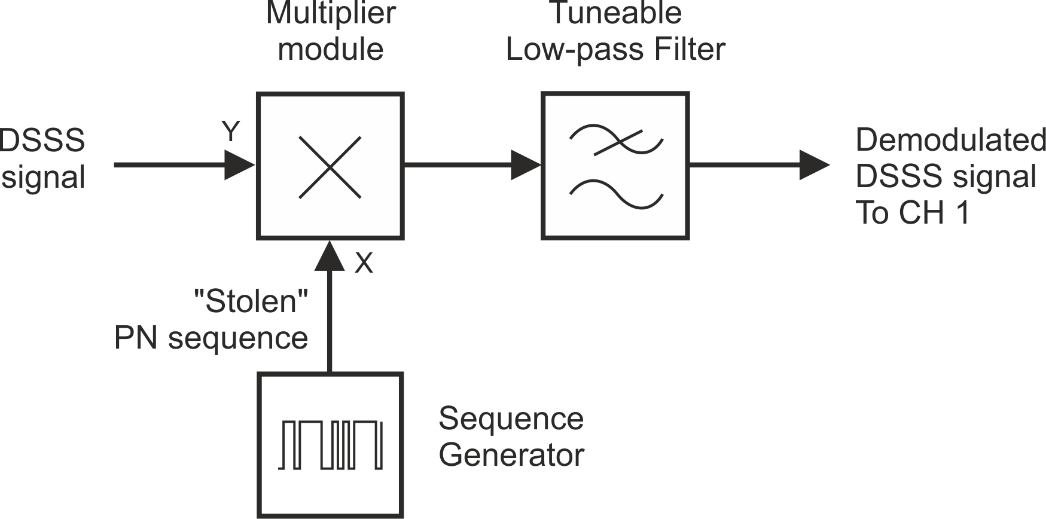


Figure 6: Block diagram for product demodulation

The entire set-up can be represented by the block diagram in Figure 7.

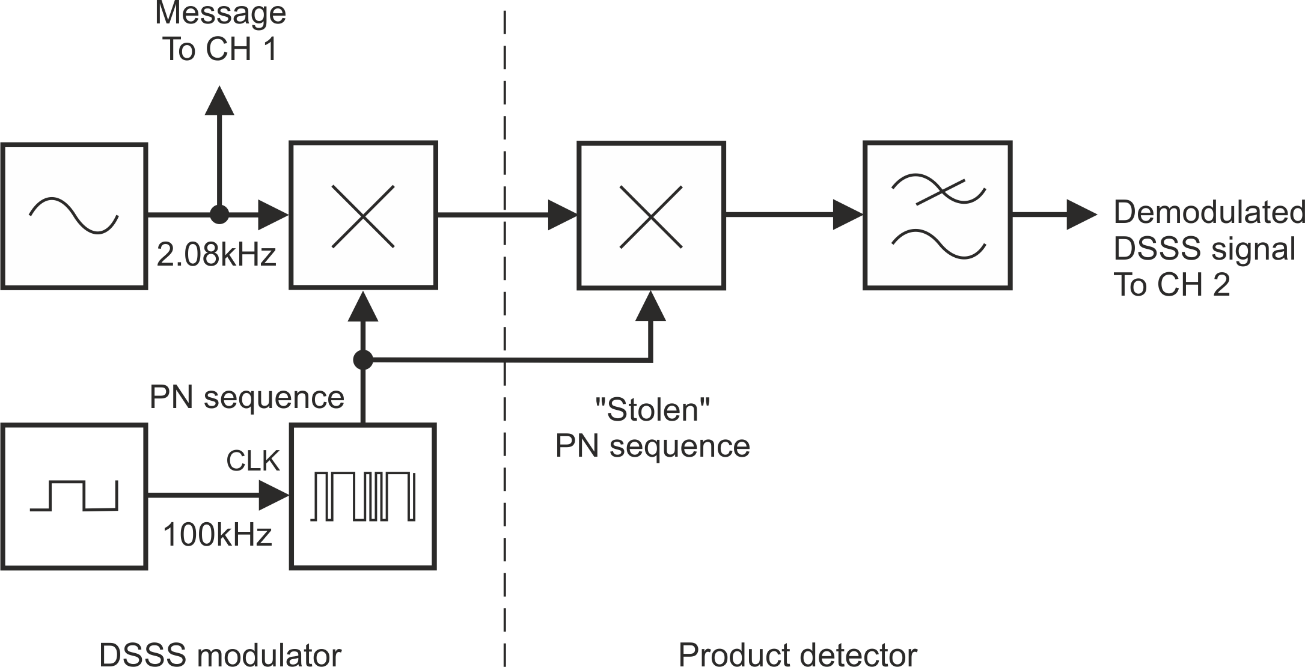


Figure 7: Complete block diagram for product demodulation

|  |  |
| --- | --- |
| 4. | Capture a screenshot of the scope display and append it to your report. Annotate your report appropriately so as to identify the waveforms captured. Use the cursors to highlight important levels and transition points in the waveform if necessary. |

Recall that the message can only be recovered by the product detector if an identical PN sequence to the DSSS modulator’s carrier is used. The next part of the experiment demonstrates this.

|  |  |
| --- | --- |
| 5. | Modify the set-up as shown in Figure 8 to make the demodulator’s local carrier a different PN sequence to the transmitter’s carrier. You will be using the Y sequence which is completely different from the X sequence. |

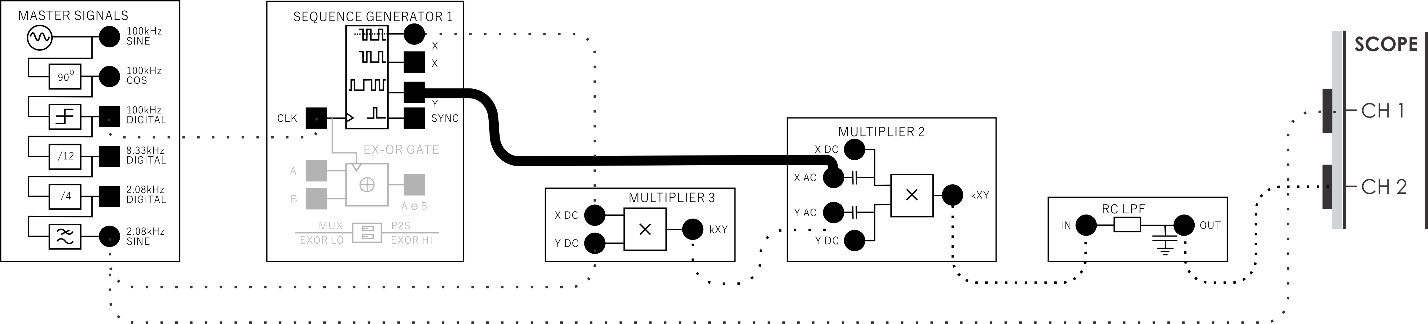


Figure 8: Demodulation with incorrect sequence

|  |  |
| --- | --- |
| 6. | Compare the message with the product detector’s new output. |

1-4 What does the signal out of the low-pass filter look like?

|  |
| --- |
|  |
|  |
|  |

1-5 Why does using the wrong PN sequence for the local carrier cause the product detector’s output to look like this?

|  |
| --- |
|  |
|  |
|  |

## Section 2: DSSS and deliberate interference (jamming)

Interference occurs when an unwanted electrical signal gets added to the transmitted signal (typically in the channel) and changes it enough to change the recovered message. Electrical noise is a significant source of unintentional interference.

However, sometimes noise is deliberately added to the transmitted signal for the purpose of interfering or “jamming” it. The next part of the experiment models deliberate interference to show how spread spectrum signals are highly resistant to it.

## 2.1 Implement: DSSS and deliberate interference (jamming)

|  |  |
| --- | --- |
| 1. | Move the patch lead from the Sequence Generator’s *Y* output back to its *X* output.  **Note:** The product detector should now be recovering the message again. |

|  |  |
| --- | --- |
| 2. | Adjust the Function Generator using its controls for an output with the following specifications: |

Table 2 Function Generator Configuration

|  |  |
| --- | --- |
| Waveform | Sine |
| Frequency | 50kHz |
| Amplitude | 2Vpk |
| DC Offset | 0V |

|  |  |
| --- | --- |
| 3. | Set the scope’s *Trigger Source* control to *CH 1*. |
| 4. | Locate the Adder module on the board and turn its *g* control fully anti-clockwise. |
| 5. | Set the Adder module’s *G* control to about the middle of its travel (unity gain). |
| 6. | Modify the set-up as shown in Figure 9. |

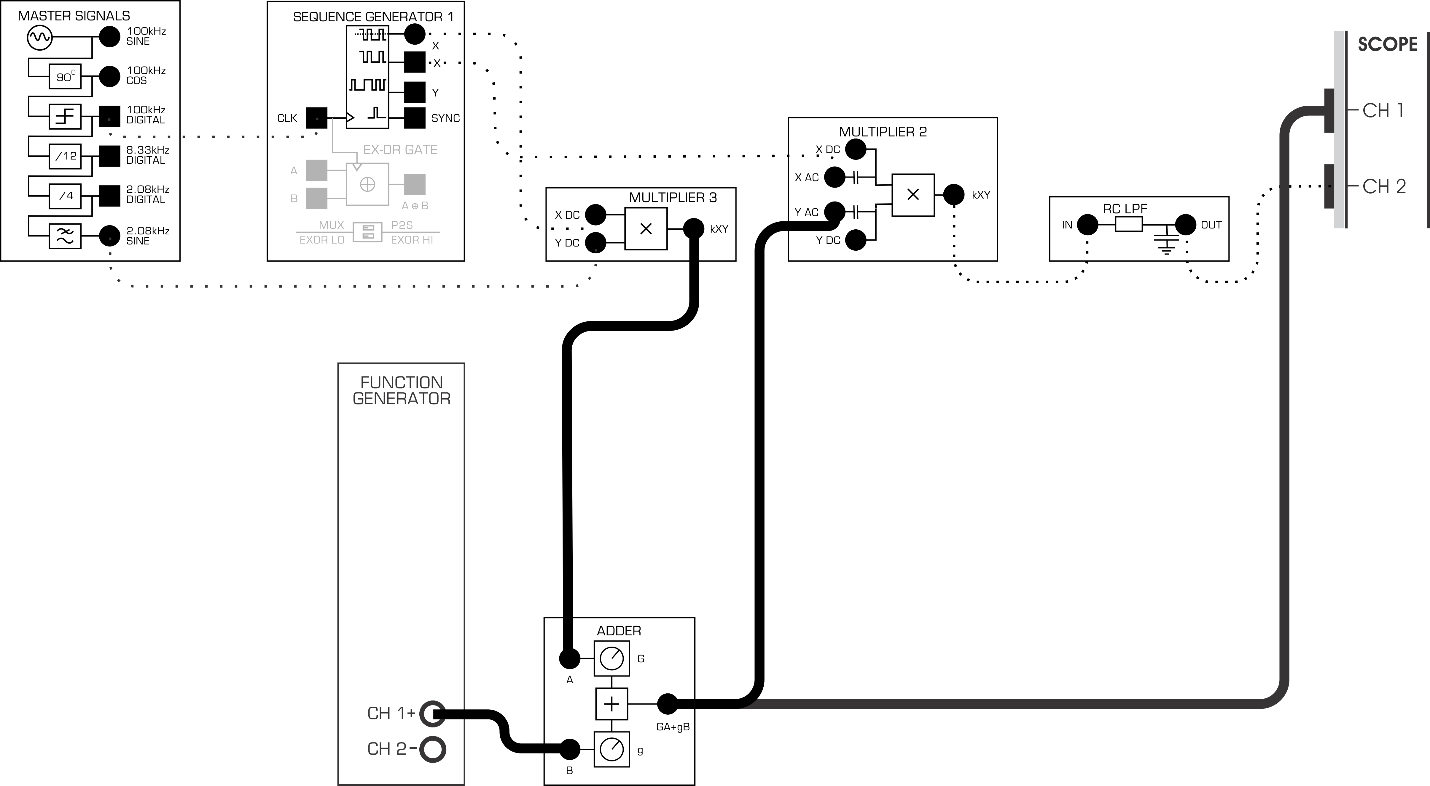


Figure 9: Patching for DSSS jamming with a single frequency

|  |  |
| --- | --- |
| 7. | Turn on the FFT window so you can view the spectrum of the DSSS plus jamming signal at the same as the time domain signals. |

The set-up in Figure 9 can be represented by the block diagram in Figure 10. The function generator is used to generate a variable frequency jamming signal that is added to the DSSS signal in the channel using the Adder module.

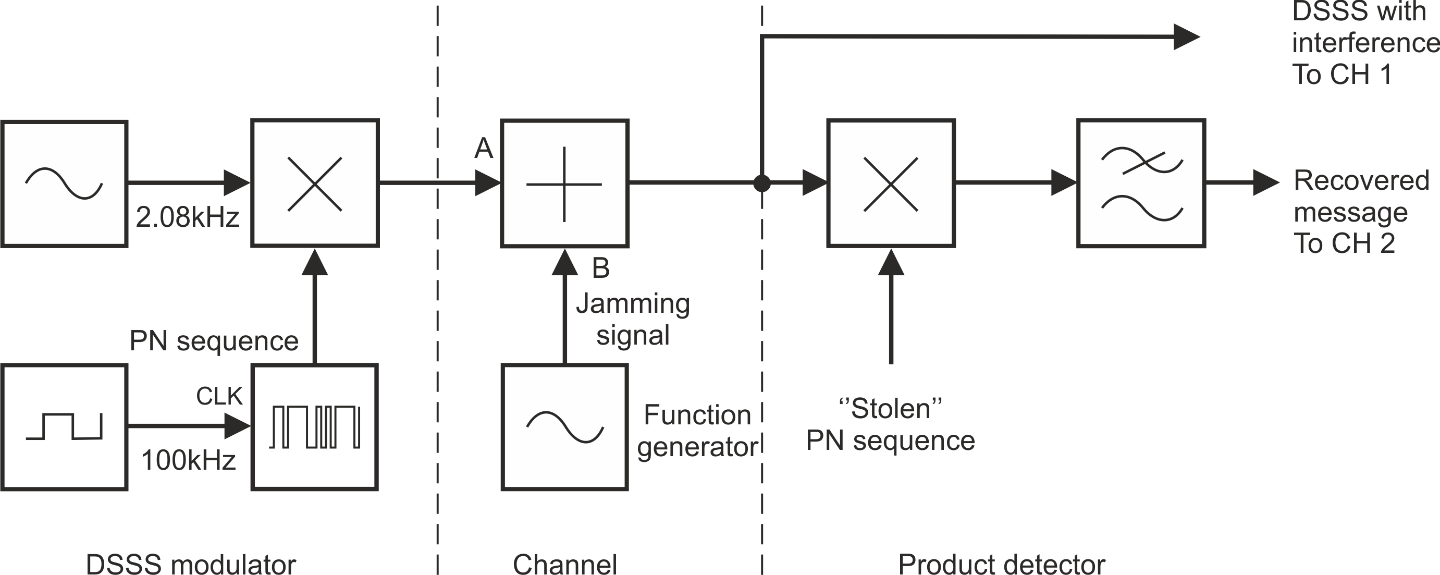


Figure 10: Block diagram for DSSS jamming with single frequency

|  |  |
| --- | --- |
| 8. | Add the jamming signal to the DSSS signal by slowly turning the Adder module’s *g* control clockwise. Stop when it’s at about half its travel. |

|  |  |
| --- | --- |
| 9. | As you increase the amplitude of the jamming signal note the effect it has on the DSSS signal and the recovered message. |

|  |  |
| --- | --- |
| 10. | Vary the jamming signal’s frequency by varying the Function Generator’s output frequency. Watch the effect of doing so on the FFT display. |

|  |  |
| --- | --- |
| 11. | Note the effect this has on the DSSS signal and on the recovered message. |
| 12. | Increase the size of the jamming signal to maximum by turning the Adder module’s *g* control fully clockwise. |

|  |  |
| --- | --- |
| 13. | Note the effect this has on the DSSS signal and on the recovered message. |

2-1 Why doesn’t the jamming signal interfere with the recovery of the message?

|  |
| --- |
|  |
|  |
|  |

A more sophisticated approach to jamming involves automatically sweeping the jamming signal through a wide range of frequencies to increase the chances of upsetting the transmitted signal. The next part of the experiment lets you see how spread spectrum handles this.

|  |  |
| --- | --- |
| 14. | Return the Adder module’s *g* control to about the middle of its travel. |
| 15. | Swap out the Function generator for a VCO module. Vary the GAIN control to control the output bandwidth. Vary the FREQ control to control the center frequency. | |
| 16. | Modify the set-up as shown in Figure 11. | |

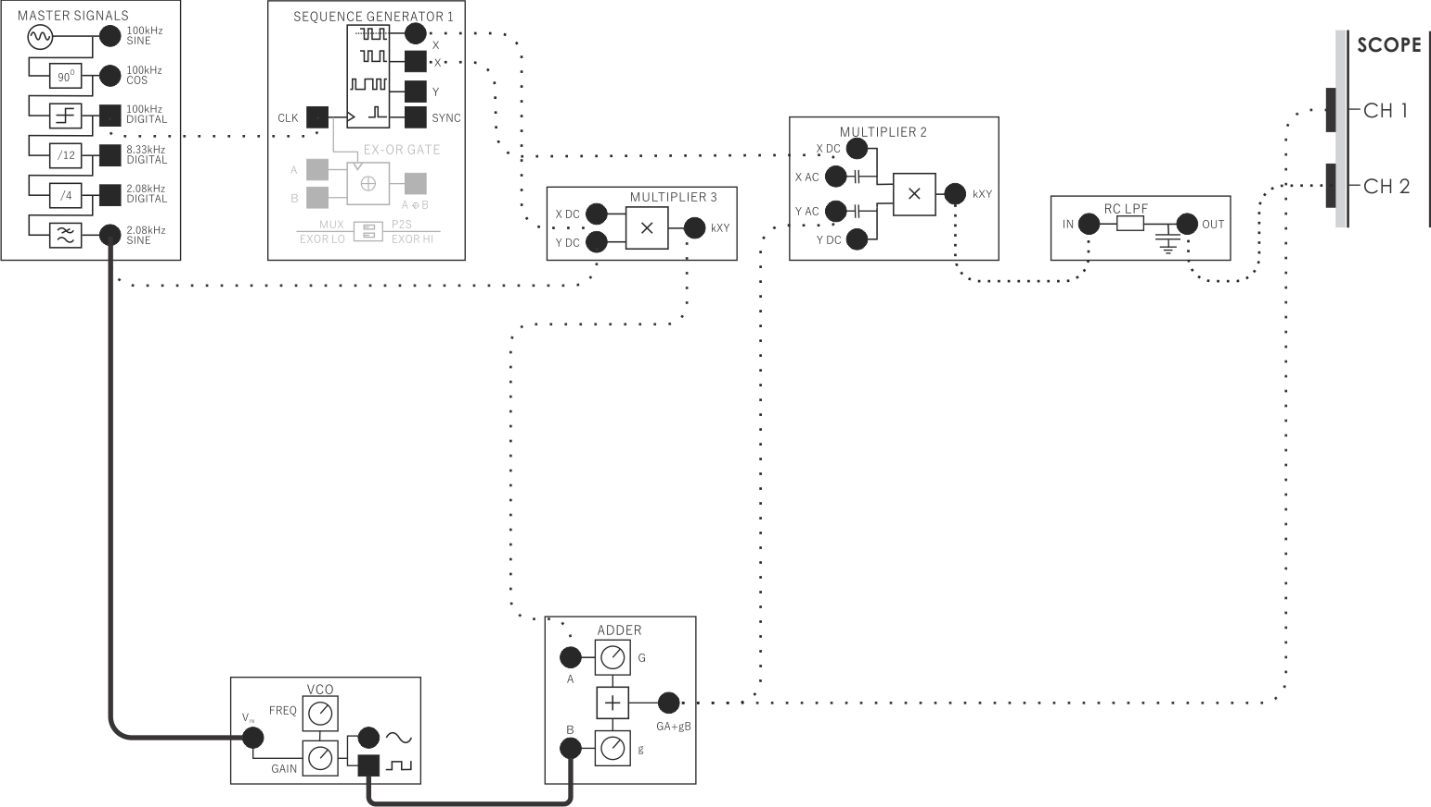


Figure 11: Patching for DSSS with FM jamming signal

This modification forces the jamming signals source’s output to sweep continuously through a wide range of frequencies.

|  |  |
| --- | --- |
| 17. | Note the effect this has on the DSSS signal and on the recovered message. |
| 18. | Increase the size of the jamming signal to maximum by turning the Adder module’s *g* control fully clockwise. |
| 19. | Note the effect this has on the DSSS signal and on the recovered message. |

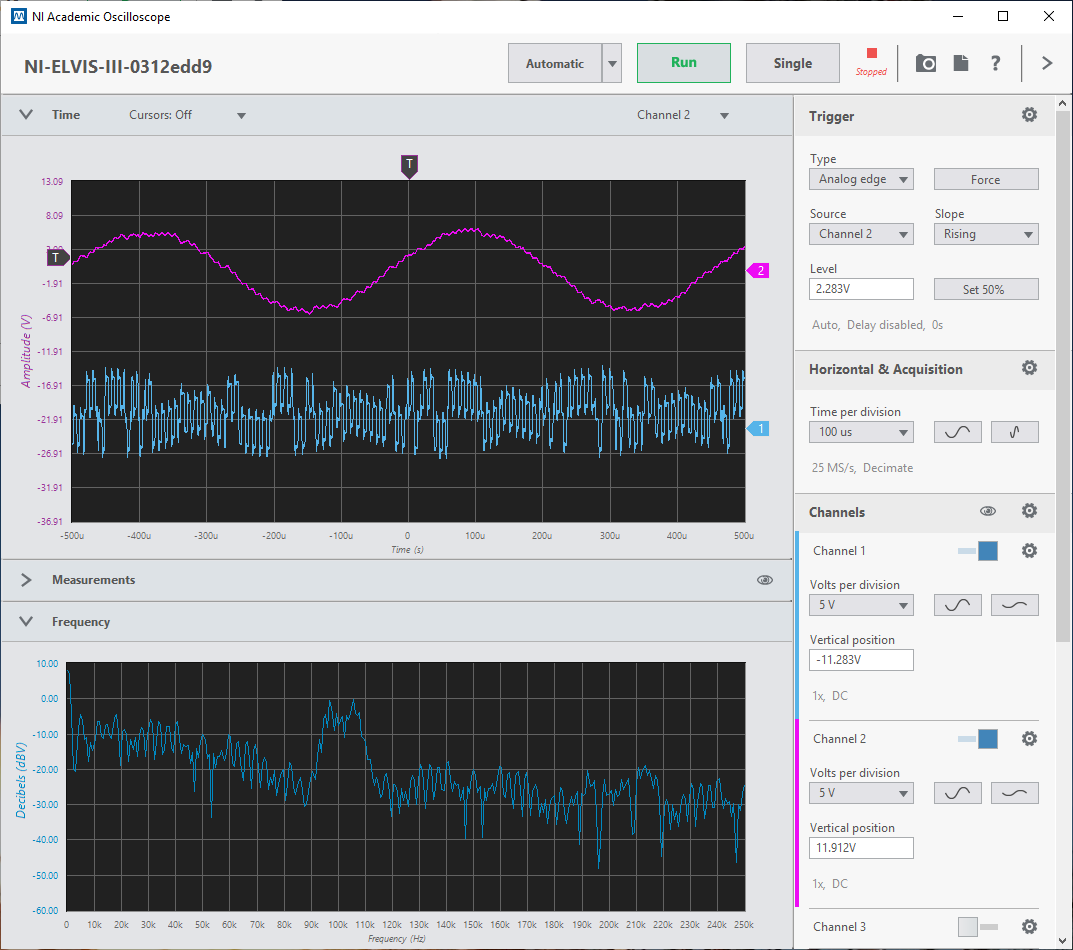


Figure 12: Spectrum of DSSS signal with FM jamming signal

2-2 Why doesn’t the sweeping jamming signal interfere with the recovery of the message?

|  |
| --- |
|  |
|  |
|  |

An even more sophisticated approach to jamming involves using many jamming signals at once (broadband jamming) to increase the chances of upsetting the transmitted signal. The next part of the experiment lets you see how spread spectrum handles this.

|  |  |
| --- | --- |
| 20. | Return the Adder module’s soft *g* control to about the middle of its travel. |

|  |  |
| --- | --- |
| 21. | Disconnect the lead to the VCO and modify the set-up as shown in Figure 13. |

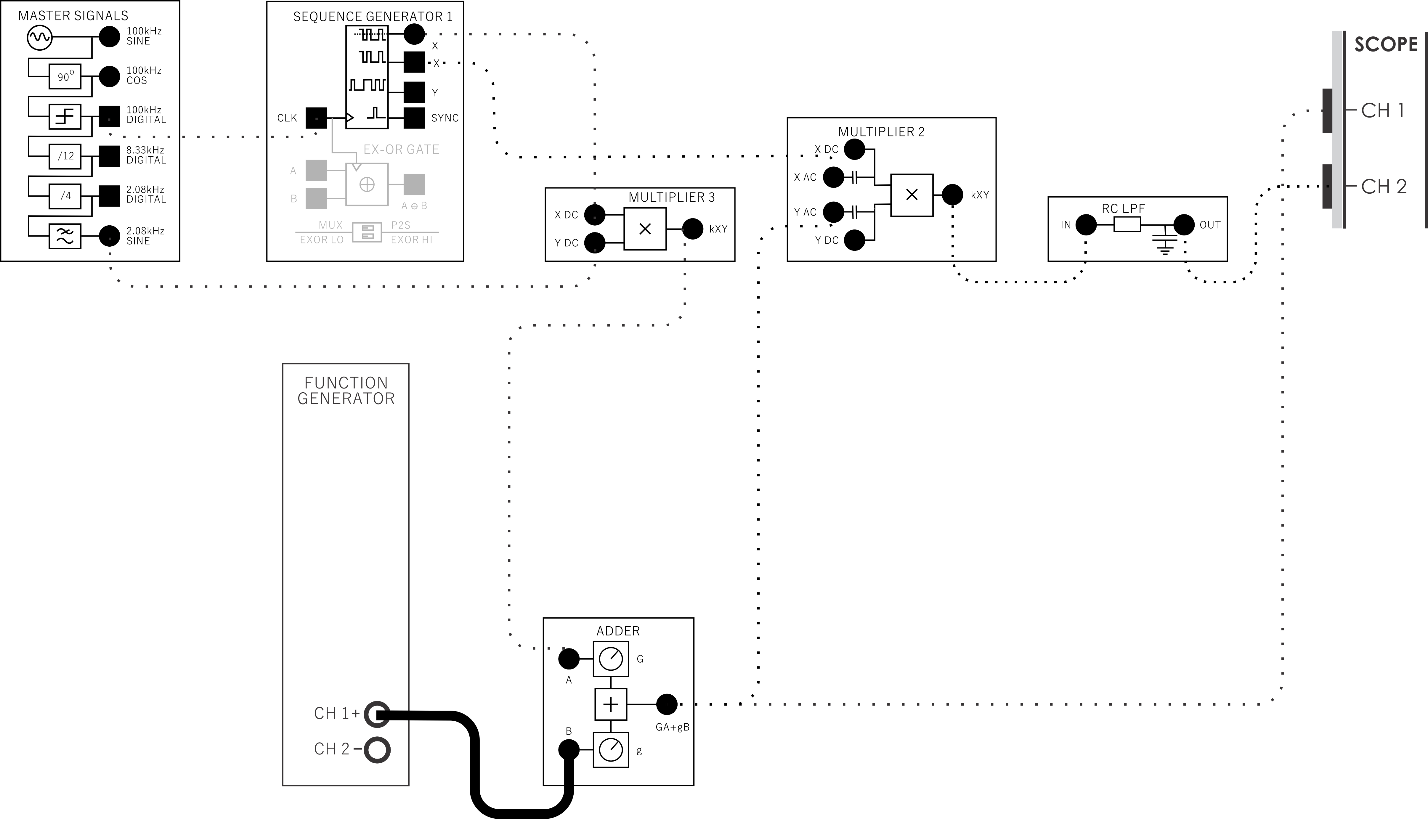


Figure 13: Patching for DSSS jamming with noise signal

This modification uses the Noise Generator module to model a jamming signal that consists of thousands of frequencies.

|  |  |
| --- | --- |
| 22. | Select the Custom waveform on the Function Generator for Channel 1 and load the noise waveform file “ECB\_120k noise.csv”. Set the “Update rate” to 500kS/s. |
| 23. | Note the effect this has on the DSSS signal and on the recovered message. |
| 24. | Increase the strength of the broadband jamming signal by increasing the gain at the ADDER module. |
| 25. | Note the effect this has on the DSSS signal and on the recovered message. |
| 26. | Increase the strength of the broadband jamming signal even more by introducing the Amplifier module into the noise path. Do so by plugging the Function Generator output to the Amplifier module input and plugging the Amplifier module output into the Adder module *B* input. |
| 27. | Turn the GAIN control on the Amplifier module to modify the noise level. |
| 28. | Note the effect this has on the DSSS signal and on the recovered message. |

2-3 Why doesn’t this broadband jamming signal interfere with the recovery of the message?

|  |
| --- |
|  |
|  |
|  |

## Section 3: Expansion activities

If the instructor allows, listen to how DSSS performs when being jammed by following these instructions. You’ll need a set of headphones or earbuds for this activity.

1. Remove the jamming signal by disconnecting the Adder module’s *B* input from the Amplifier module output.
2. Disconnect the Function Generator output from the Amplifier module input.
3. Connect the RC LPF Filter module’s output to the Amplifier module input.
4. Turn the *GAIN* control on the Amplifier module fully anti-clockwise.
5. Without wearing the headphones, plug them into the Amplifier module headphone socket.
6. Put the headphones on.
7. Adjust the Amplifier module’s *GAIN* control until the 2.08kHz tone is a comfortable sound level.
8. Investigate what happens when the wrong PN sequence is used to demodulate the DSSS signal (like you did in section 1.4, step 5) by moving the patch lead from the Sequence Generator’s *X* output to its *Y* output.
9. Return the patch lead from the Sequence Generator’s *Y* output back to its *X* output.
10. Investigate what happens when a single sinewave is used to jam the DSSS signal (like you did in section 2.1) by connecting the Function Generator’s output to the Adder module’s *B* input.
11. Investigate what happens when a broad-band signal is used to jam the DSSS signal (like you did in section 2.1) by connecting the Function Generator with custom noise loaded to the Adder module’s *B* input.