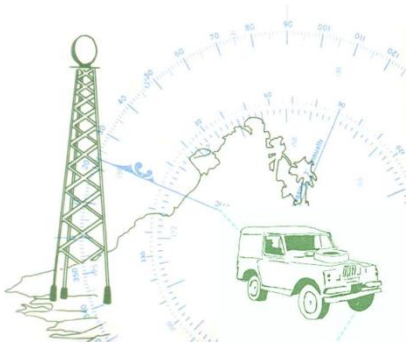
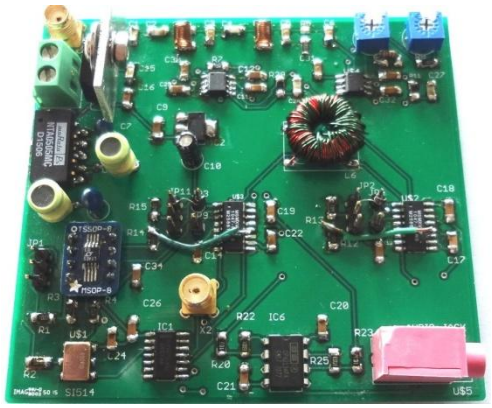


SDR_Ursinho

Design, Simulation and Assembly of a Direct Conversion High Frequency SDR Software Defined Receiver



Jeremy Clark VE3PKC

Copyright Information

Clark Telecommunications
Jeremy Clark
500 Duplex Suite 506
Toronto M4R-1V6, Ontario, Canada
416-488-5382
jclark@clarktelecommunications.com
www.clarktelecommunications.com

Table of Contents

1 - Introduction

1.1 - Design Philosophy

During the summer of 2015, I completed an eBook about HF High Frequency Radio (Ref.1). The book emphasized various aspects of HF such as digital communications, receiver structures, synthesizers and propagation prediction for voice and data links in terms of SNR/BER. As part of the book, I wanted to include a design for a simple direct conversion hardware/software radio that embodied the various subjects that I had discussed. SDR_Ursinho is the result of that effort.

This eBook discusses the design, simulation, assembly and testing of a Zero IF Direct Conversion HF receiver using compact SMD surface mount devices that can be easily soldered by radio amateurs. The design uses new components such as the MSMXVHF Mixer/Switched Capacitor Filter by Mixed Signal Integration (Ref.7), the Si514 I2C Programmable XO by Silicon Labs (Ref.10), the LTC6904 I2C Programmable Oscillator from Linear (Ref.8), the AD603 Variable Gain RF Amplifier from Analog Devices (Ref.6) and Wurth CAIR miniature air core inductors for a compact BPF (Ref.5). Sideband selection and signal decoding is done by DSP routines in a PC soundcard. Frequency selection and LPF control is done via I2C over a Raspberry Pi2B GPIO bus using Python routines.

The receiver name derives from the following: SDR_Ursinho = Software Defined Radio _Ursinho (Brazilian Portuguese for Small BEaR). This receiver is meant for educational purposes as opposed to a radio with the latest cutting edge specifications. It is ideal for a College or University Telecommunications Lab Course. A full parts list is given in App.G.

1.2 - Block & Level Diagram

Figure 1.1 shows the Block & Level diagram for the SDR_Ursinho. The receiver is a simple 25m band design, for strong South American signals in the 11-13MHz range. Input signals are first band pass filtered. The filter is made with small SMD components. The signal is then amplified by a cascade of two programmable AD603 amplifiers. Each stage has a adjustable gain from 0 - 40dB which is set by a DC voltage Vg. The amplified signal is then fed to an Amidon T50-2 toroidal balun.

An Si514 is used as the VFO and runs at 4 x Fc. This drives the quad circuit consisting of two flip flops. The quad circuit produces 2 square waves at Fc separated by 90deg. These form the I cosine and Q sine signals used to drive the two MSMXVHF mixers. Each mixer contains a programmable LPF low pass filter. The switched capacitor LPFs corner frequencies are controlled by an LTC6904. Both the LTC6904 and Si514 are controlled over the I2C pins of a Raspberry Pi2B GPIO bus. Python code is used to program both the LTC6904 and Si514 and is listed in App.E & App.F.

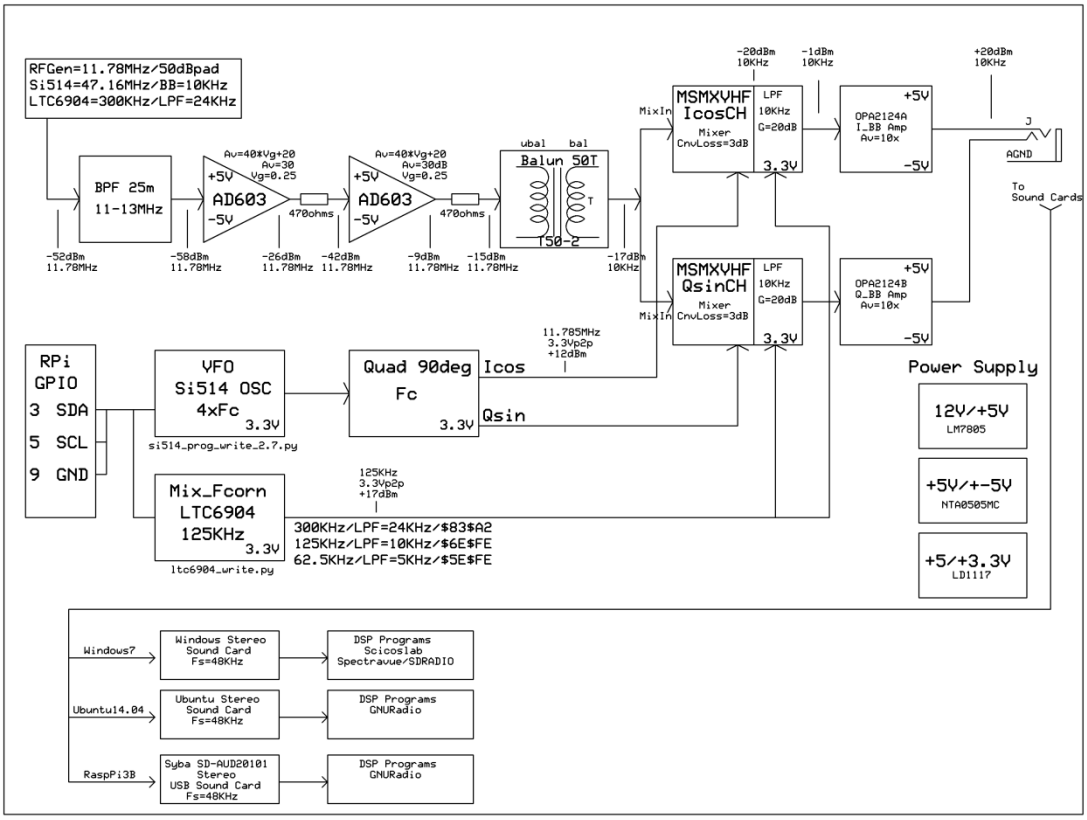


Figure 1.1 - SDR_Ursinho Block & Level Diagram

1.3 - Development Environment

App.A provides details of the development environment used for the receiver. All modules were first prototyped on a CSC Protoboard 103. Once they were understood and worked correctly, a PCB was designed, all the circuits were transferred over and soldered. A two sided board with flooded RF ground was used. 1206 SMD components were used so that they could be hand soldered. Instrumentation was simple and low cost including a homemade 25MHz RF generator.

1.4 - Hartley Direct Conversion Simulation

The SDR_Ursinho uses a so called Hartley Direct Conversion structure. The receiver is modeled in Scicos (Ref.2) and shown in Figure 1.2. Input signals are multiplied by I cosine and Q sine generators, summed and the I branch phase delayed by 90deg. In the literature a +90deg phase shift is normally shown in the Q branch, but this is equivalent to a -90deg or delay in the I branch. USB selection is accomplished by setting the amplitude of the sin(wosct) generator to -1 and LSB selection by setting the amplitude to +1. App.B covers the I & Q demodulation mathematics.

Figure 1.3 shows the LPF output of [0.5VDC + cos(2KHz)t] for a USB input signal of 11782KHz. Figure 1.4 shows the LPF output of 0.5VDC only, for an LSB input signal of 11778KHz, illustrating the rejection of the LSB. Ref.3 covers in detail how to use ScicosLab/Scicos/Modnum for telecommunications simulation.

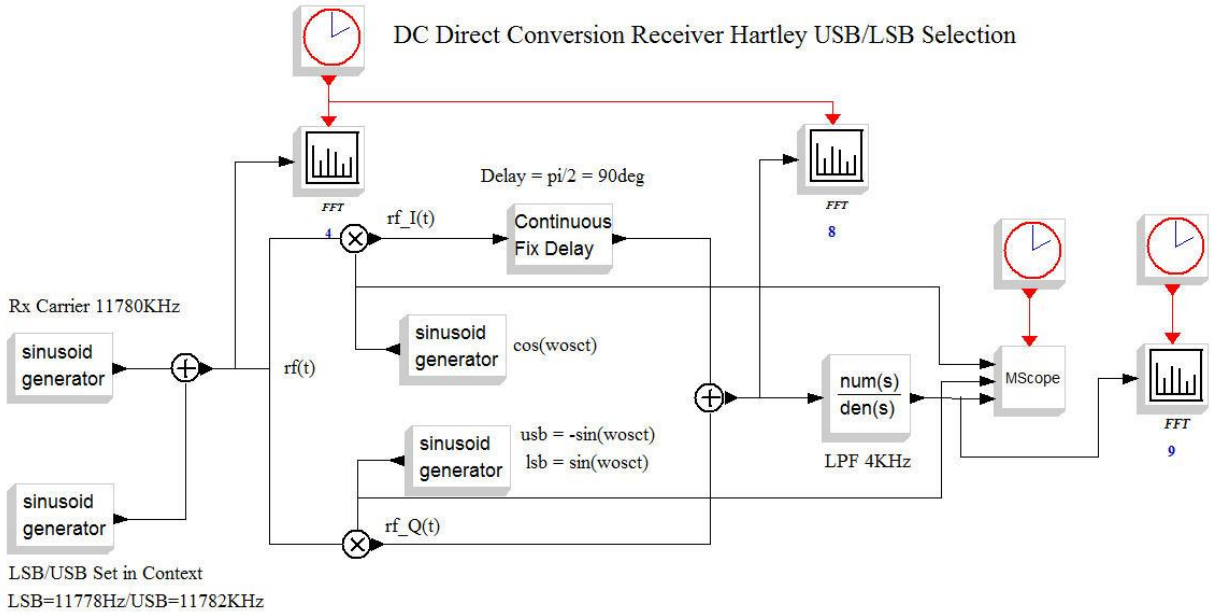


Figure 1.2 - Hartley Direct Conversion Receiver USB/LSB Selection

direct_harley.cos

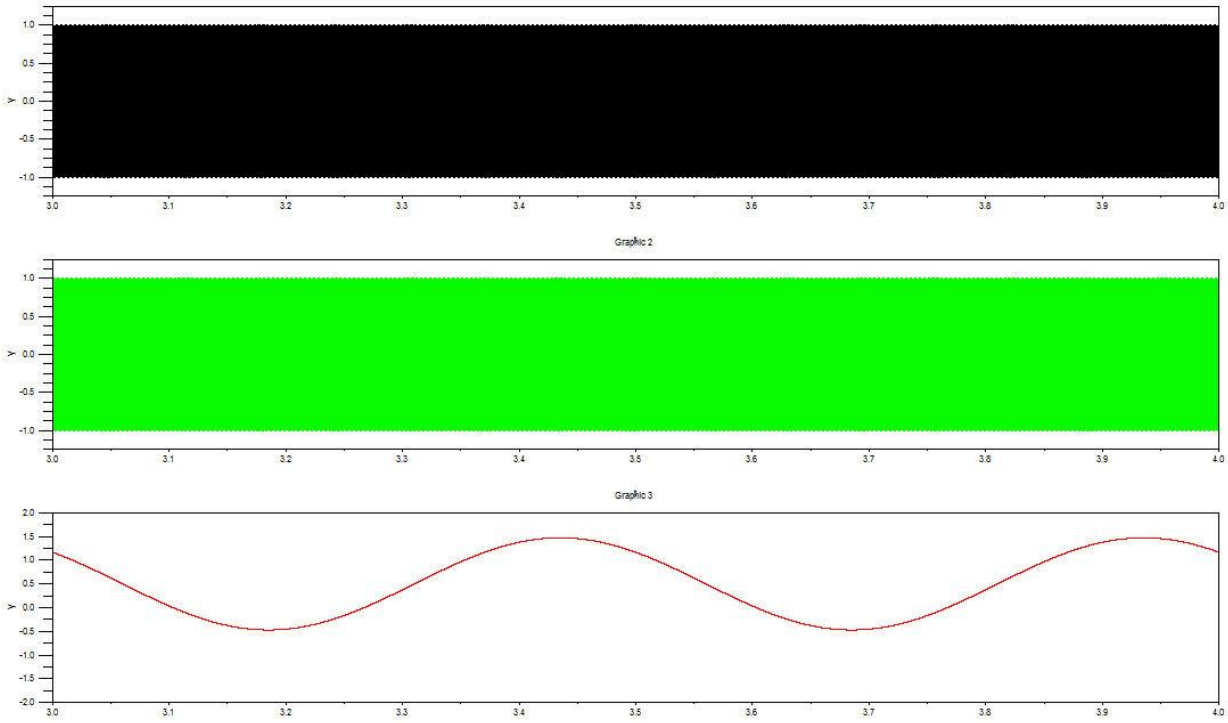


Figure 1.3 - Hartley Icos, Qsin and LPF_usb = 0.5 + cos2KHz

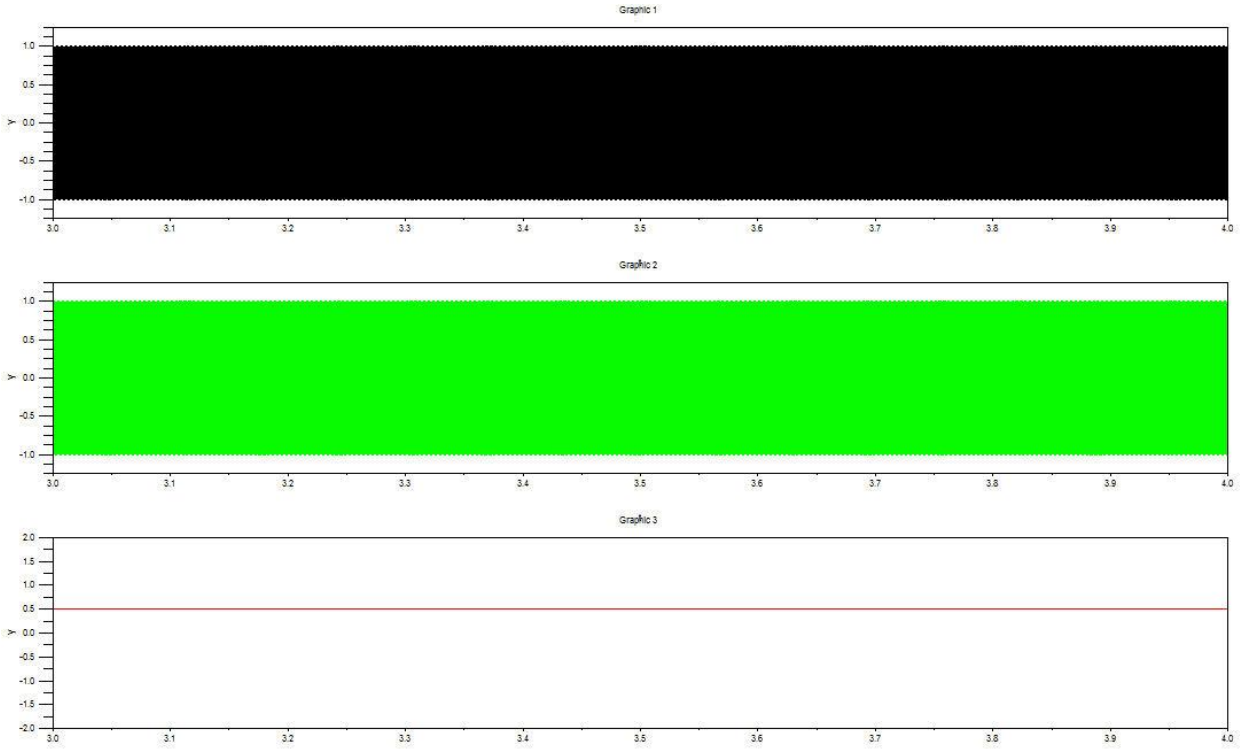


Figure 1.4 - Hartley Icos, Qsin and LPF_lsb = 0.5

2 - BPF Band Pass Filter Design

2.1 - BPF ELSIE Design

The 25m band is ideal for strong signals, so I designed the input BPF for 11 - 13MHz. I used the ELSIE program by J. Tonne W4ENE (Ref.4). The BPF purpose is to restrict input noise and interference as well as any harmonic leakage from the various oscillators (square waves have 3rd/5th/7th.. harmonics). Figure 2.1 shows a 3rd order design for a center frequency of 12MHz and BW=2MHz. Parts are to the closest 5%. Input and output terminations are 50ohms unbalanced. The filter uses 3 x 110nH inductors with 2 x 1500pF, 2 x 180pF and 1 x 1200pf capacitors.

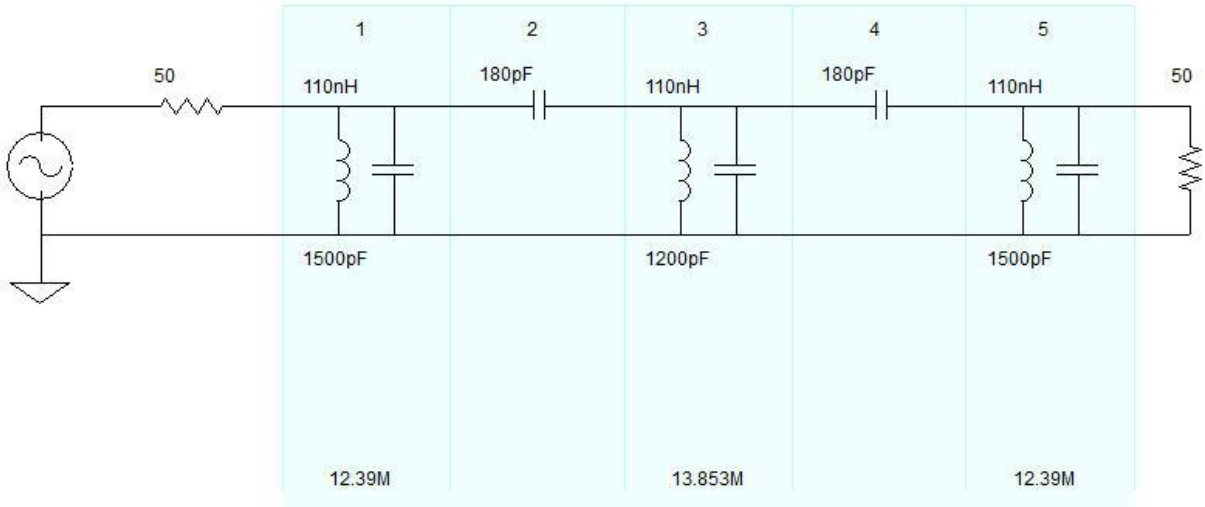


Figure 2.1 BPF Design Elsie Filter Program Fc=12MHz, BW=2MHz, n=3 bpf25m.lct

Figure 2.2 shows the transmission response with insertion loss of about 1.5dB at 12MHz and rapid fall off either side. Response at 10MHz = -24dB and at 14MHz = -16dB. Note that the group delay is not flat, so this would be a problem for wideband signals, but should not be a problem for signals of 3KHz bandwidth. Figure 2.3 shows the filter input impedance with a nominal 50ohms at 11.9MHz.

I tried unsuccessfully to build the filter with small breadboards and wire wound power filter inductors I had at home, but the Qs were too low. Finally I obtained miniature air core inductors from Wurth (Ref.5) and got the filter to work on the receiver PCB, using a flooded RF ground.

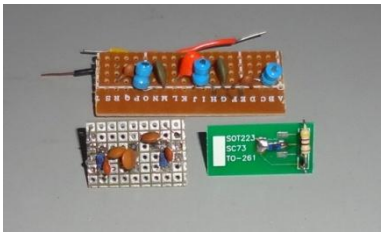


Figure 2.2 First Filter Prototypes

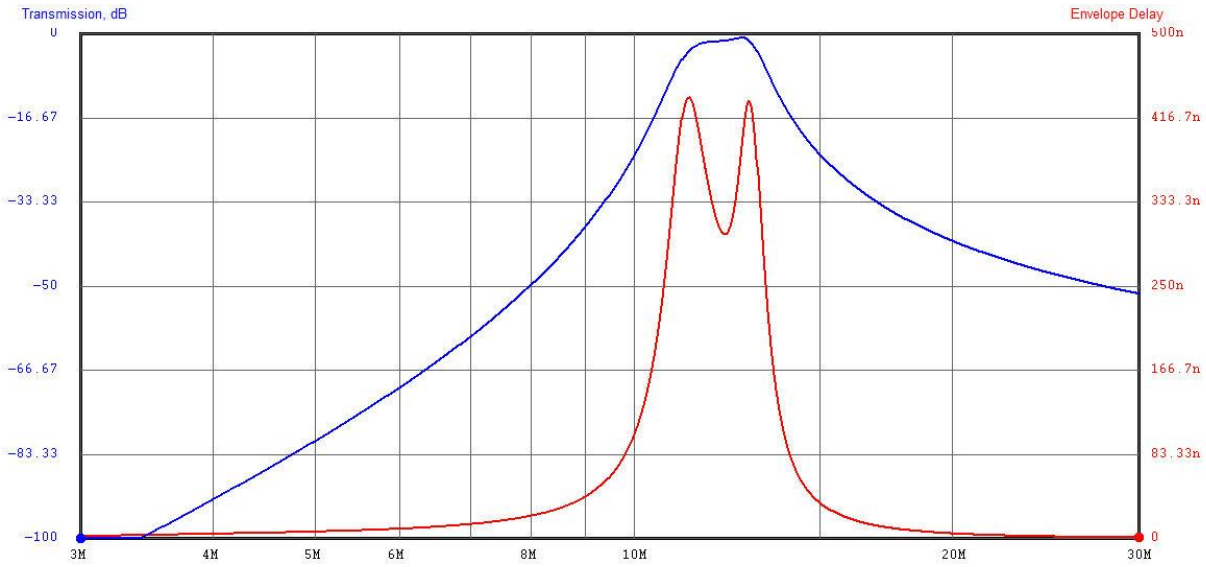


Figure 2.2 BPF Transmission & Group Delay Responses

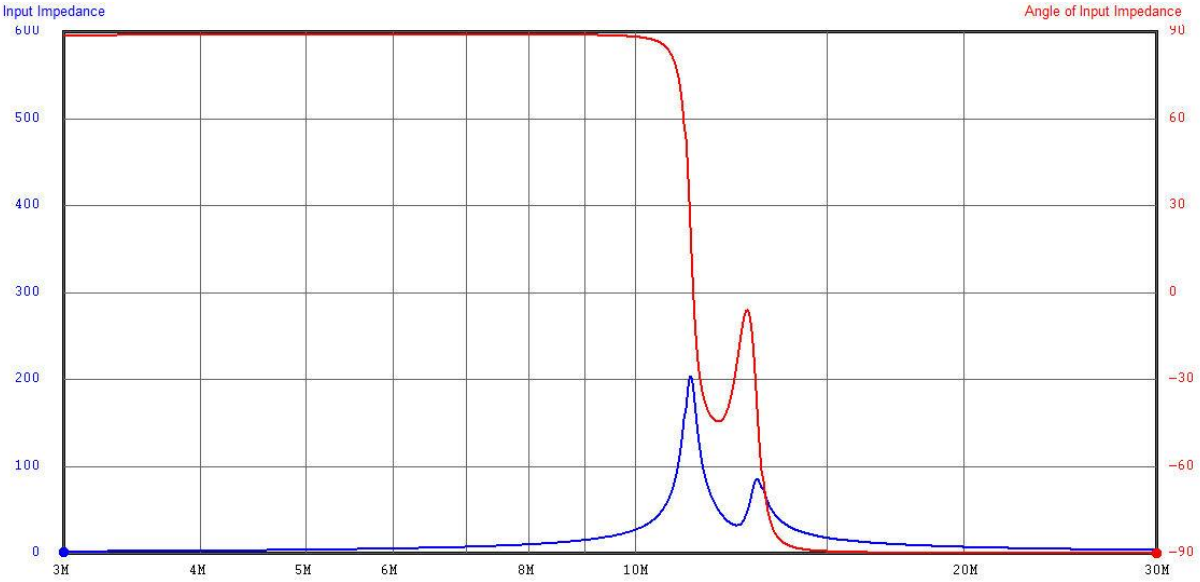


Figure 2.3 BPF Input Impedance Magnitude & Angle

2.2 - BPF LTspice Simulation

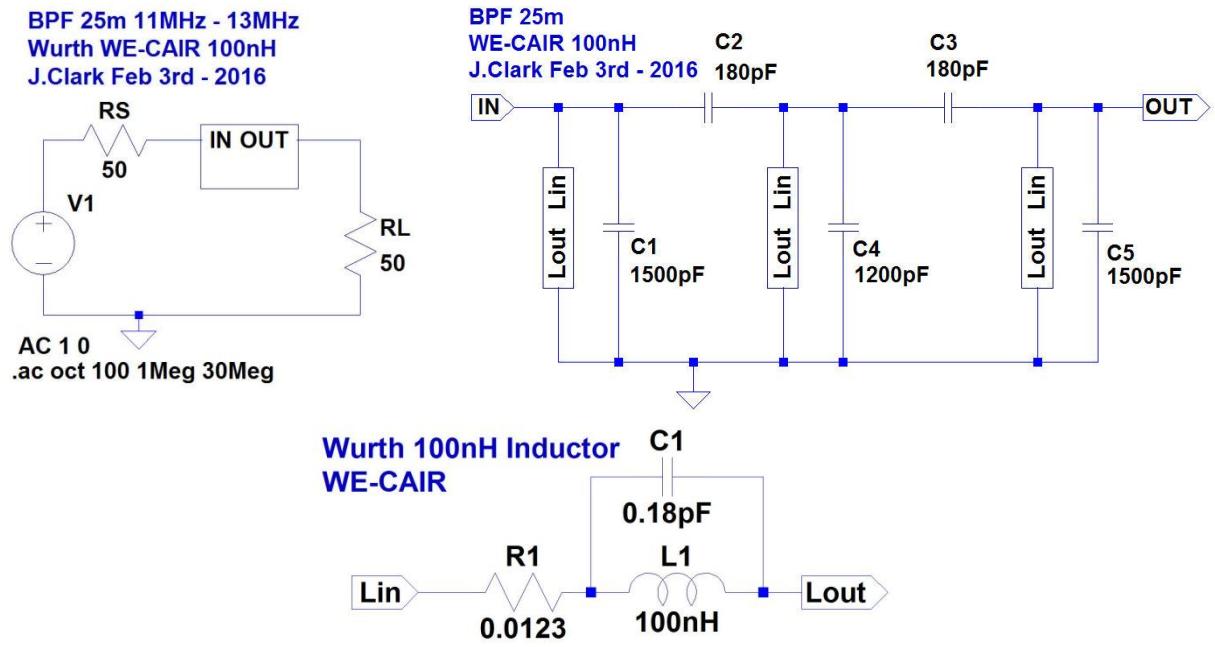


Figure 2.4 LTspice BPF Test Fixture and Filter Model bpf_25m_wurth_test.asc

The original filter design calls for 110nH inductors, but the closest I could find in an SMD form are 100nH from Wurth (Ref.5). Figure 2.4 shows the LTspice simulation for the filter which takes into account the 100nH inductance as well as the practical Q. The insertion loss at 12MHz = -7.5dB. Response at 10MHz = -37dB and at 14MHz = -13.5dB. So the real filter has more loss and is not as tight as the design with perfect inductors.

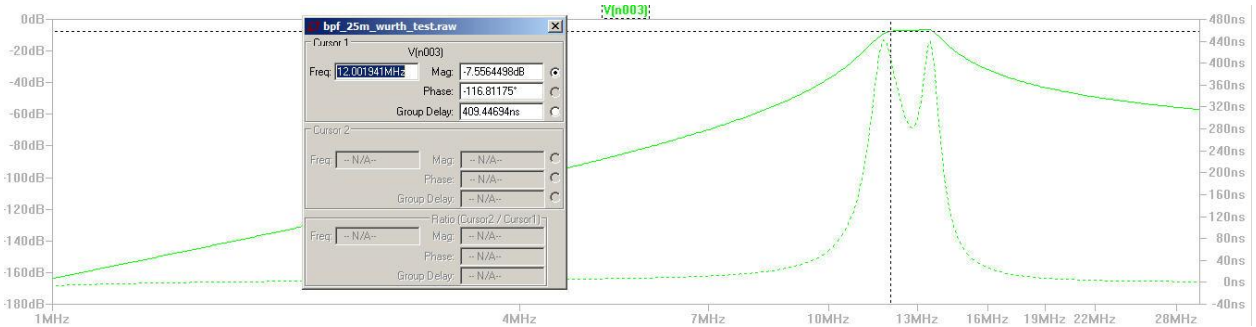


Figure 2.5 LTspice BPF Filter Response and Group Delay

2.3 - BPF Hardware Measurement

Figure 2.6 shows the actual BPF implementation on the PCB. Table 2.1 shows the measured frequency response and Table 2.2 shows the graph of this response. Note that the response matches the LTspice simulation.

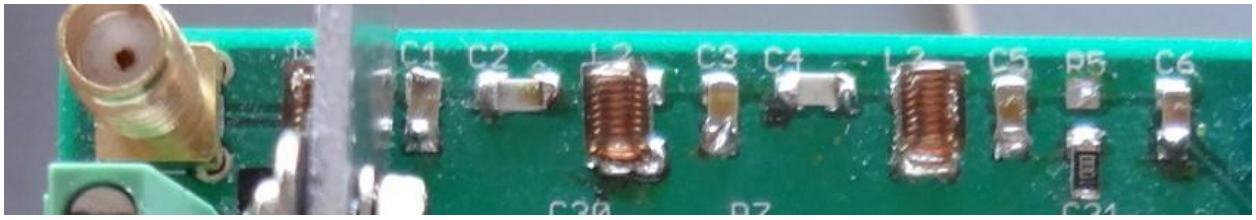


Figure 2.6 BPF Implemented on PCB Wurth 100nH Inductors

Freq MHz	BPF_In dBm	BPF_Out dBm	Insertion Loss dB
2	-13.2	<<-45	-57.5
4	-10.0	<<-45	-57.5
8	+3.0	<<-45	-57.5
10	+9.8	-35.0	-47.5
11	+12.2	-8.0	-20.5
11.5	+13.2	-0.7	-13.2
11.78	+13.6	+3.0	-9.5
12	+13.2	+5.0	-7.5
16	+7.0	-12.0	-24.5
20	+1.3	-26.5	-39
30	-6.2	-41.0	-53.5
Si514 EVBU Generator into 50ohm termination = 12.5dBm			

Table 2.1 BPF Filter Measurements

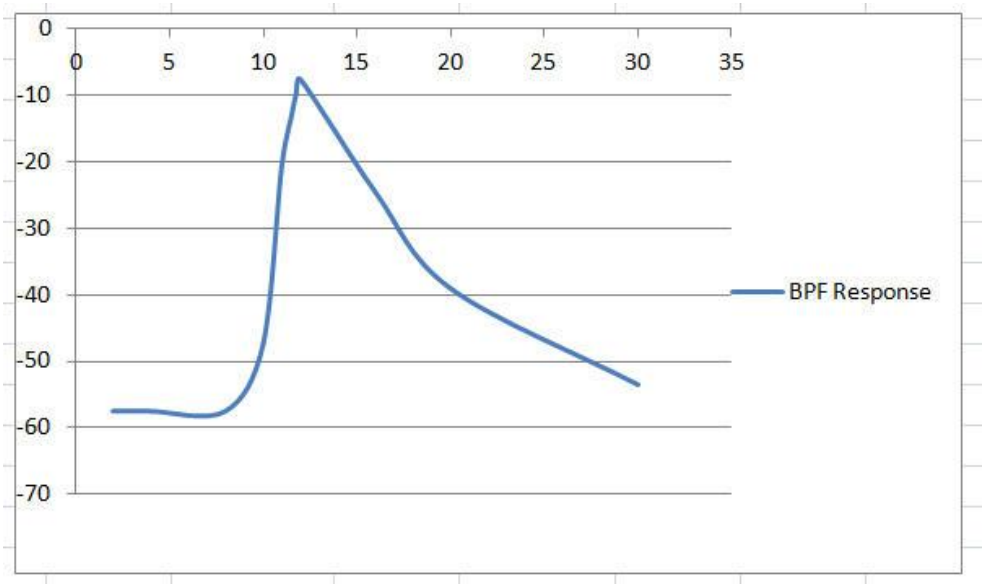


Table 2.2 BPF Filter Response