802.15.4 Project

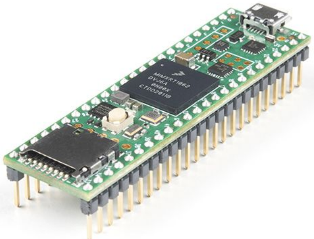
The Northernites

Sam Weismuller, SangRyul “Eric” Pae, Jay Cordaro

MAS 2023 Students UC San Diego

Introduction

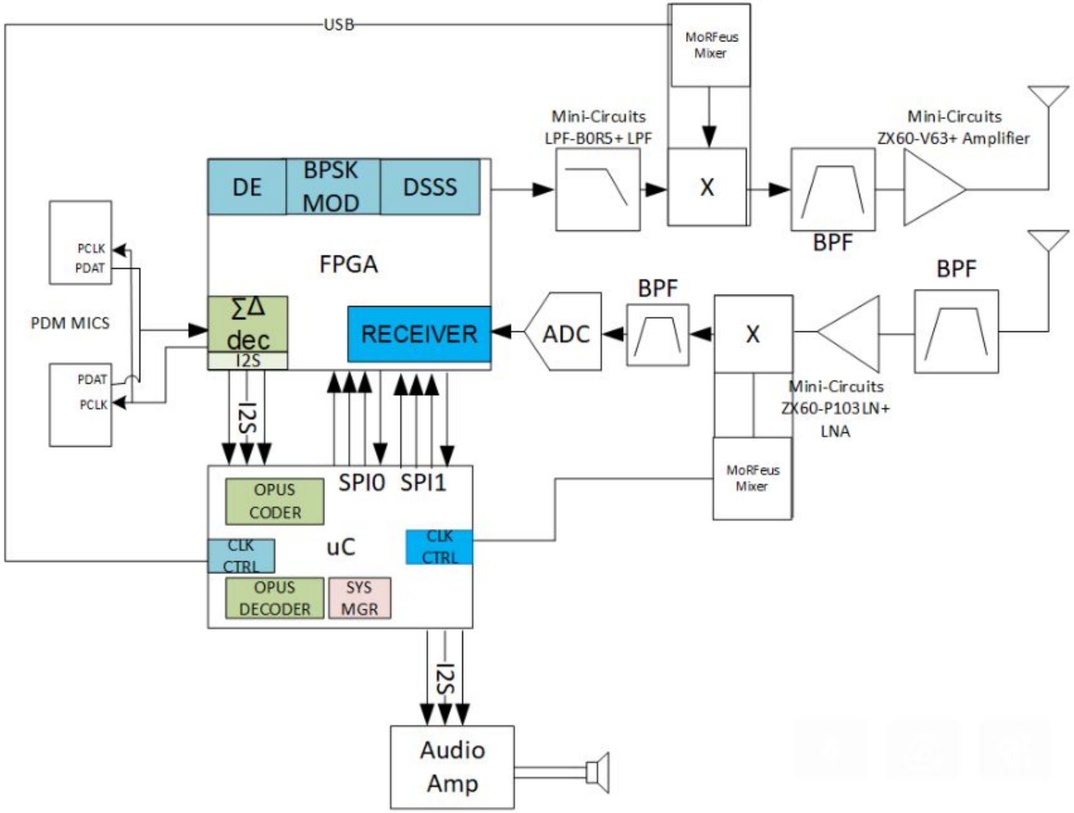
Design and build WPAN walkie talkies using 802.15.4 on 902-928MHz ISM Band



IEEE 802.15.4

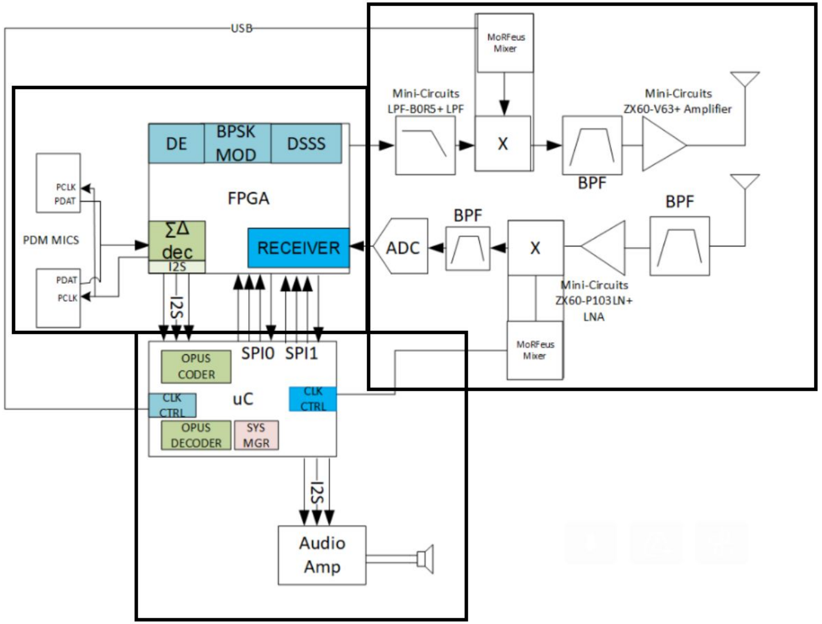
- Simple standard to local-area networks
- Covers 900 MHz and 2.4 GHz ISM bands, among others
- Clause 13 defines Pulse-shaped BPSK and Clause 12 defines OQPSK mode
  - Only Clause 13 BPSK is required in 900 MHz band
- Carrier-Sense Multiple Access with Collision Avoidance

Block Diagram



Subsystems

- 3 subsystems:
  - RF front end
  - FPGA
  - Microcontroller



Progress

Audio Front End/FPGA:

- Built floating point Matlab model of sigma-delta decimator, checked in
- Captured PDM audio using test jig
- Wrote SystemVerilog code and test benches for SPI and I2S, checked in
- Floating point Matlab model of transmitter blocks, checked in

Microcontroller:

- Implemented Opus source coder demo reading/writing audio data from an SD card, checked in
- Tested different bit error rates and code rates, checked in

RF Front End:

- Decision on RF front end scheme based on what we have.
- Parts search per general 802.15.4 required specification.
    - comparison required comp. spec vs. reference device`s comp spec vs. available components spec. checked in
- List up affordable parts to order: compromise between cost vs. demo level.

802.15.4 Project

Audio Front End

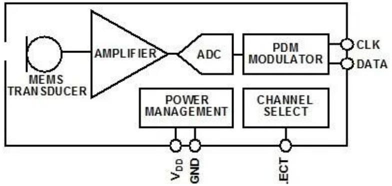
PDM Microphones

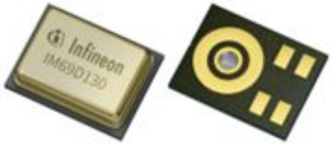
Analog MEMS microphones

- Sound pressure moves a micromechanical plate which causes a change in capacitance
  - Noise sensitive
  - Difficult to route --need analog supply, ferrite bead and <u>short</u> differential trace between microphone and host
  - Lower power
  - Cheaper

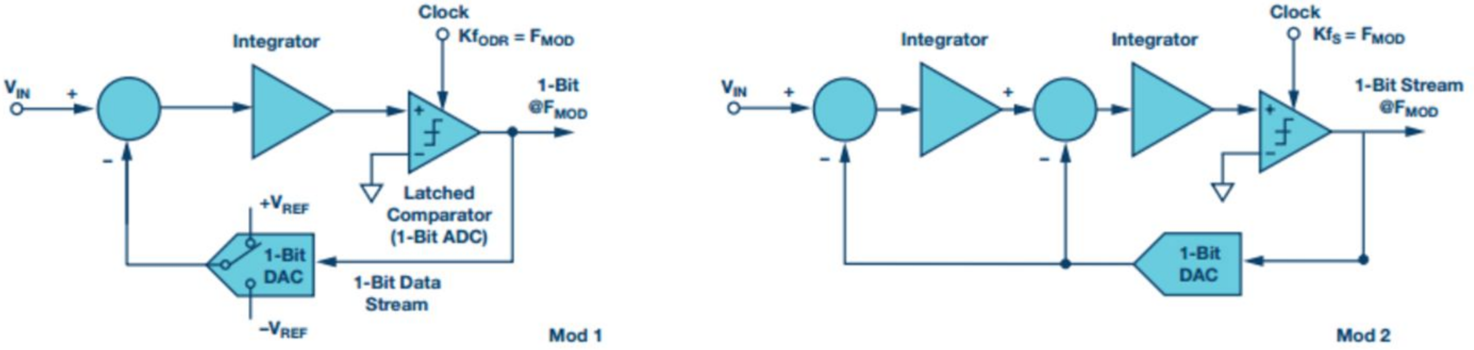
Pulse Density Modulation MEMS microphones

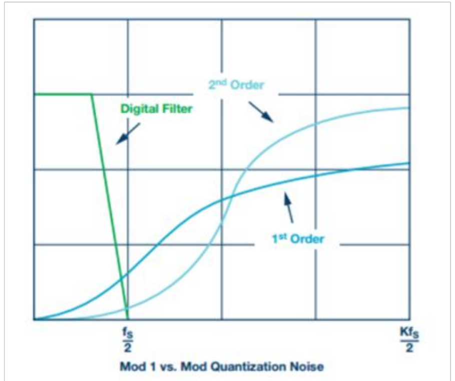
- Uses same MEMS sensor but converts analog voltage into digital inside the mic using sigma delta modulation
- Robust to interference
- Simpler to design-in – longer trace possible, beamforming, etc.
- Two microphones on same DAT trace (**single bit dual data rate**)
- High power (recent PDM microphones are much lower power, however)





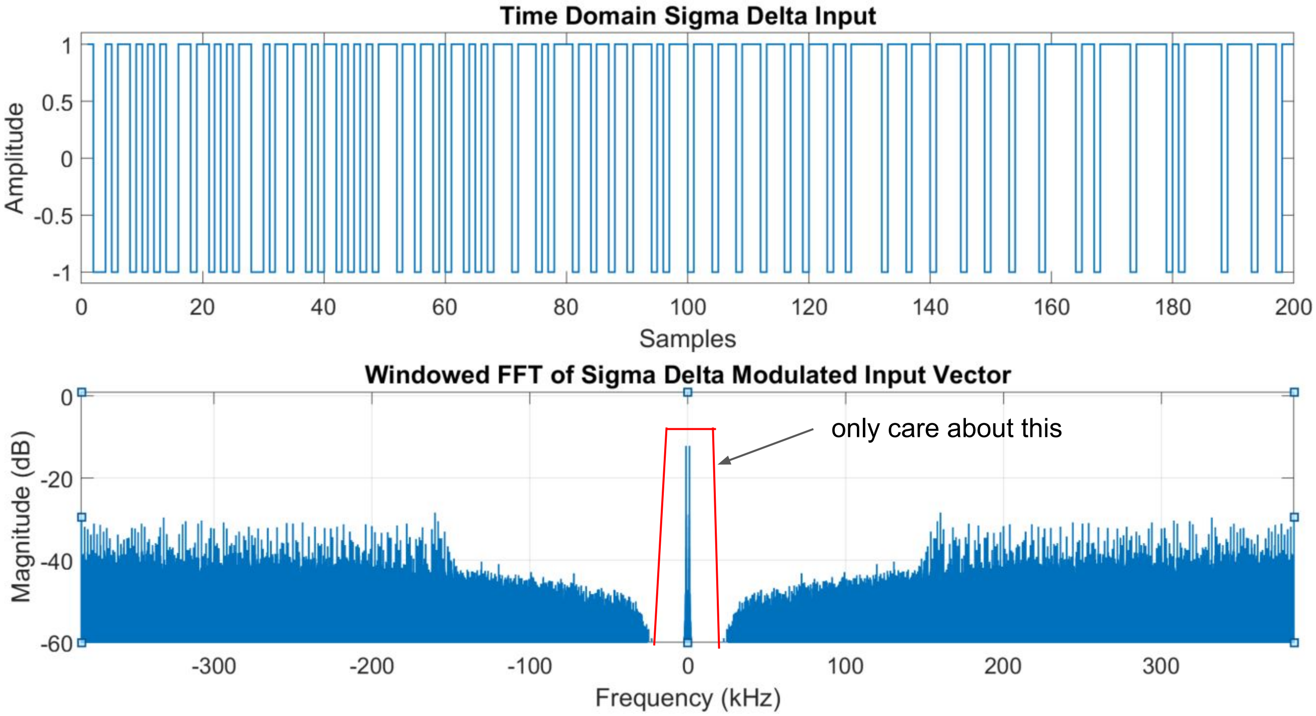
Sigma Delta Modulation





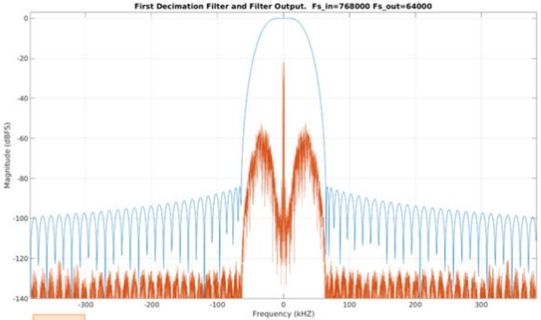
- Grossly oversample signal and quantize to +/-1
- Feedback error and sum with incoming data
- oversampling and error feedback does noise shaping pushing quantization noise above frequency of interest
- PDM mics use 4th or 5th order SD Modulation (not 1 or 2 like shown here)
- Output of PDM Mics is **1-bit digital data**

Sigma Delta Modulation

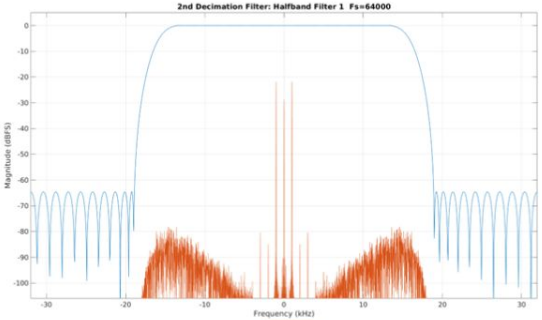


only care about this

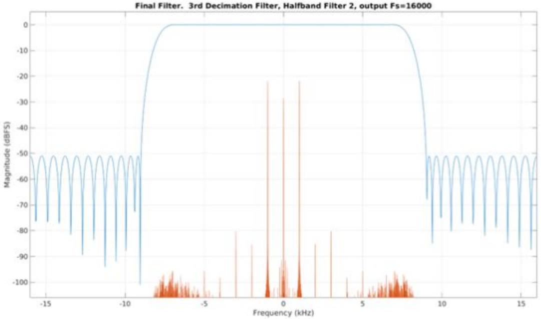
1-bit 768kHz PDM to 16-bit 16kHz PCM Conversion



Filter and
Decimate by 12

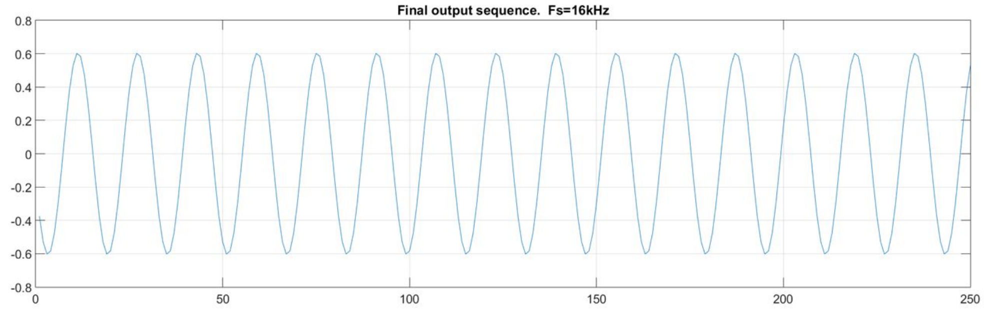


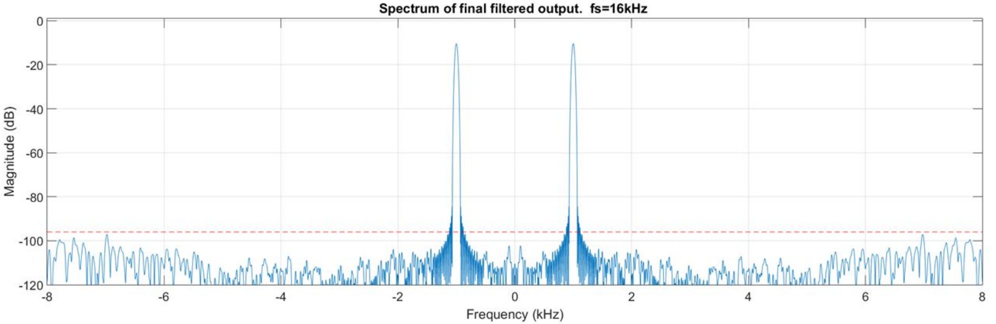
Filter and
Decimate by 2



Filter and
decimate
by 2

Final 16-kHz PCM Output





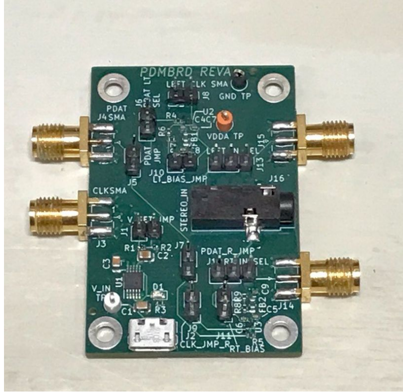
Overkill for this project…Relaxed filter constraints to save FPGA resources

PDM Filtering and Decimation

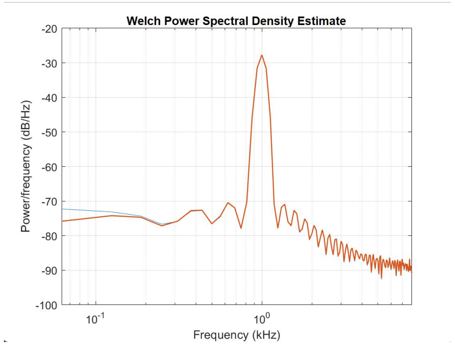
- Could brute force it (about 300 Taps)
- CIC Filter followed by 2 stages of half-band filters is typical
- Can do multi-stage half-band, IIR, all kinds of things
- Fred Harris suggested using a polyphase filter. Optimum for FPGAs.

Test Setup

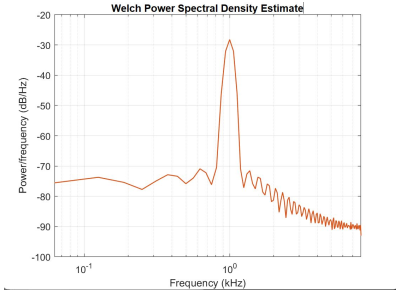
- Don’t have an anechoic chamber – makes repeatability difficult
- Built a PCB to convert analog audio to PDM using On Semi FAN3852



Audio Test Decimate by 12 using 36 or 48 taps in 1st stage

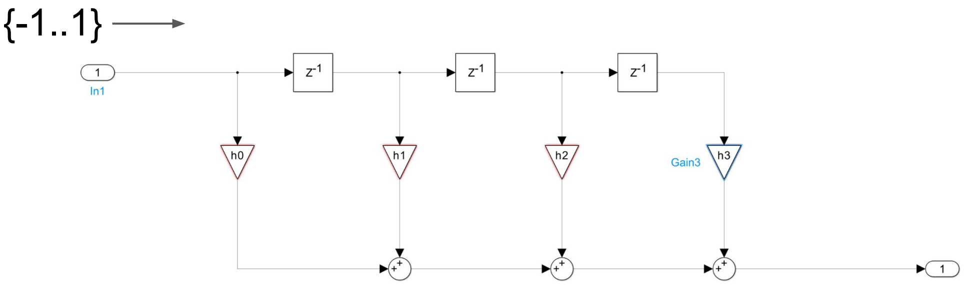


- 50dB attenuation in 1st stage
  - 36 Taps in 1st stage
- Fewer taps
- quantization noise audible



- 60dB attenuation
  - 48 Taps in 1st stage
- Less quantization noise

Implementation



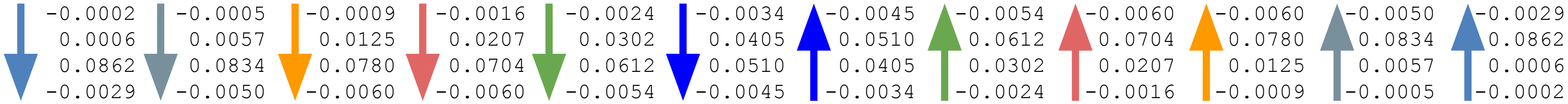
```
for jj=2:decimation_rate
        x_poly(jj,:)=[0 x(decimation_rate-jj+2:decimation_rate:end)];
end

for ll=1:decimation_rate
        y_poly_dec_leg(ll,:)=filter(poly_hh(:,ll),1,x_poly(ll,:));
end
y_poly_dec=sum( y_poly_dec_leg([1:size(y_poly_dec_leg,1)],:));
```

- One polyphase leg shown with standard FIR structure
- Could certainly implement with this structure in an FPGA
- Note that input vector to filter can be only be 16 values:
  - [1 1 1 1]
  - [-1 1 1 1]
  - [1 -1 1 1]
  - …
  - [-1 -1 -1 -1]

Polyphase Coefficients

poly_hh =

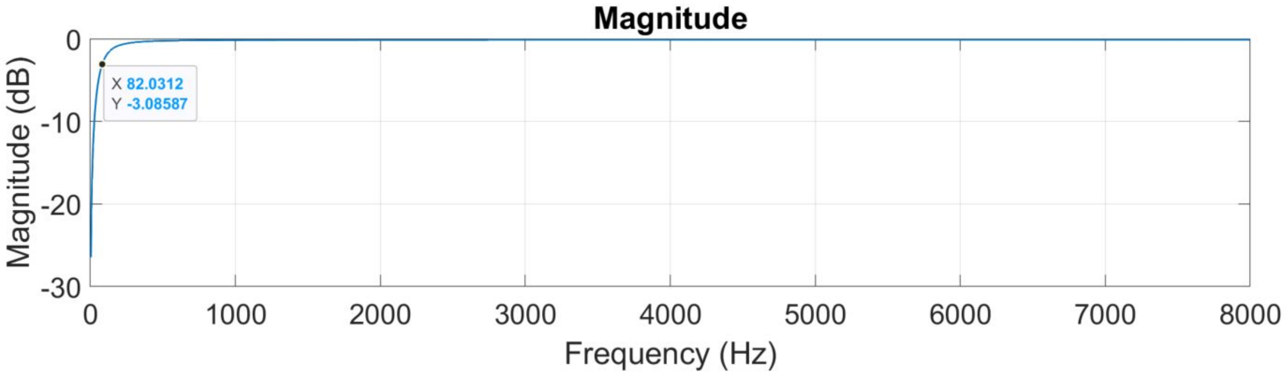


```
  -0.0002  -0.0005  -0.0009  -0.0016  -0.0024  -0.0034  -0.0045  -0.0054  -0.0060  -0.0060  -0.0050  -0.0029
   0.0006   0.0057   0.0125   0.0207   0.0302   0.0405   0.0510   0.0612   0.0704   0.0780   0.0834   0.0862
   0.0862   0.0834   0.0780   0.0704   0.0612   0.0510   0.0405   0.0302   0.0207   0.0125   0.0057   0.0006
  -0.0029  -0.0050  -0.0060  -0.0060  -0.0054  -0.0045  -0.0034  -0.0024  -0.0016  -0.0009  -0.0005  -0.0002
```

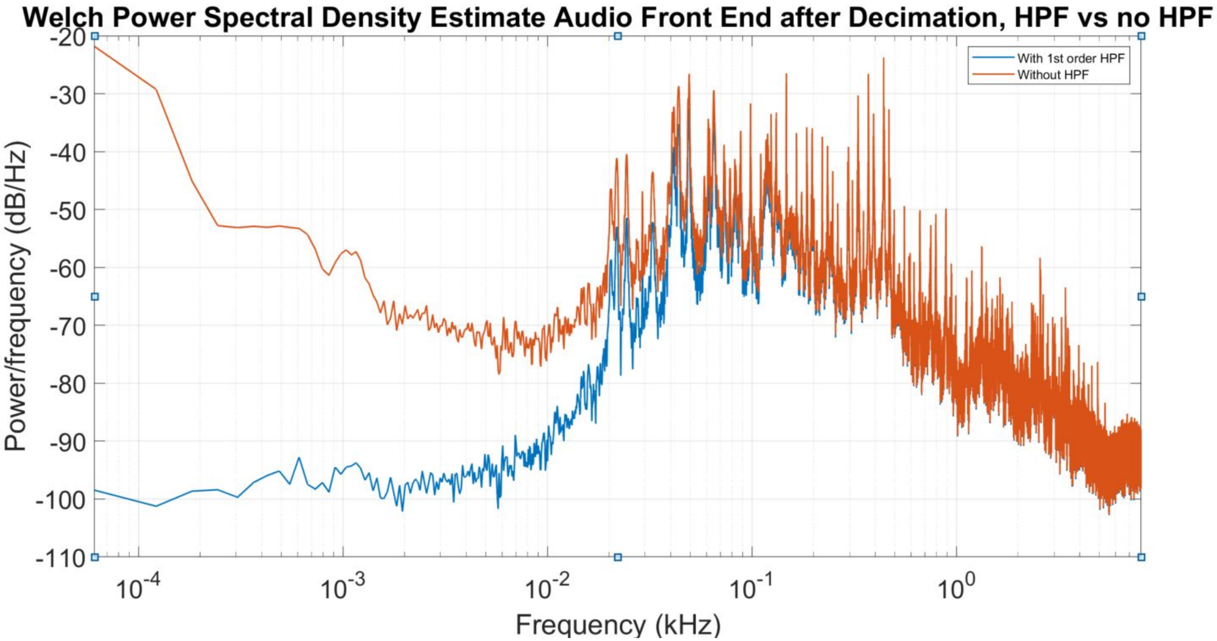
- Coefficients in each leg are mirrored and flipped
- Only store ½ the coefficients

High Pass Filter

- After decimation, add a 1st order high pass filter for DC offset
- $H(z) = \alpha \frac{1 - z^{-1}}{1 - \alpha z^{-1}}$
- $\alpha = 1 - 2^{-5}$ gives a filter corner of ~ 80Hz
- human ear doesn't care about nonlinear phase at low freq + cleans up voice quality a little

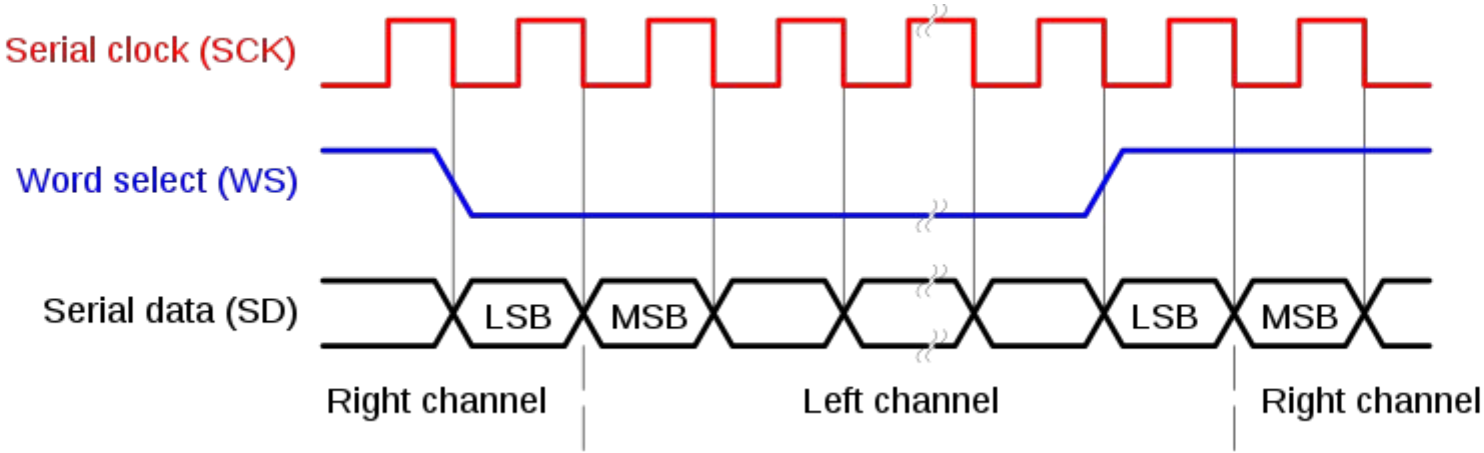


PSD comparison of Decimated Output With/Without HPF



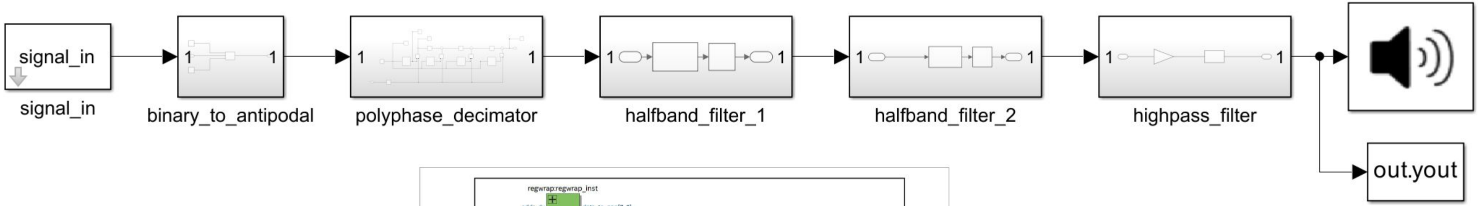
I2S

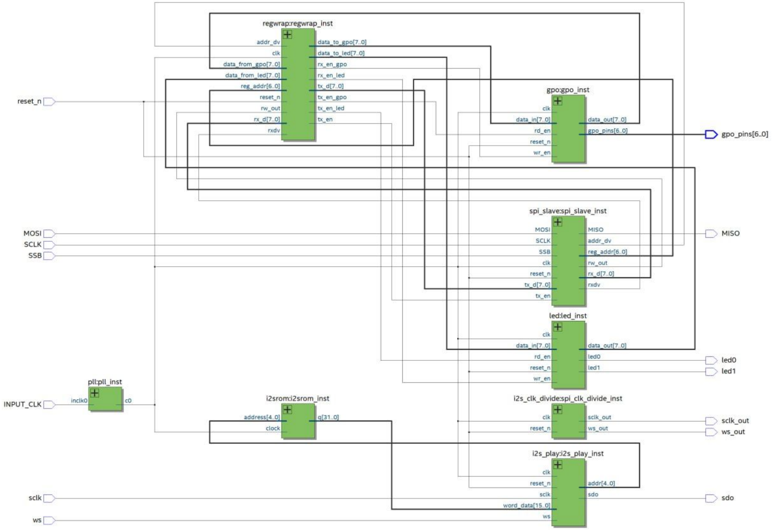
- I2S is a Philips (now NXP) standard from 1990s.
- Widely used in Audio
- 3 pins:



- FPGA will be I2S Target, Teensy will be I2S Controller

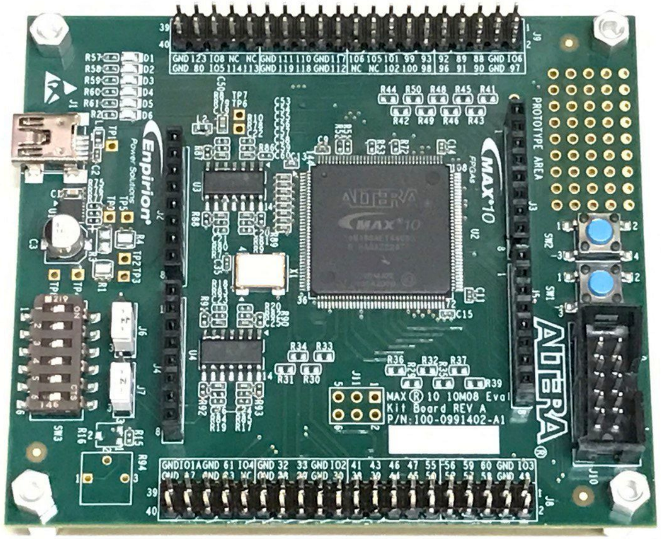
Simulink Model of Audio Front End and FPGA Top Level





FPGA

- Implements
  - SPI interfaces to uC
  - PDM decimator, I2s to uC
  - BPSK modulation and pulse shaping
  - TX to DDS
  - input from ADC
  - Demodulation
  - Demodulated bits to uC
- Intel 10M08 EVB
  - 55nm
  - 13 pins low-speed I/O
  - 18 pins high-speed single ended I/O + 15 differential I/O
  - 24 18x18 DSP blocks
  - $50



Future Work

FPGA:

- Implement Sigma Delta decimator in FPGA. Send out I2S, check waveform
- Implement BPSK modulator in FPGA. Check digital waveform using Logic Analyzer. vector match
- RX Input from ADC
- Demodulator

Integration:

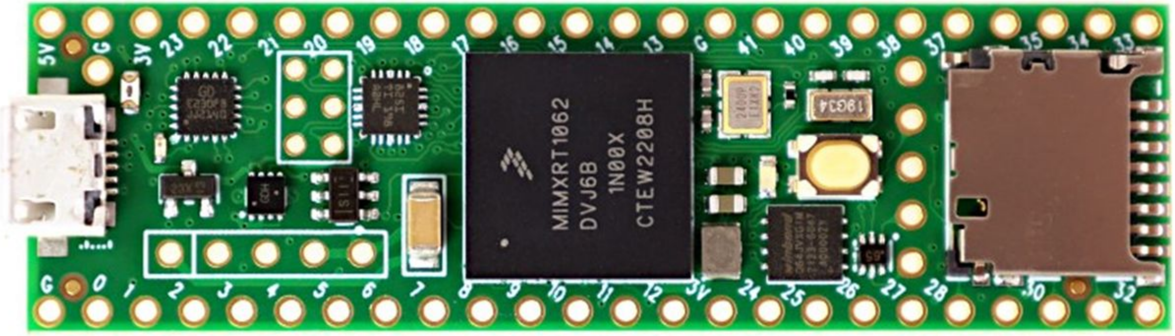
- Connect Microcontroller to FPGA. Read and write FPGA registers via SPI
- Validate I2S from FPGA

802.15.4 Project

Microcontroller

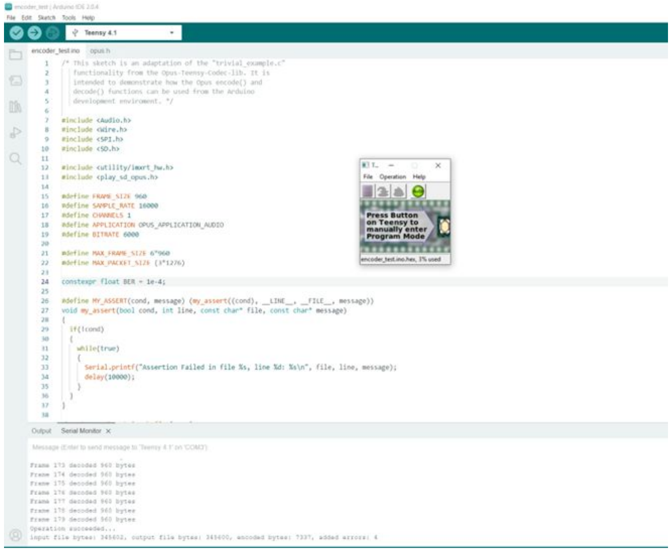
Teensy 4.1

- ARM Cortex-M7 at 600 MHz
- Float point math unit, 64 & 32 bits
- 7936K Flash, 1024K RAM
- 2 I2S, 3 SPI
- SD card slot
- $30!



Development Environment

- “Teensyduino” plug in for Arduino IDE
- Supports Arduino built-in functions and serial monitor
- Teensy loader application can directly load binaries without the IDE



encoder_test | Arduino IDE 2.0.4
File Edit Sketch Tools Help
Teensy 4.1
encoder_test.ino
opus.h
T...
File Operation Help
Output
Serial Monitor
Message (Enter to send message to 'Teensy 4.1' on 'COM3')

```
/* This sketch is an adaptation of the "trivial_example.c"
   functionality from the Opus-Teensy-Codec-lib. It is
   intended to demonstrate how the Opus encode() and
   decode() functions can be used from the Arduino
   development enviroment. */

#include <Audio.h>
#include <Wire.h>
#include <SPI.h>
#include <SD.h>

#include <utility/imxrt_hw.h>
#include <play_sd_opus.h>

#define FRAME_SIZE 960
#define SAMPLE_RATE 16000
#define CHANNELS 1
#define APPLICATION OPUS_APPLICATION_AUDIO
#define BITRATE 6000

#define MAX_FRAME_SIZE 6*960
#define MAX_PACKET_SIZE (3*1276)

constexpr float BER = 1e-4;

#define MY_ASSERT(cond, message) (my_assert((cond), __LINE__, __FILE__, message))
void my_assert(bool cond, int line, const char* file, const char* message)
{
  if(!cond)
  {
    while(true)
    {
      Serial.printf("Assertion Failed in file %s, line %d: %s\n", file, line, message);
      delay(10000);
    }
  }
}
```

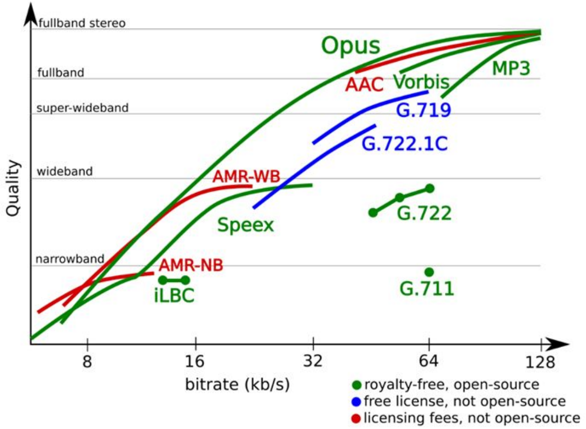
Press Button on Teensy to manually enter Program Mode

encoder_test.ino.hex, 3% used

```
Frame 173 decoded 960 bytes
Frame 174 decoded 960 bytes
Frame 175 decoded 960 bytes
Frame 176 decoded 960 bytes
Frame 177 decoded 960 bytes
Frame 178 decoded 960 bytes
Frame 179 decoded 960 bytes
Operation succeeded...
input file bytes: 345602, output file bytes: 345600, encoded bytes: 7337, added errors: 4
```

Opus Codec

- Open-source audio source coder used by Skype
- One of the best source coders available
- Encoder input and decoder output is 16-bit Pulse Code Modulation data
- Encoded bit rates as low as 6kbps
- Input sample rates as low as 8kbps

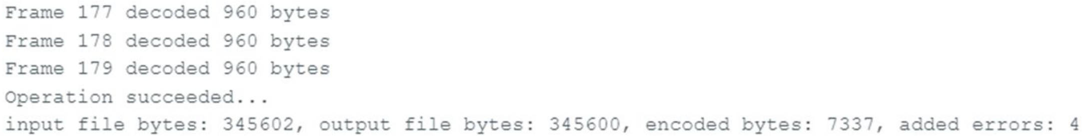


Opus Library for Teensy

- The Arduino-Teensy-Codec-lib has an open-source implementation of the Opus Codec for the Teensy
- It allows you to specify the encoded bit rate and input/output sample rates
- Library is designed for packetization, allowing you specify your desired packet size for each iteration of the encoder

Opus Demo

- Uses Matlab to generate 16-bit PCM binary files from Jay’s PDM to PCM demo or from a wav file
- Input binary is saved to a SD card and put in Teensy’s SD card slot
- Teensy reads the input binary, encodes it, injects bit errors, decodes it, and then saves the output to a new binary file
- Output binary is read by Matlab and converted to wav file
- This allows us to test the quality of the audio at different encoded bit rates and BERs!



```
Frame 177 decoded 960 bytes
Frame 178 decoded 960 bytes
Frame 179 decoded 960 bytes
Operation succeeded...
input file bytes: 345602, output file bytes: 345600, encoded bytes: 7337, added errors: 4
```

Opus Demo Results (Speech)

- 6000 bps:
  - BER 0: 🔊
  - BER 1e-4: 🔊
- 12000 bps:
  - BER 0: 🔊
  - BER 1e-4: 🔊

Opus Demo Results (Music)

- 6000 bps: 🔊
- 12000 bps: 🔊
- 16000 bps: 🔊

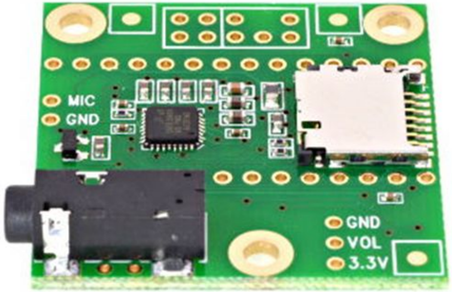
- 12000 bps:
  - BER 1e-4: 🔊
  - BER 1e-3: 🔊

Demo Takeaways

- 12000 kbps is suitable for most audio
- Effect of BER is similar at different levels of compression
- The quality for non-speech audio is worse at higher levels of compression than for speech audio

Future Work

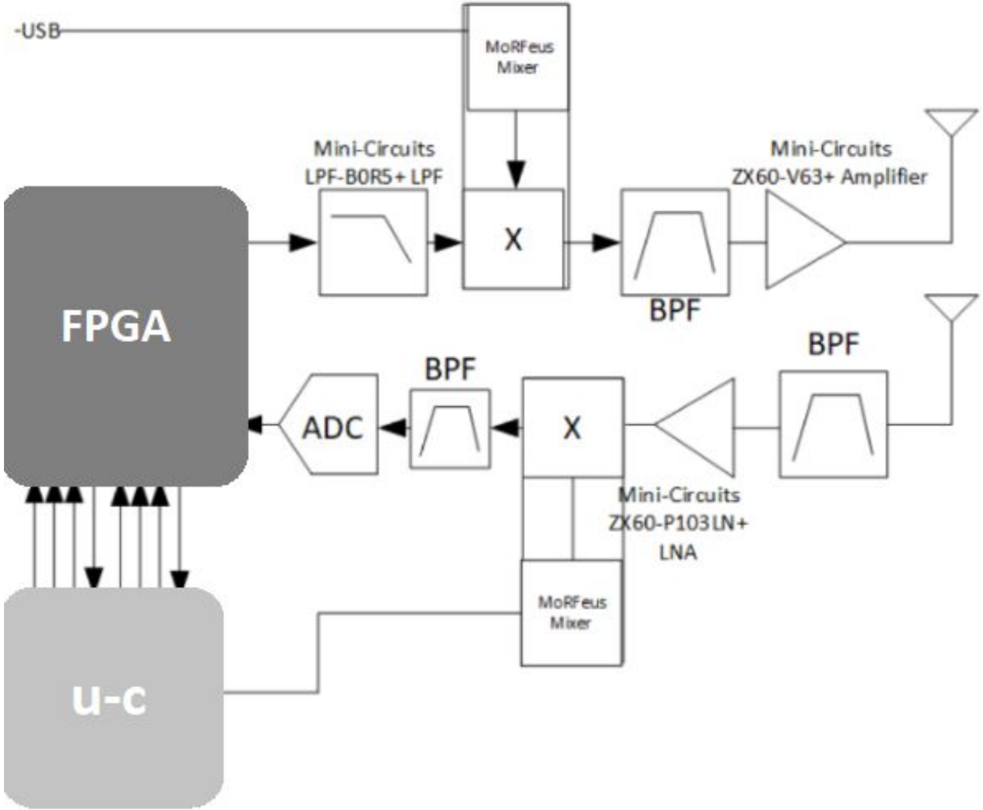
- I2S and SPI interface to FGPA, want to get working ASAP
- I2S interface to output speaker, may use the Teensy Audio shield board
- Packetizing opus decoder output into 802.15.4 packets



802.15.4 Project

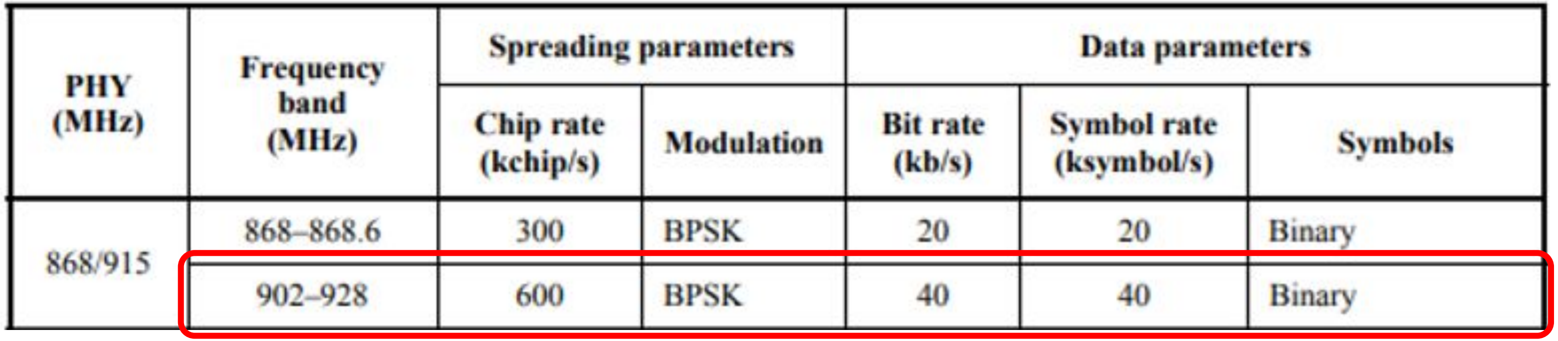
RF Front End + Mixed signal

Block Diagram



-USB
MoRFeus Mixer
Mini-Circuits LPF-B0R5+ LPF
X
Mini-Circuits ZX60-V63+ Amplifier
BPF
FPGA
BPF
BPF
ADC
X
Mini-Circuits ZX60-P103LN+ LNA
MoRFeus Mixer
u-c

Modulation and Spreading



PHY (MHz)	Frequency band (MHz)	Spreading parameters		Data parameters		
		Chip rate (kchip/s)	Modulation	Bit rate (kb/s)	Symbol rate (ksymbol/s)	Symbols
868/915	868–868.6	300	BPSK	20	20	Binary
	902–928	600	BPSK	40	40	Binary

- no IQ modulator demodulator
- no AGC

TRX

Antenna

- Frequency range: 902-928 MHz (US)
- Gain: typically 2-5 dBi
- Polarization: linear or circular
- Radiation pattern: omnidirectional or directional
- Connector: typically SMA or u.FL

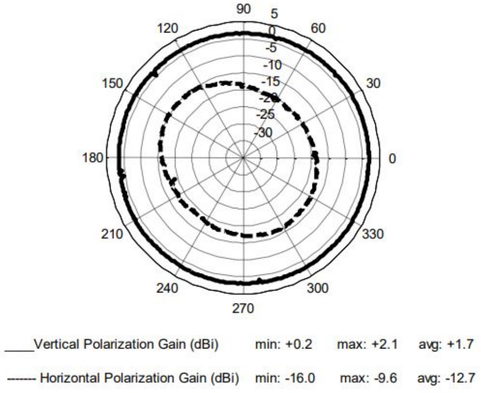


Figure 2 Extended Position, Vertical

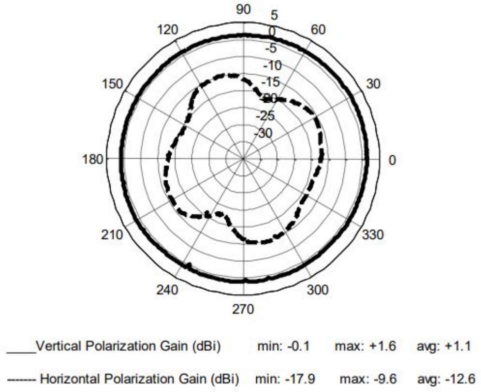
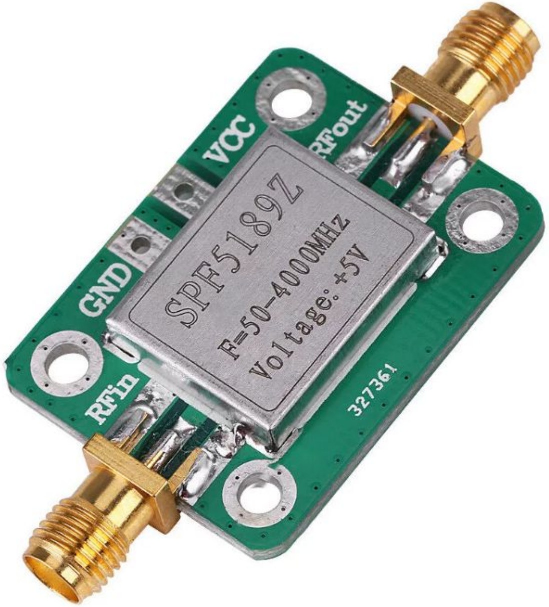


Figure 3 Folded Position, Vertical

RX

LNA



SPF5189

- **RF Low Noise Amplifier**
- **Freq: 50-4000MHz**
- **Gain : ~17dB**
- **NF: 0.6dB**

Rx RF BPF

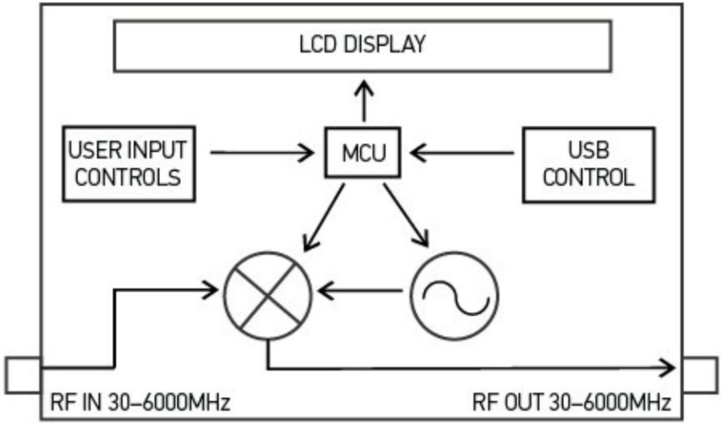
FBP-915s

- **Impedance:50 ohms**
- **Center frequency:915MHz**
- **Passband:902～928MHz**
- **Band Insertion Loss:<2.5 dB**
- **Out-of-band isolation: > 40 dB @ 815 MHz and 1015 MHz**
- **Maximum load power:300 mW (25 dBm)**
- **Interface:SMA-M, SMA-F**

Mixer + PLL + TCXO

moRFeus

- RF input frequency: 30 MHz – 6 GHz
- RF output frequency: 30 MHz – 6 GHz
- LO frequency: 85 MHz – 5400 MHz
- Fractional-N synthesizer
- LO step size: 1.5 – 3 Hz*
- Up/Down conversion both are OK

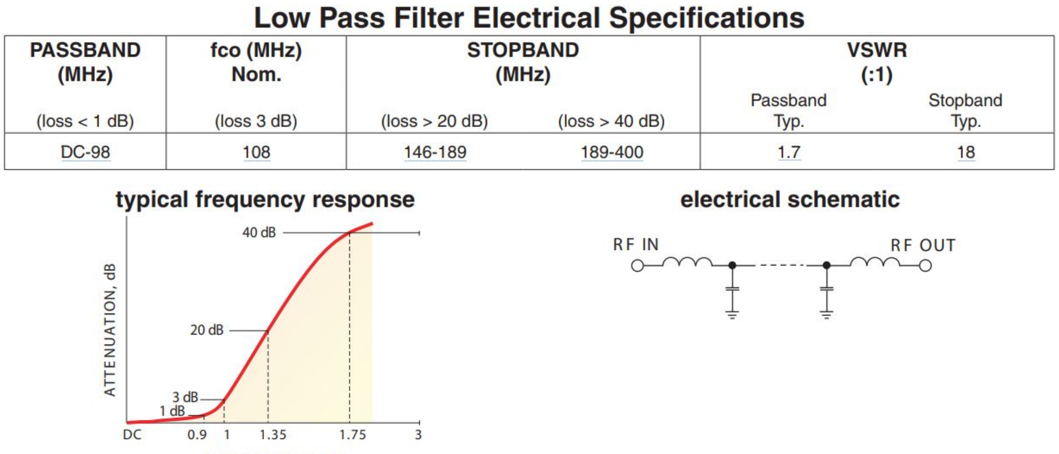


(*alternative option for 2nd unit:)
(Mini-Circuits ZAD-2 mixer + SigGen with ref sync)
(* UCSD SigGen)

Rx IF filter

SLP100+

- **Pass band: DC~98MHz,**
- **IL:0.58 @900MHz**
- **2fo rejection : 48dB,**
- **3fo rejection : ~75dB**



Low Pass Filter Electrical Specifications

PASSBAND (MHz)	fco (MHz) Nom.	STOPBAND (MHz)		VSWR (:1)	
				Passband Typ.	Stopband Typ.
(loss < 1 dB)	(loss 3 dB)	(loss > 20 dB)	(loss > 40 dB)		
DC-98	108	146-189	189-400	1.7	18

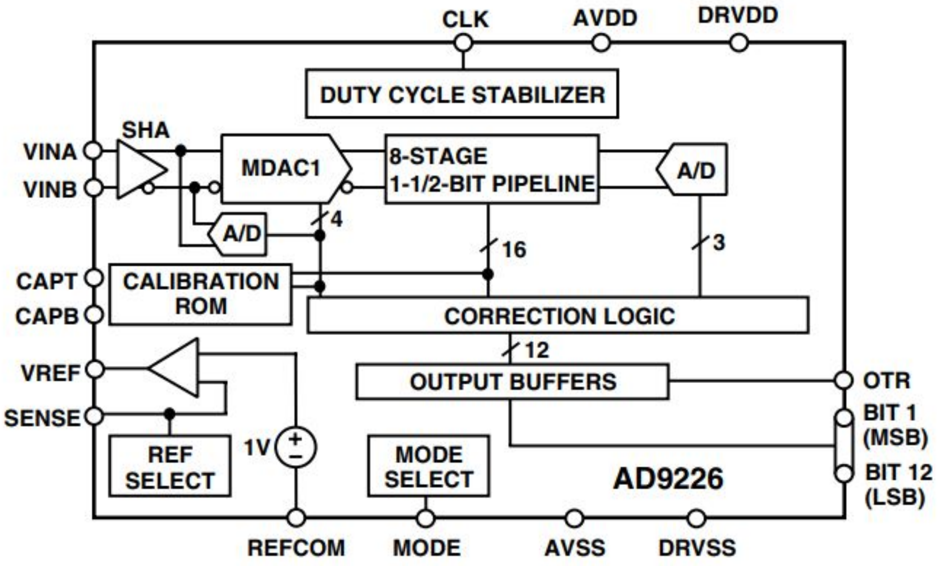
typical frequency response

ATTENUATION, dB
40 dB
20 dB
3 dB
1 dB
DC
0.9
1
1.35
1.75
3

electrical schematic

RF IN
RF OUT

ADC



AD9226 ADC Module

Signal-to-Noise Ratio: 69 dB @ fIN = 31 MHz

Spurious-Free Dynamic Range: 85 dB @ fIN = 31 MHz

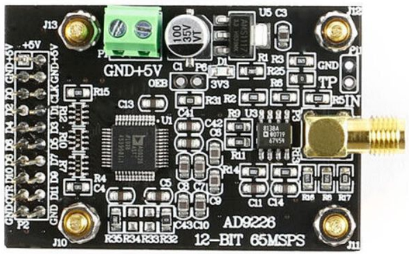
Intermodulation Distortion of –75 dBFS @ fIN = 140 MHz

ENOB = 11.1 @ fIN = 10 MHz

Low-Power Dissipation: 475 mW

Differential Nonlinearity Error: 0.6 LSB

Integral Nonlinearity Error: 0.6 LSB



TX

DDS

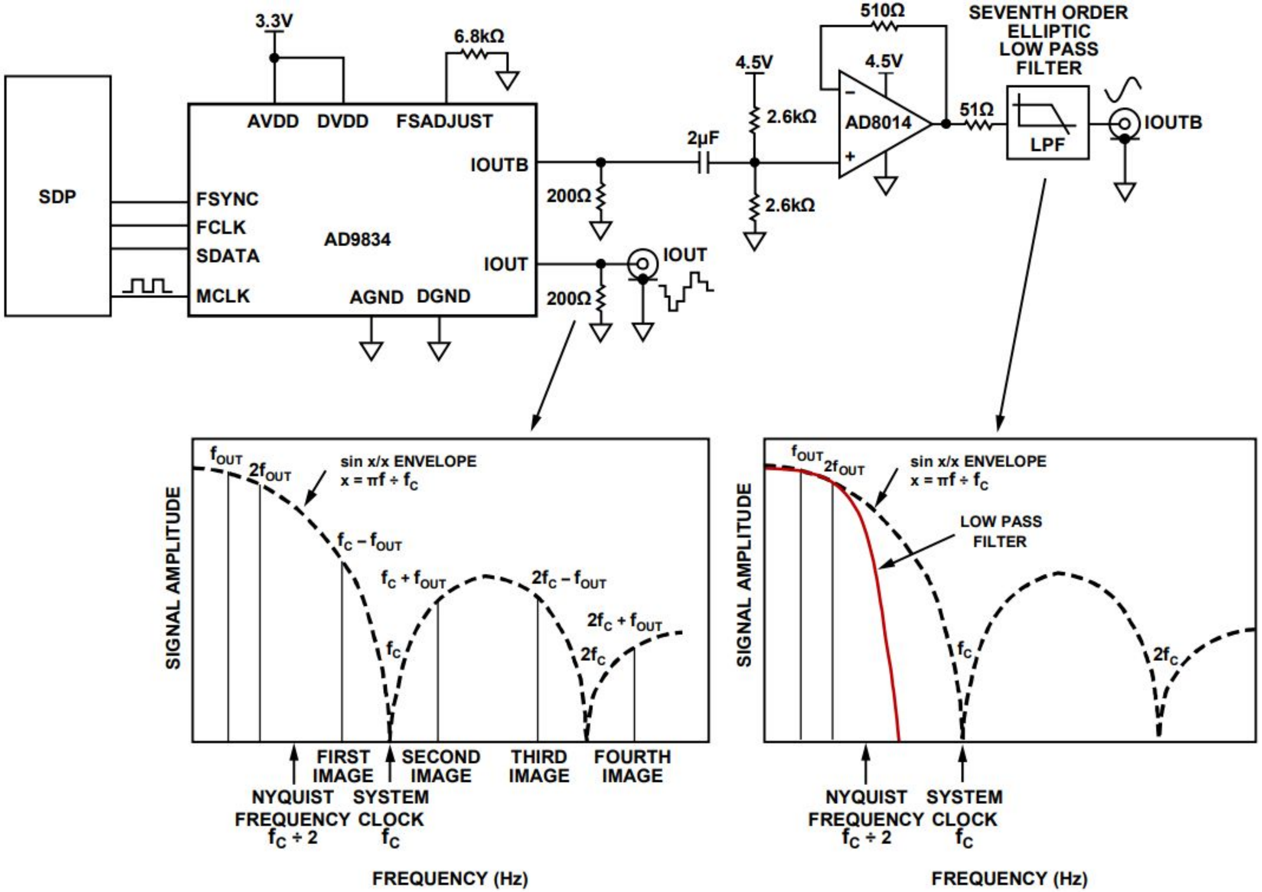


Figure 1. Low Power Waveform Generator (Simplified Schematic: All Connections and Decoupling Not Shown)

Functional Block Diagram

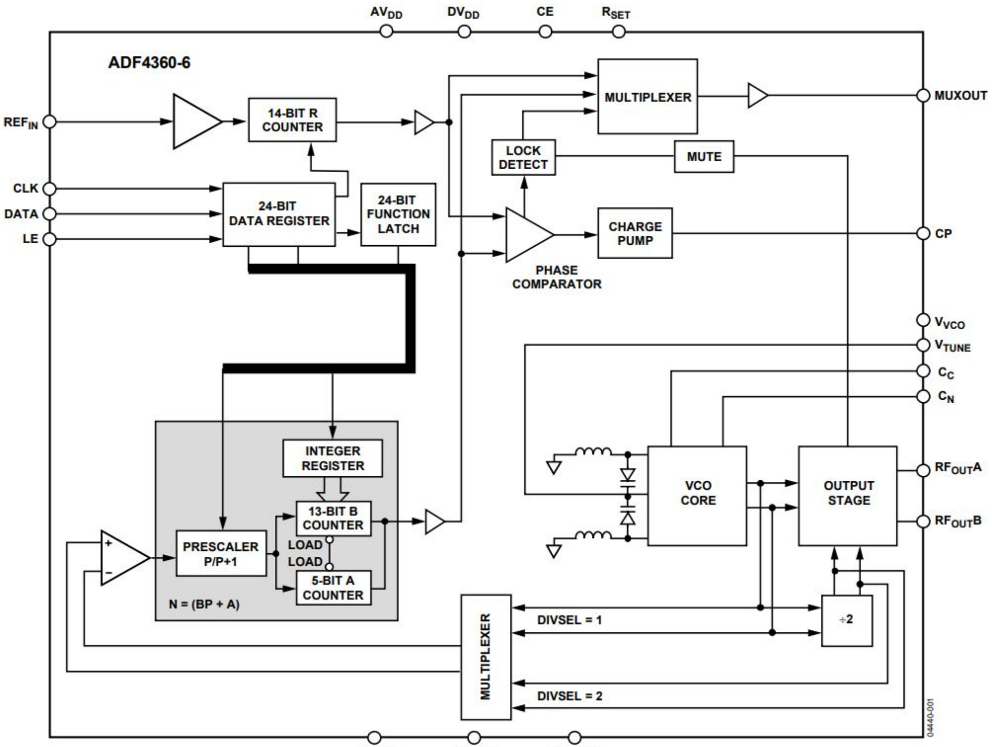


Figure 1.

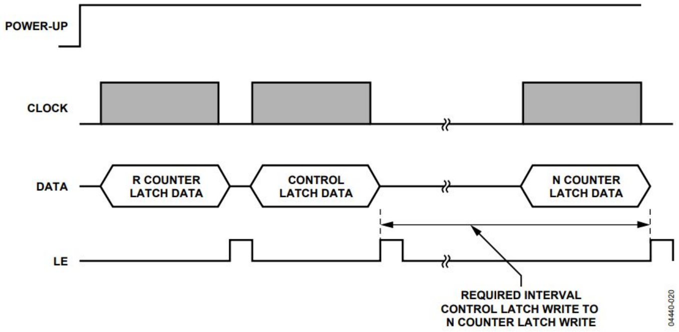
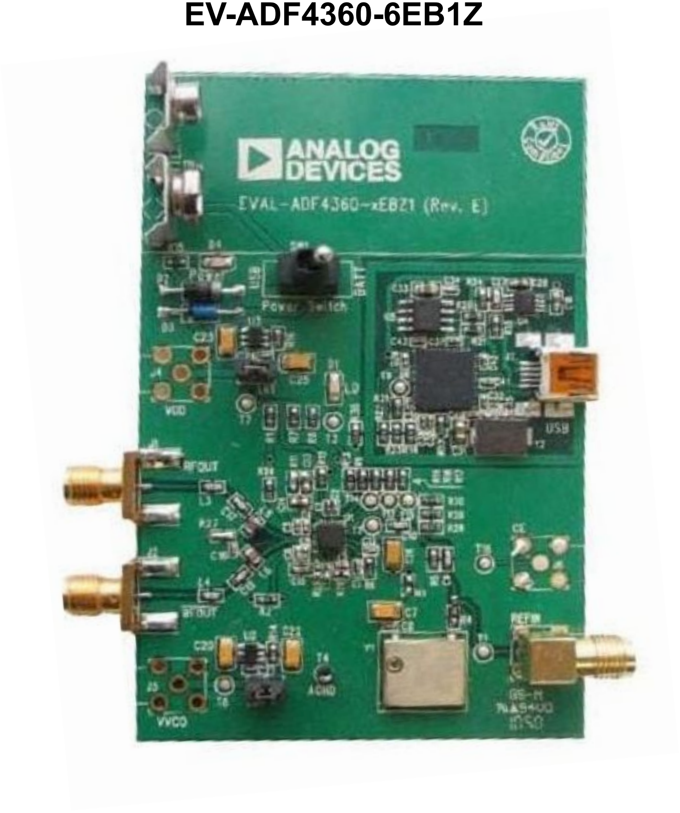


Figure 16. ADF4360-6 Power-Up Timing

DDS

EV-ADF4360-6EB1Z



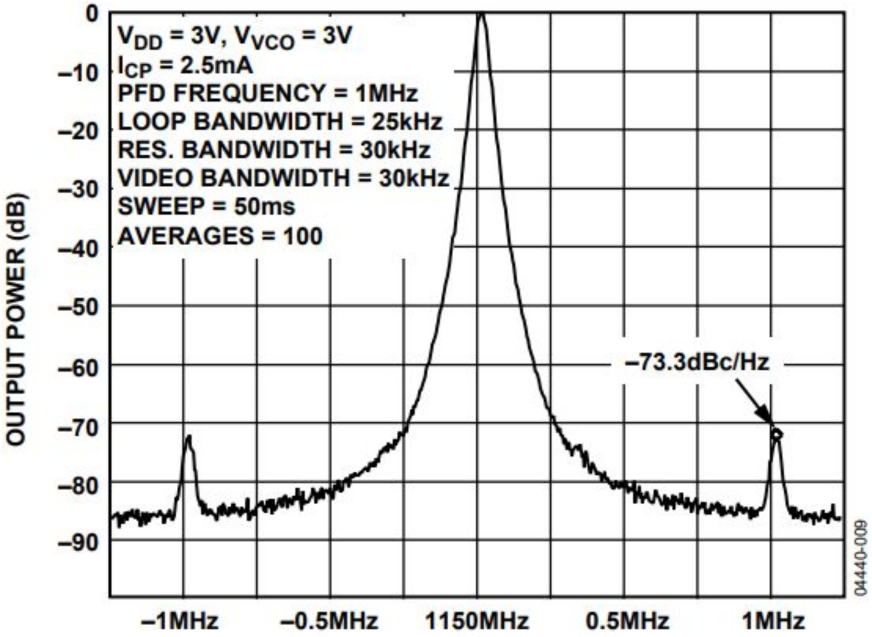
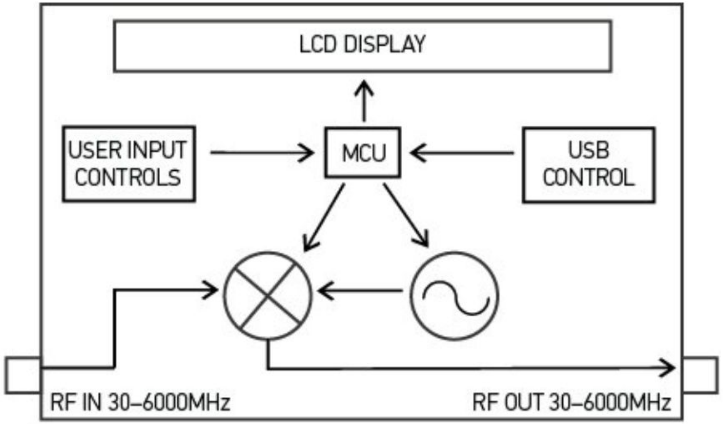


Figure 9. Reference Spurs at 1150 MHz
(1 MHz Channel Spacing, 25 kHz Loop Bandwidth)

Mixer + PLL + TCXO

moRFeus

- RF input frequency: 30 MHz – 6 GHz
- RF output frequency: 30 MHz – 6 GHz
- LO frequency: 85 MHz – 5400 MHz
- Fractional-N synthesizer
- LO step size: 1.5 – 3 Hz*
- Up/Down conversion both are OK



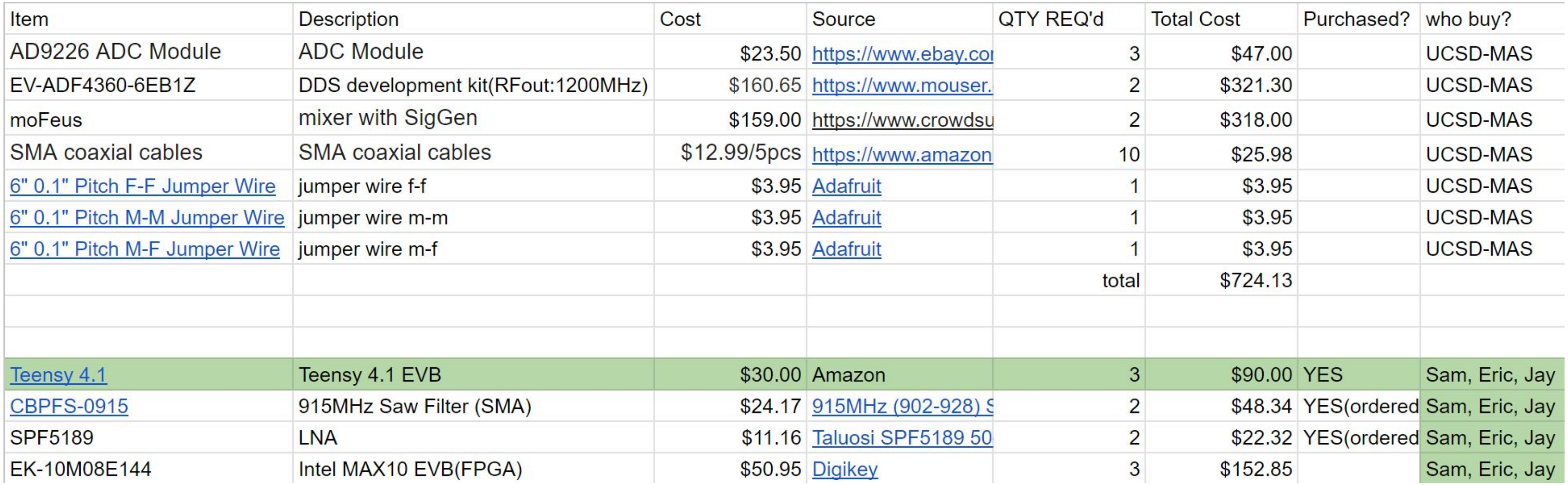
(*alternative option for 2nd unit:)
(Mini-Circuits ZAD-2 mixer + SigGen with ref sync)
(* UCSD SigGen)

PA

ZX60-V63+

- **Frequency:** 50MHz ~ 6GHz
- **P1dB :** 18.5dBm
- **Gain :** 15.4dB
- **Noise Figure :** 4.3dB
- **Voltage - Supply :** 4.8V ~ 5.2V
- **Current - Supply :** 69mA
- SMA Connectors

RF Frontend + Mixed signal parts list



Item	Description	Cost	Source	QTY REQ'd	Total Cost	Purchased?	who buy?
AD9226 ADC Module	ADC Module	$23.50	https://www.ebay.coı	3	$47.00		UCSD-MAS
EV-ADF4360-6EB1Z	DDS development kit(RFout:1200MHz)	$160.65	https://www.mouser.	2	$321.30		UCSD-MAS
moFeus	mixer with SigGen	$159.00	https://www.crowdsu	2	$318.00		UCSD-MAS
SMA coaxial cables	SMA coaxial cables	$12.99/5pcs	https://www.amazon	10	$25.98		UCSD-MAS
6" 0.1" Pitch F-F Jumper Wire	jumper wire f-f	$3.95	Adafruit	1	$3.95		UCSD-MAS
6" 0.1" Pitch M-M Jumper Wire	jumper wire m-m	$3.95	Adafruit	1	$3.95		UCSD-MAS
6" 0.1" Pitch M-F Jumper Wire	jumper wire m-f	$3.95	Adafruit	1	$3.95		UCSD-MAS
				total	$724.13		
Teensy 4.1	Teensy 4.1 EVB	$30.00	Amazon	3	$90.00	YES	Sam, Eric, Jay
CBPFS-0915	915MHz Saw Filter (SMA)	$24.17	915MHz (902-928) S	2	$48.34	YES(ordered	Sam, Eric, Jay
SPF5189	LNA	$11.16	Taluosi SPF5189 50	2	$22.32	YES(ordered	Sam, Eric, Jay
EK-10M08E144	Intel MAX10 EVB(FPGA)	$50.95	Digikey	3	$152.85		Sam, Eric, Jay

Future work

- Link budget analysis : range vs. BER
- RF Frontend + chanel model with Simulink for timing recovery simulation of baseband.
- Rx/Tx chain integration and performance test.
- Mixed signal part & FPGA interoperability test.
- Troubleshooting for whole system operation.