# Progressing 802.11ah Implementation in GNU Radio with gr-halow

GRCon24
Knoxville Tennessee

SEPTEMBER 19TH, 2024

SAMUEL MILLER

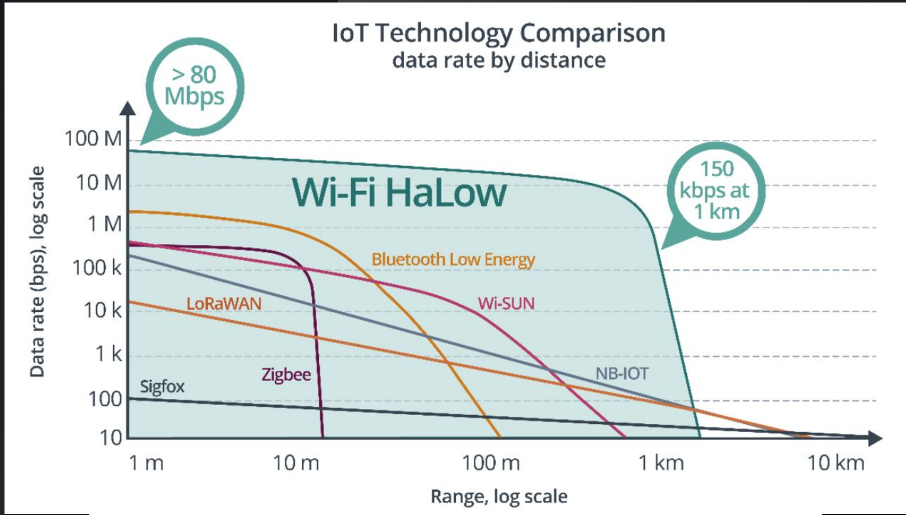
# What is the Problem?

802.11ah (HaLow) technology is not readily available so it is easier and more convenient to keep using other internet of things (IoT) technologies.

**10** Wi-Fi CERTIFIED HaLow devices
*versus*
**75569** Wi-Fi Certified 802.11b devices
**12501** Wi-Fi Certified 802.11g devices
**2501** Wi-Fi Certified 802.11n devices*

Open Source implementations:

- ZigBee: gr-ieee802-15-4
- Bluetooth: gr-bluetooth
- 802.11a/g/p: gr-ieee802-11
- LoRa: gr-lora & gr-lora_sdr
- HaLow: ???




| Attributes | Wi-Fi HaLow | Bluetooth Low Energy | Z-Wave | Zigbee | Wi-SUN | Sigfox | LoRaWAN | NB-IoT |
|---|---|---|---|---|---|---|---|---|
| Frequency | Sub-1 GHz | 2.4 GHz | Sub-1 GHz | 2.4 GHz / Sub-1 GHz | Sub-1 GHz | Sub-1 GHz | Sub-1 GHz | Licensed |
| Data rate (bps) | 150 k - 86.7 M8 | 125 k - 2 M | 9.6 k – 100 k | 250 k | 6.25 k – 800 k (50 k default) | 100 or 600 | 300 – 27 k | 20 k – 127 k |
| Range (m) | > 1 k | < 100 | < 30 | < 20 | < 1 k | < 40 k | < 10 k | < 10 k |
| Modulation | OFDM over BPSK, QPSK, 16/64/256 QAM | GFSK | GFSK | BPAK/ OQPSK | MR-FSK / MR-OFDM / MR-OQPSK | DBPSK/ GFSK | CSS | QPSK |
| Battery life | Years | Years | Years | Years | Years | Years | Years | Years |
| Security | WPA3 | 128-bit AES in CCMode | Security 2 (S2) | 128-bit AES in CCMode | IEEE 802.1X | Session-level security | 128-bit AES in CCMode | 3GPP security |
| OTA firmware updates | Supports | Supports | - | - | - | - | - | - |
| Subscription required | No | No | No | No | No | Yes | Yes | Yes |
| TCP/IP (internet) | Supports | - | - | - | - | - | - | - |
| Network topology | Star / Relays | P2P* / Mesh | Mesh | Mesh | Mesh | Star | Star | Star |
| Open standard | IEEE 802.11ah | Bluetooth SIG | Proprietary | IEEE 802.15.4 | IEEE 802.15.4g | Proprietary | Proprietary | 3GPP LTE Cat-NB1/NB2 |

* Peer-to-peer    Source information used for this table is publicly available

Source: Newracom

*Source: Wi-Fi Alliance

# A Solution and the Why

Create an open-source implementation of 802.11ah, gr-halow, to enable researchers, hobbyists, and developers experimentation with HaLow through SDR emulation

- HaLow Bandwidths (1, 2, 4, 8, 16 MHz) are within the range of consumer SDRs
  - Survey capability tested with RTL-SDR, AirSpy R2, HackRF, Pluto SDR
- HaLow Data rates are achievable by consumer processors without significant overflows
  - Survey capability tested with x86_64 Intel N95 CPU
- Sits in the middle of the distance *vs* data rate tradeoff for omnidirectional antennas; suitable for mobile data-demanding IoT applications
  - Drone telemetry and sensor streaming
  - Amateur Radio Emergency Data Network

Source: Teledatics HaLow 66 mile Distance Record

# My Approach

Make as much progress as possible towards an open source transceiver for unencrypted 1 MHz HaLow channels

I depend on gr-ieee802_11 for most of the PHY RX chain. Eventually, I would like the 802.11ah implementation to live within gr-ieee802_11 instead of its own standalone gr-halow repository.

HaLow Activity Scanning

1

Working - Determine if there is HaLow activity, and if there is, what channel it is on

Full Receive Implementation

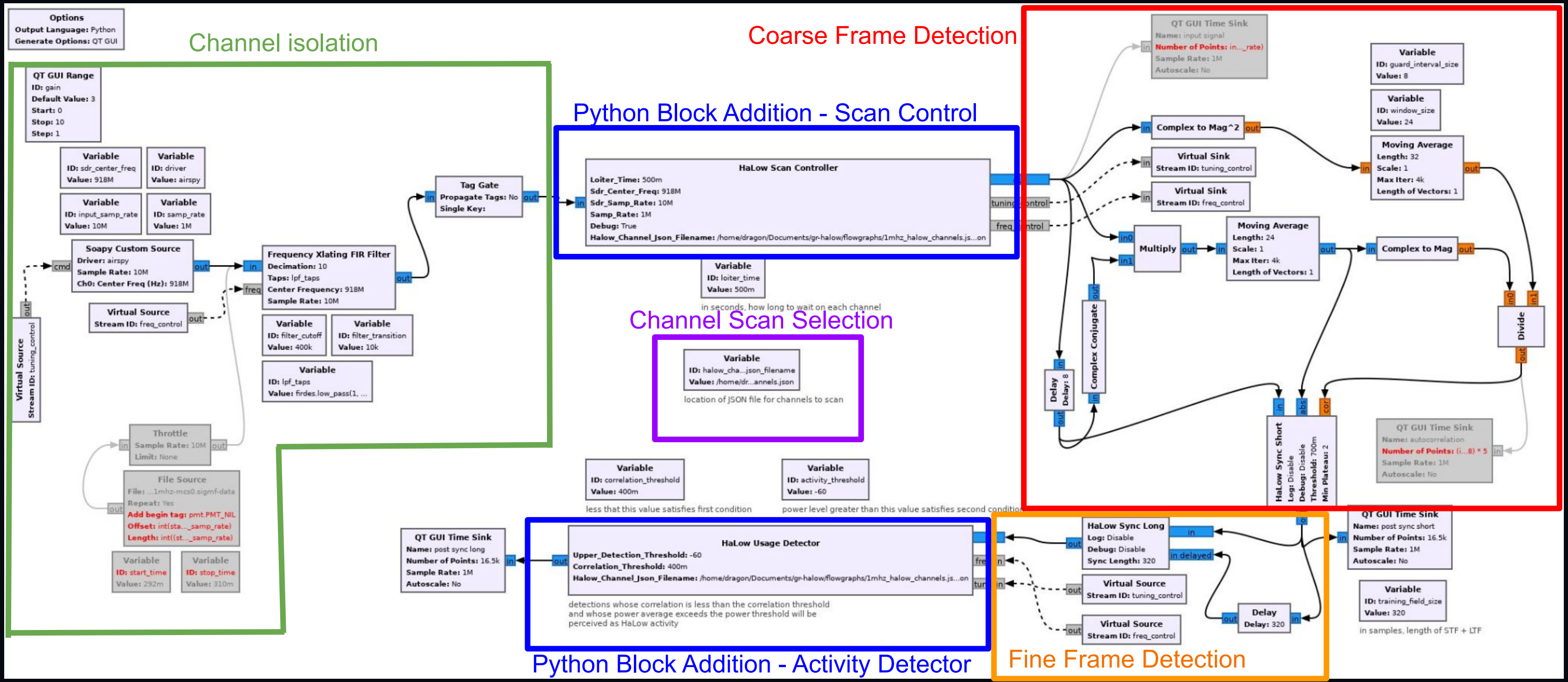
2

Implemented, not working - Decode information from the signal and data field of a HaLow packet

Full Transmit Implementation

3

Not implemented - Send information via HaLow



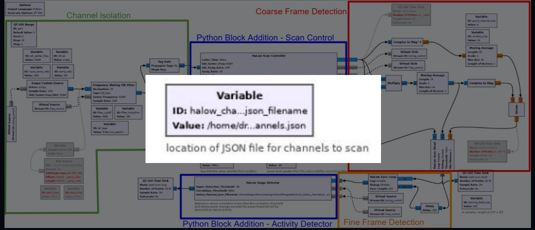

GNU RADIO VERSION 3.10.10.0

# HaLow Activity Scanning

Working - Determines if there is HaLow activity, and if there is, what channel it is on

irongiant33 on GitHub: gr-halow

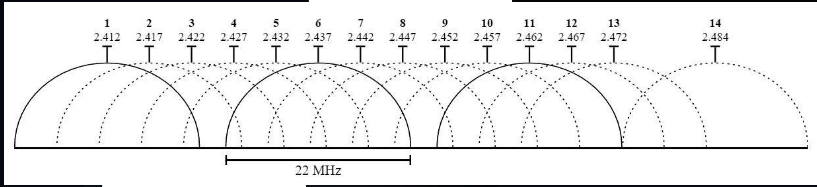
# HaLow Modifications




## 802.11a/g Channels

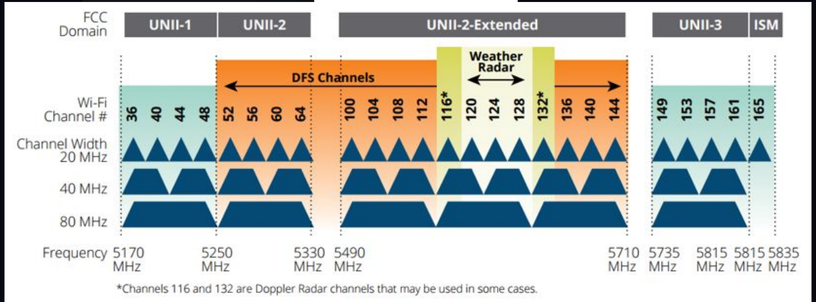
A list of all available channels, their frequencies, and bandwidths per p.4121 of the IEEE standard

802.11a




Source: Delta EU

802.11g

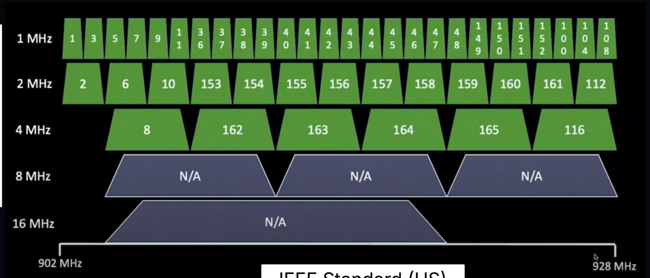



Source: Aruba Blogs

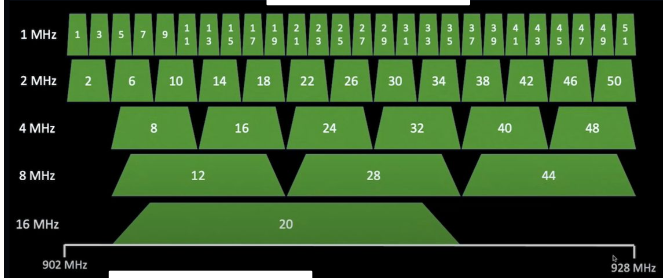
## HaLow Channels

Hardware only has to implement the 1 MHz and 2 MHz channels to be IEEE compliant. However, not all devices will be compliant and some may even list different channel numbers.

Alfa HaLow-U




IEEE Standard (US)




Source: Troy Martin

## Implemented:

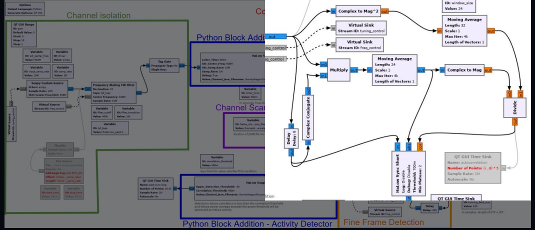
## JSON Config

gr-halow uses a JSON configuration file so users can specify the channel numbers, frequencies, and bandwidths of the channels they are interested in

```
{
        "1": {
                "freq": 902.5e6,
                "bw":   1e6
        },
        "3": {
                "freq": 903.5e6,
                "bw":   1e6
        },
        "5": {
                "freq": 904.5e6,
                "bw":   1e6
        },
        "7": {
                "freq": 905.5e6,
                "bw":   1e6
        },
```

Everything is based off of the IEEE 802.11-2020 MAC and PHY Specification

# HaLow Modifications




## 802.11a/g/p Timing

**Constant** number of subcarriers, **constant** relationship between the channel bandwidth and OFDM symbol period (80 samples per OFDM symbol) and guard interval

Table 17-5—Timing-related parameters

| Parameter | Value (20 MHz channel spacing) | Value (10 MHz channel spacing) | Value (5 MHz channel spacing) |
|---|---|---|---|
| $N_{SD}$: Number of data subcarriers | 48 | 48 | 48 |
| $N_{SP}$: Number of pilot subcarriers | 4 | 4 | 4 |
| $N_{ST}$: Number of subcarriers, total | 52 ($N_{SD} + N_{SP}$) | 52 ($N_{SD} + N_{SP}$) | 52 ($N_{SD} + N_{SP}$) |
| $\Delta_F$: Subcarrier frequency spacing | 0.3125 MHz (=20 MHz/64) | 0.156 25 MHz (= 10 MHz/64) | 0.078 125 MHz (= 5 MHz/64) |
| $T_{FFT}$: Inverse Fast Fourier Transform (IFFT) / Fast Fourier Transform (FFT) period | 3.2 µs ($1/\Delta_F$) | 6.4 µs ($1/\Delta_F$) | 12.8 µs ($1/\Delta_F$) |
| $T_{PREAMBLE}$: PHY preamble duration | 16 µs ($T_{SHORT} + T_{LONG}$) | 32 µs ($T_{SHORT} + T_{LONG}$) | 64 µs ($T_{SHORT} + T_{LONG}$) |
| $T_{SIGNAL}$: Duration of the SIGNAL BPSK-OFDM symbol | 4.0 µs ($T_{GI} + T_{FFT}$) | 8.0 µs ($T_{GI} + T_{FFT}$) | 16.0 µs ($T_{GI} + T_{FFT}$) |
| $T_{GI}$: GI duration | 0.8 µs ($T_{FFT}/4$) | 1.6 µs ($T_{FFT}/4$) | 3.2 µs ($T_{FFT}/4$) |
| $T_{GI2}$: Training symbol GI duration | 1.6 µs ($T_{FFT}/2$) | 3.2 µs ($T_{FFT}/2$) | 6.4 µs ($T_{FFT}/2$) |
| $T_{SYM}$: Symbol interval | 4 µs ($T_{GI} + T_{FFT}$) | 8 µs ($T_{GI} + T_{FFT}$) | 16 µs ($T_{GI} + T_{FFT}$) |
| $T_{SHORT}$: Short training sequence duration | 8 µs ($10 \times T_{FFT}/4$) | 16 µs ($10 \times T_{FFT}/4$) | 32 µs ($10 \times T_{FFT}/4$) |
| $T_{LONG}$: Long training sequence duration | 8 µs ($T_{GI2} + 2 \times T_{FFT}$) | 16 µs ($T_{GI2} + 2 \times T_{FFT}$) | 32 µs ($T_{GI2} + 2 \times T_{FFT}$) |

## HaLow Timing

**Variable** number of subcarriers, **variable** relationship between channel bandwidth and OFDM symbol period and guard interval

Table 23-4—Timing-related constants

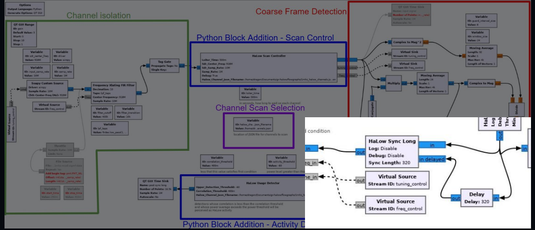
| Parameter | CBW1 | CBW2 | CBW4 | CBW8 | CBW16 | Description |
|---|---|---|---|---|---|---|
| $N_{SD}$ | 24 | 52 | 108 | 234 | 468 | Number of data subcarriers per OFDM symbol |
| $N_{SP}$ | 2 | 4 | 6 | 8 | 16 | Number of pilot subcarrier per OFDM symbol |
| $N_{ST}$ | 26 | 56 | 114 | 242 | 484 | Total number of useful subcarriers per OFDM symbol |
| $N_{SR}$ | 13 | 28 | 58 | 122 | 250 | Highest data subcarrier index per OFDM symbol |
| $\Delta_F$ | 31.25 kHz | | | | | Subcarrier frequency spacing |
| $T_{DFT}$ | 32 µs = $1/\Delta_F$ | | | | | IDFT/DFT period |
| $T_{GI}$ | 8 µs = $T_{DFT}/4$ | | | | | Guard interval duration |
| $T_{GI2}$ | 16 µs | | | | | Double guard interval |
| $T_{GIS}$ | 4 µs = $T_{DFT}/8$ | | | | | Short guard interval duration |
| $T_{SYML}$ | 40 µs = $T_{DFT} + T_{GI} = 1.25 \times T_{DFT}$ | | | | | Duration of OFDM symbol with normal guard interval |

## Implemented: Scope constrained to 1 MHz Channel

The variability in the parameters for each channel bandwidth in HaLow means that software-defined implementations will have to adapt parameters based on the channel that the user wishes to decode. This will add overhead in the form of object properties that will have to propagate throughout the receive chain

Everything is based off of the IEEE 802.11-2020 MAC and PHY Specification

# HaLow Additions



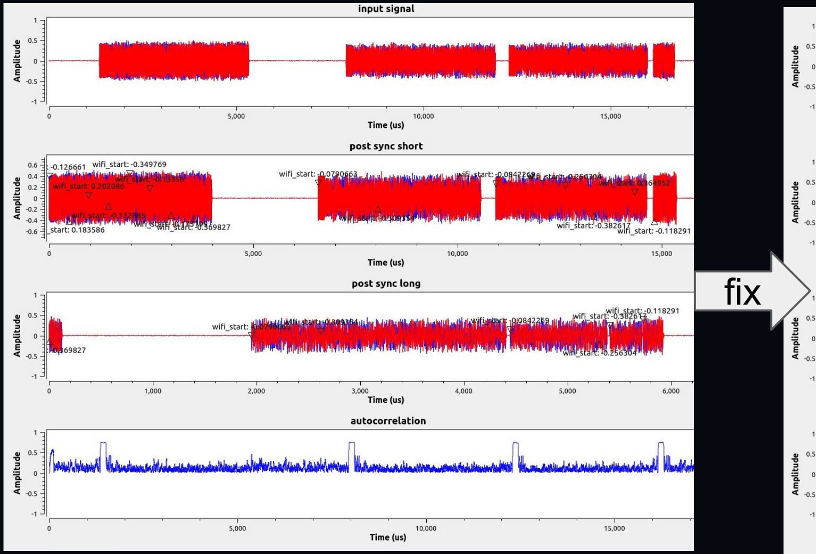

## 802.11a/g/p Long Training Field (LTF) Frame Timings

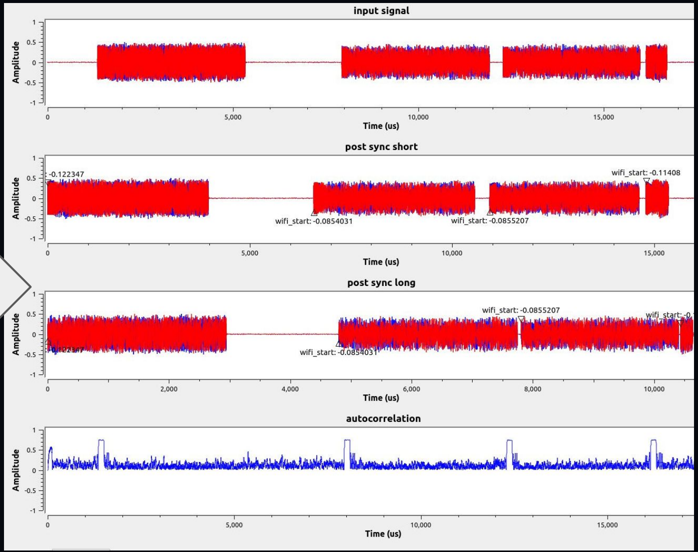
Without proper corrections to the timing in the short and long sync blocks, there was erratic tagging on HaLow packets. The tags did not correspond with the beginning of a HaLow packet.

## HaLow Long Training Field (LTF) Frame Timings

Creating variables for the number of samples per OFDM symbol, number of samples per guard interval, and duration of the STF and LTF (in samples) led to placement of tags at the correct beginning of the HaLow packet

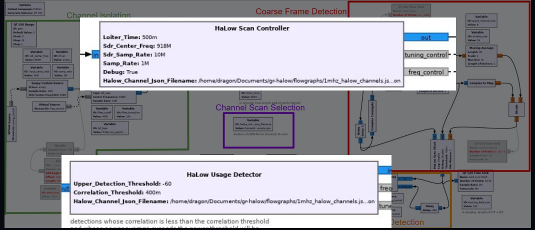
However, on neighboring channels there were still some false alarms so I needed a way of filtering those out as much as possible



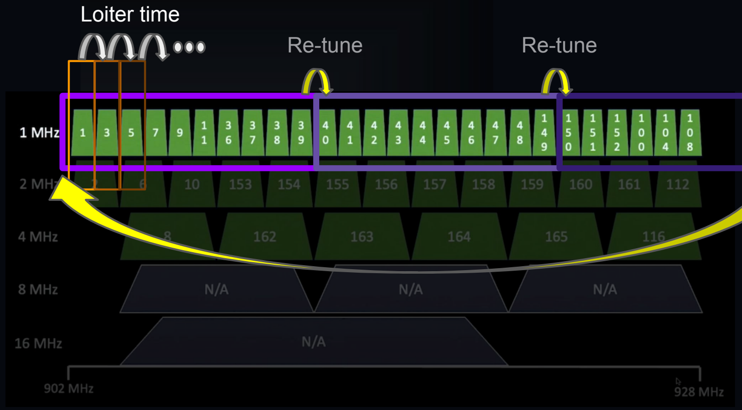

Everything is based off of the IEEE 802.11-2020 MAC and PHY Specification

# HaLow Additions



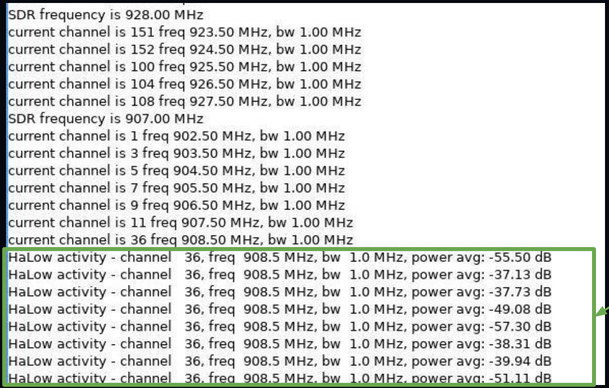

## HaLow Scan Control

The SDR bandwidth should be at least 1 MHz to capture a HaLow channel, but the scan controller adjusts based off of bandwidth available to the SDR. The scan controller filters out the channels in the JSON configuration file that it is not able to scan based off of available bandwidth. Within each tuning window, the frequency translating FIR filter isolates each channel for activity analysis. The loiter time determines how long the scan controller lingers on each channel.

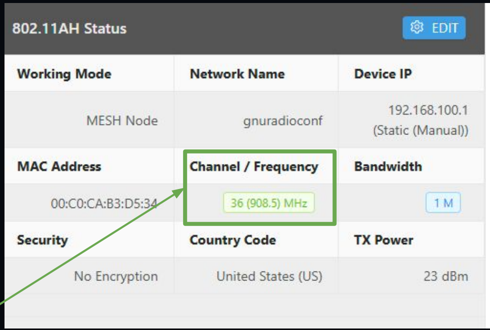



## HaLow Activity Detector

To reduce false alarms, the HaLow activity detector accepts a **threshold for the frequency offset** and the **average power level**. If the incoming tag value is below the threshold for the frequency offset and the next 10 samples average power exceeds the power threshold, a detection is printed to the screen.

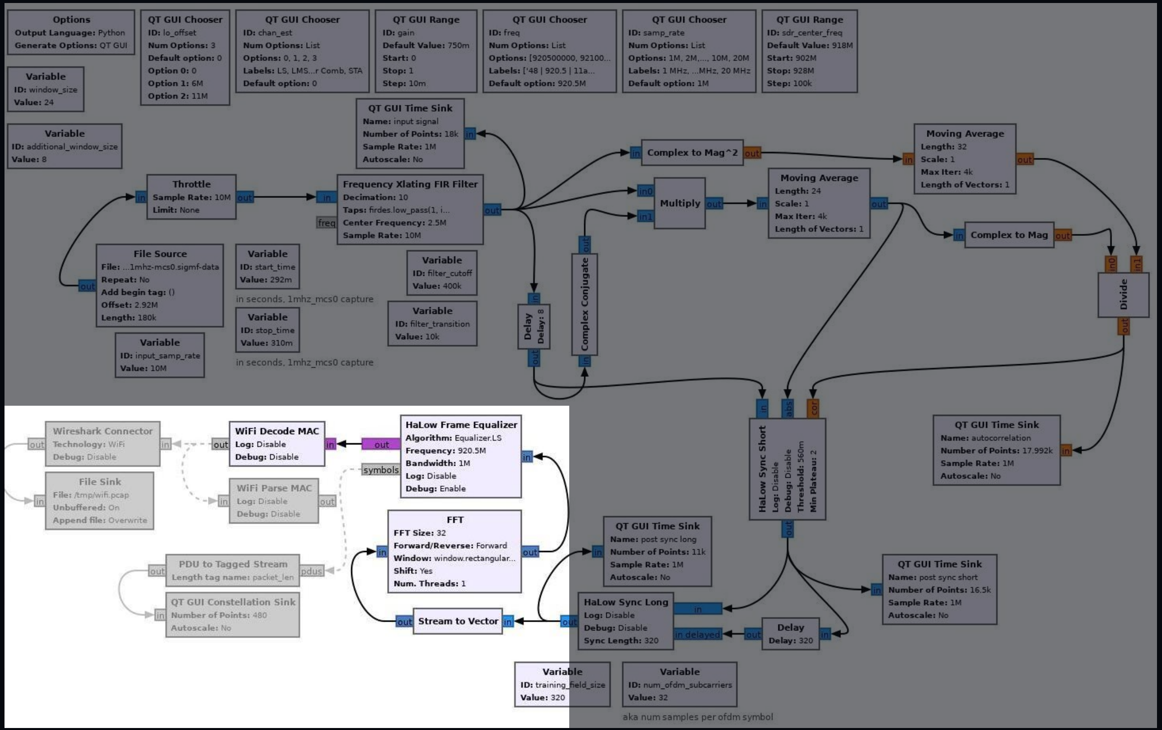



GNU Radio Console (Airspy R2)




Alfa HaLow-U Configuration Webpage


18m - 500ms Loiter
HaLow Detected



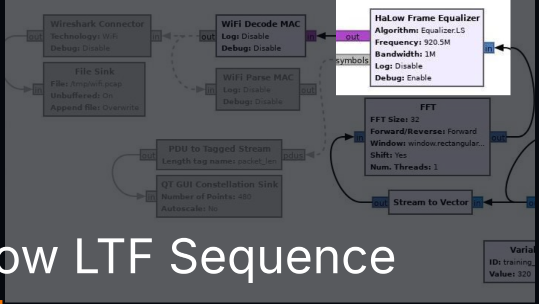

GNU RADIO VERSION 3.10.10.0

# HaLow Full Receiver

Implemented, not working -Determines if there is HaLow activity, and if there is, what channel it is on

irongiant33 on GitHub: gr-halow

# HaLow Modifications




## 802.11a LTF Sequence

The LTF sequence is used in 802.11a/g for least squares equalization; i.e., correcting for the difference in the channel effects across all 64 subcarriers.

## HaLow LTF Sequence

HaLow uses an identical process for equalization, but since there are 32 subcarriers, the LTF sequence is shorter.

A long OFDM training symbol consists of 53 subcarriers (including the value 0 at dc), which are modulated by the elements of the sequence L, given by

$$L_{-26,26} = \{1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 0, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, 1\} \quad (17\text{-}8)$$

A long OFDM training symbol shall be generated according to the following equation:

$$r_{LONG}(t) = w_{TLONG}(t) \sum_{k=-N_{ST}/2}^{N_{ST}/2} L_k \exp(j2\pi k \Delta_F (t - T_{G12})) \quad (17\text{-}9)$$

802.11ah 1 MHz LTF

$$r^{(i_{TX})}_{LTF1\_1,2}(t) = \frac{1}{\sqrt{N^{Tone}_{LTF} N_{STS}}} w_{T_{LTF1}}(t) \sum_{k=-N_{SR}}^{N_{SR}} \sum_{m=1}^{N_{STS}} \left( [Q_k]_{i_{TX},m} \Upsilon_{k,BW} [A^k_{LTF}]_{m,1} LTF_k \cdot \exp\left(j2\pi k \Delta_F (t - T_{GI2} - T_{CS}(m))\right)\right) \quad (23\text{-}38)$$

$$LTF_{-16:15} = \{0, 0, 0, 1, -1, 1, -1, -1, 1, -1, 1, 1, -1, 1, 1, 1, 0, -1, -1, -1, 1, -1, -1, -1, 1, -1, 1, 1, 1, -1, 0, 0\}$$

- **N_ST** - total number of subcarriers
- **N_SR** - highest data subcarrier index per OFDM symbol
- **Delta_F** - subcarrier frequency spacing
- **T_GI2** - Training symbol guard interval duration

## Implemented:

## iFFT for 1 MHz

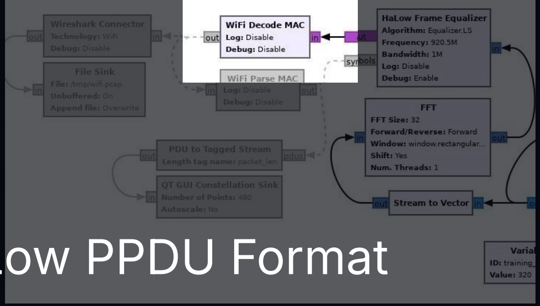
## LTF Sequence

Applying an inverse fourier transform to the shorter HaLow sequence should allow equalization to take place for HaLow; however, I have not been able to verify this.

A lot of the other values in the HaLow LTF equation have to do with multiple spatial time streams, but it made it simpler that the Alfa HaLow-U only has one.

Everything is based off of the IEEE 802.11-2020 MAC and PHY Specification

# HaLow Modifications




## 802.11a PPDU Format

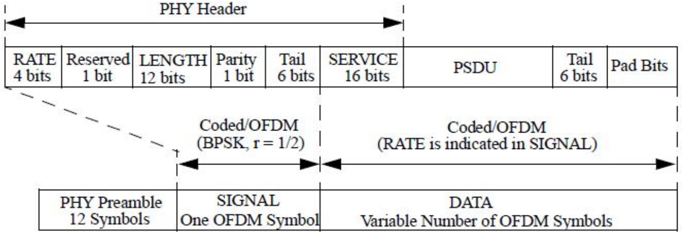



Figure 17-1—PPDU format

Table 17-6—Contents of the SIGNAL field

| R1–R4 | Rate (Mb/s) (20 MHz channel spacing) | Rate (Mb/s) (10 MHz channel spacing) | Rate (Mb/s) (5 MHz channel spacing) |
|---|---|---|---|
| 1101 | 6 | 3 | 1.5 |
| 1111 | 9 | 4.5 | 2.25 |
| 0101 | 12 | 6 | 3 |
| 0111 | 18 | 9 | 4.5 |
| 1001 | 24 | 12 | 6 |
| 1011 | 36 | 18 | 9 |
| 0001 | 48 | 24 | 12 |
| 0011 | 54 | 27 | 13.5 |

## HaLow PPDU Format

The general structure for S1G_1M is defined as in Figure 23-3. This frame format is used for S1G_1M PPDU SU transmission.

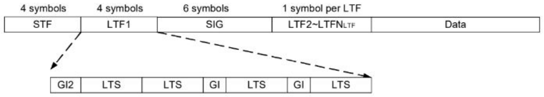



Figure 23-3—S1G_1M format

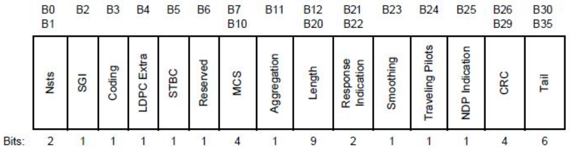



Figure 23-16—Structure of the 6 symbol SIG field of S1G_1M PPDU

Table 23-41—S1G-MCSs for 1 MHz, $N_{ss} = 1$

| MCS Idx | Mod | R | $N_{BPSCS}$ | $N_{SD}$ | $N_{SP}$ | $N_{CBPS}$ | $N_{DBPS}$ | $N_{ES}$ | Data_rate (kb/s) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | 8 μs GI | 4 μs GI |
| 0 | BPSK | 1/2 | 1 | 24 | 2 | 24 | 12 | 1 | 300.0 | 333.3 |
| 1 | QPSK | 1/2 | 2 | 24 | 2 | 48 | 24 | 1 | 600.0 | 666.7 |
| 2 | QPSK | 3/4 | 2 | 24 | 2 | 48 | 36 | 1 | 900.0 | 1000.0 |
| 3 | 16-QAM | 1/2 | 4 | 24 | 2 | 96 | 48 | 1 | 1200.0 | 1333.3 |
| 4 | 16-QAM | 3/4 | 4 | 24 | 2 | 96 | 72 | 1 | 1800.0 | 2000.0 |
| 5 | 64-QAM | 2/3 | 6 | 24 | 2 | 144 | 96 | 1 | 2400.0 | 2666.7 |
| 6 | 64-QAM | 3/4 | 6 | 24 | 2 | 144 | 108 | 1 | 2700.0 | 3000.0 |
| 7 | 64-QAM | 5/6 | 6 | 24 | 2 | 144 | 120 | 1 | 3000.0 | 3333.3 |
| 8 | 256-QAM | 3/4 | 8 | 24 | 2 | 192 | 144 | 1 | 3600.0 | 4000.0 |
| 9 | 256-QAM | 5/6 | 8 | 24 | 2 | 192 | 160 | 1 | 4000.0 | 4444.4 |
| 10 | BPSK | 1/2 with 2× repetition | 1 | 24 | 2 | 24 | 6 | 1 | 150.0 | 166.7 |

## Implemented: Function for HaLow 1 MHz PPDU Decoding

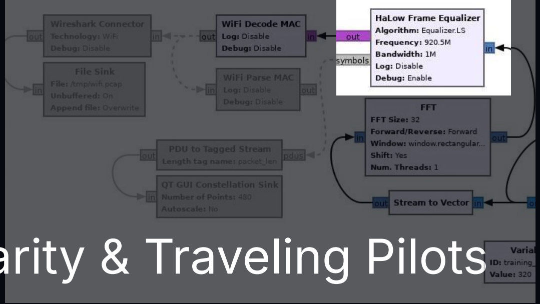
In the future, each PPDU should have its own object (i.e. legacy Wi-Fi, halow 1 MHz, HaLow 2 MHz Short Frame, HaLow 2 MHz Long Frame, etc.) that has its own decoder function and properties because they are all unique

My 1 MHz scope constraint helped me here because I only had to make the adjustment for the PPDU format shown on the right

Not pictured are the corrections for 2x repetition for MCS 10 and deinterleaving that I needed to add.

Everything is based off of the IEEE 802.11-2020 MAC and PHY Specification

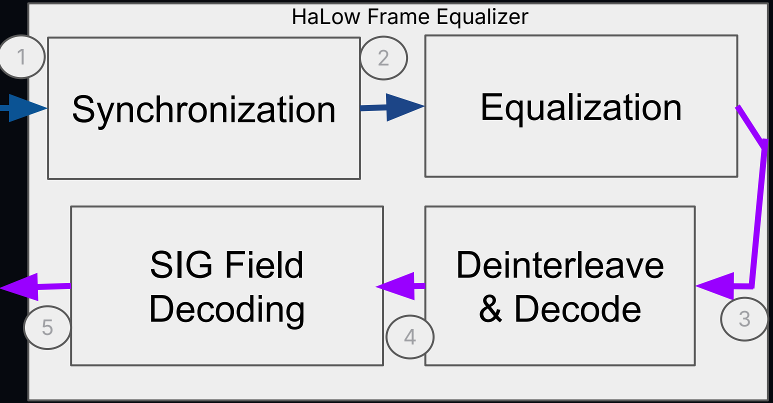
# HaLow Improvements




## Intermediate Testing

A lot is compressed into a single GNU Radio block, which makes it hard to modify and test to make sure everything right.

1. I assume the input to the frame equalizer is good given my success with the HaLow scanner
2. Unsure how to test the output of synchronization
3. Post equalization should yield interleaved bits, but they are LDPC encoded. How to verify equalizer?
4. LDPC decoded bits should be recognizable from PPDU format
5. Compare MCS value to the value it was set to




## Polarity & Traveling Pilots

Fixed pilot polarity corrections affect frequency offset correction, so that is possibly skewing my input to the equalizer. HaLow uses the same polarity values as 802.11ah

There is the possibility of traveling pilots in HaLow to better track changing channel conditions. I am currently assuming there are no traveling pilots, but this might be a bad assumption. I also have not been able to find a control on the Alfa HaLow-U that will guarantee no traveling pilots. The benefit of traveling pilots is their polarity is always positive.

Fixed Pilot EQN 23-50

$$P_n^{\{-7,7\}} = \{\Psi_{(n \bmod 2)+2}, \Psi_{((n+1) \bmod 2)+2}\}$$

$$P_n^{k \notin \{-7,7\}} = 0$$

Table 21-21—Pilot values for 80 MHz transmission

| $\Psi_0$ | $\Psi_1$ | $\Psi_2$ | $\Psi_3$ | $\Psi_4$ | $\Psi_5$ | $\Psi_6$ | $\Psi_7$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | –1 | –1 | 1 | 1 | 1 |

Traveling Pilot EQN 23-51

$$P_n^k = \begin{cases} 1.5 \times P_{n,\,\text{fix}}^{k_{\text{Pilot\_Fix}}^{(l)}}, & k \in K_{\text{Pilot\_Travel}}(n) \text{ and } k = K_{\text{Pilot\_Travel}}^{(l)}(n) \\ 0, & \text{otherwise} \end{cases}$$

Table 23-21—Traveling pilot positions for NSTS=1, 1 MHz S1G PPDU

| Pilot Index $l$ | Pattern Index $m$ | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 0 | –2 | –10 | –5 | –13 | –8 | –3 | –11 | –6 | –1 | –9 | –4 | –12 | –7 |
| 1 | 12 | 4 | 9 | 1 | 6 | 11 | 3 | 8 | 13 | 5 | 10 | 2 | 7 |

# Resources

- irongiant33/gr-halow: An open-source implementation of Wi-Fi HaLow, a.k.a. IEEE 802.11ah (github.com)
- IEEE MAC and PHY specification for 802.11ah (free to download): https://ieeexplore.ieee.org/document/9363693
- Google Drive Link for HaLow SigMF Files: https://drive.google.com/drive/folders/1DEHoJCMcezHTnvYwK1-GE37gLbkNcJ1I?usp=drive_link
    - Upload to IQEngine (https://staging.iqengine.org/upload) is pending
- DragonOS - helpful videos & tools for SDR hobbyists
- Troy Martin - 802.11ah Real-World Performance Results: Wi-Fi HaLow 802.11ah & Real-World Performance Results | Troy Martin | WLPC Phoenix 2024 (youtube.com)
- Andreas Spiess on HaLow: https://www.youtube.com/watch?v=rj9GZQtFs8k
- Ben Jeffery on HaLow: 802.11ah Wi-Fi HaLOW: The 1 Kilometer WiFi Standard (youtube.com)
- WiFi Certified HaLow Product Finder - Product Finder Results | Wi-Fi Alliance

# THANK YOU

FOR ATTENDING

@samiammiller

KD2MWK

irongiant33

# IEEE Standards Breakdown: https://ieeexplore.ieee.org/document/9363693

- Ch15 is 802.11 (dsss in 2.4 GHz, 1Mbps, 2Mbps rates) p.2749
- Ch16 is 802.11b (HT dsss in 2.4ghz 1, 2, 5.5, 11Mbps rates) p.2773
- Ch17 is 802.11a (ofdm in 5ghz, 5MHz, 10MHz, 20MHz channel widths 6,9,12,18,24,36,48,54 Mbps rates in 10MHz channel) p.2802
    - P.2807 PPDU format
    - P.2810-2811 I'm pretty sure this is 802.11a modulation dependent parameters and timing parameters. It looks similar to HaLow at just 10x the rates
- Ch18 is 802.11b-corrigendum1 (extended rate phy "ERP" dsss in 2.4ghz, backwards compatible with 802.11a/b) p 2848
- Ch19 is 802.11g (ofdm in 2.4ghz, but also backwards compatible with ch18 dsss in 2.4ghz and ch17 ofdm in 5ghz) p.2860
    - P.2862 - definition of non-ht, ht-mf, ht-gf. Pretty sure gr-ieee80211 only supports non-ht. Support for ch17/18 packets. Mixed format (MF) has preamble that can be decided by ch17/18 but data that cannot. Greenfield (GF) cannot be recognized at all by ch17/18.
    - P.2873 PPDU format. Makes sense for delay of 16us because STF and LTF are each 8us.
    - P.2880 timing parameters. 48 complex data numbers. 52 sub carriers, highest sun carrier index is 26. 312.25 khz subcarrier spacing
- Ch20 is directional multi gig, 802.11ad? p.2962
- Ch21 is very high throughput, 802.11ac? p.3010
- Ch22 is television very high throughput 802.11af? p.3137
- Ch23 is 802.11ah (ofdm in S1G, essentially 802.11g knocked down 10x) p.3186

# Commercial Device Breakdown

Morse Micro
- MM6108-MF08651-US ($30 - PCB Module): https://www.mouser.com/ProductDetail/Morse-Micro/MM6108-MF08651-US?qs=mELouGlnn3dMCf9rE7Pbkw%3D%3D
- MM6108-EKH03-05US-E ($250 - Router): https://www.mouser.com/ProductDetail/Morse-Micro/MM6108-EKH03-05US-E?qs=mELouGlnn3fleiQYzZodAg%3D%3D
- MM6108-EKH01 ($500 - Development Kit): https://www.mouser.com/c/?marcom=169968848

Newracom
- Alfa HaLow-U ($125): https://store.rokland.com/products/alfa-network-halow-u-802-11ah-halow-usb-adapter-support-ap-client-mode
- Alfa AHPI7292S Raspberry Pi Hat ($65): https://store.rokland.com/products/alfa-network-ahpi7292s-ieee-802-11ah-sub-1-ghz-module-in-raspberry-pi-hat-form-factor
- Teledatics XPAH ($99): https://teledatics.io/collections/all
- Teledatics HaloMax ($109): https://www.crowdsupply.com/teledatics-incorporated/halomax-tm-long-range-wireless

Taixin
- LilyGo T-HaLow ($30): https://www.lilygo.cc/products/t-halow
- Other products on AliExpress:
  - ($70) 1.2KM Wireless Long Distance WIFI AP Transmitter Sender Receiver For 4MP 5MP 8MP IP PTZ Camera Ethernet Equipment - AliExpress 30