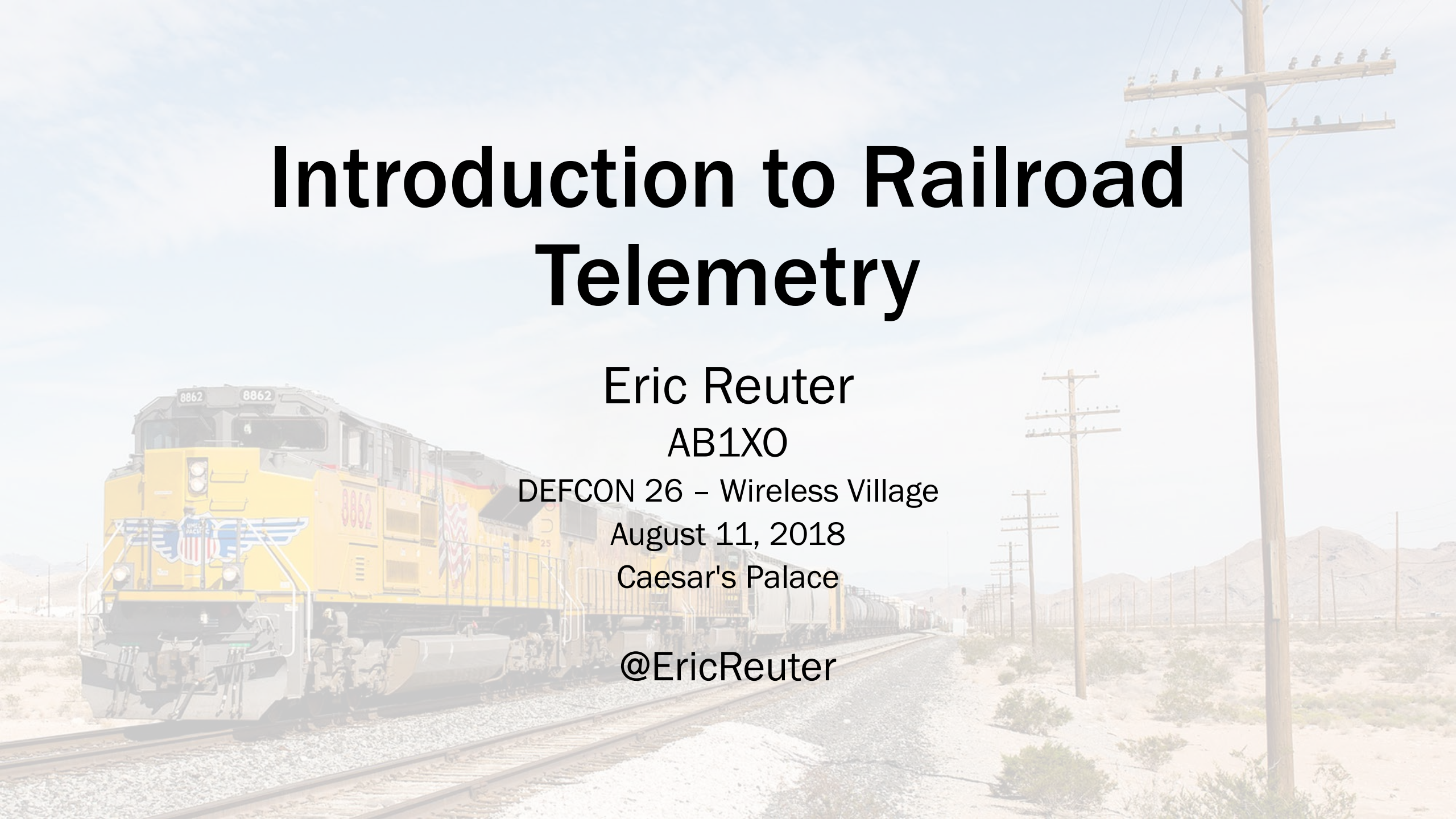
Introduction to Railroad Telemetry

Eric Reuter
AB1XO

DEFCON 26 – Wireless Village
August 11, 2018
Caesar's Palace

@EricReuter



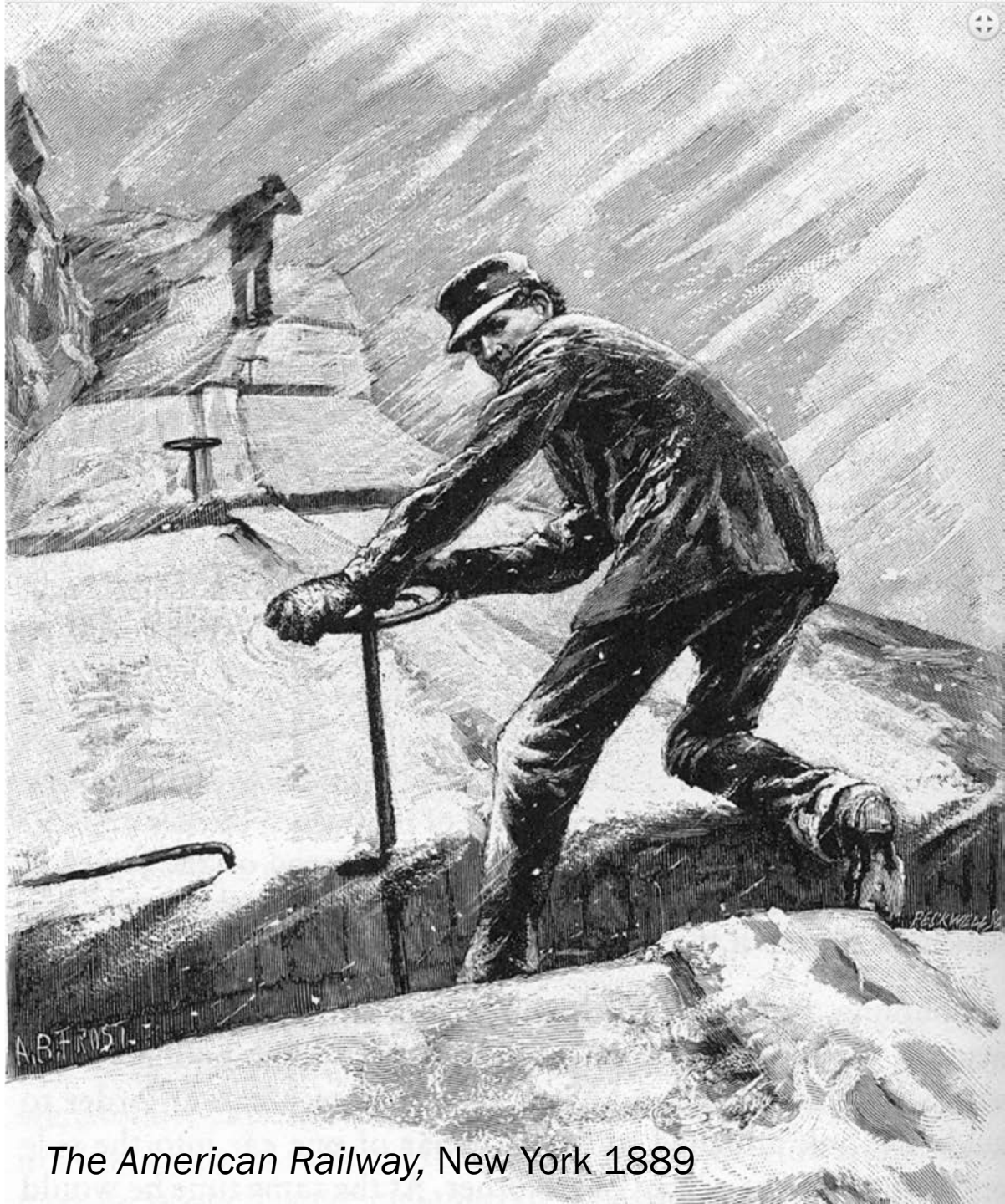
Railroad RF Applications (North America)

• Voice communication (analog & digital)	159-161 MHz
• Positive Train Control (PTC)	220-222 MHz
• Advanced Train Control System (ATCS)	896-928 MHz
• Distributed Power control (DP)	452.9 MHz
• End-of-Train/Head-of-Train telemetry (EOT)	452.9/457.9 MHz
• Automatic Equipment Identification (AEI)	902-921 MHz

End-of-Train Device (EOT)

Railroad Brakes

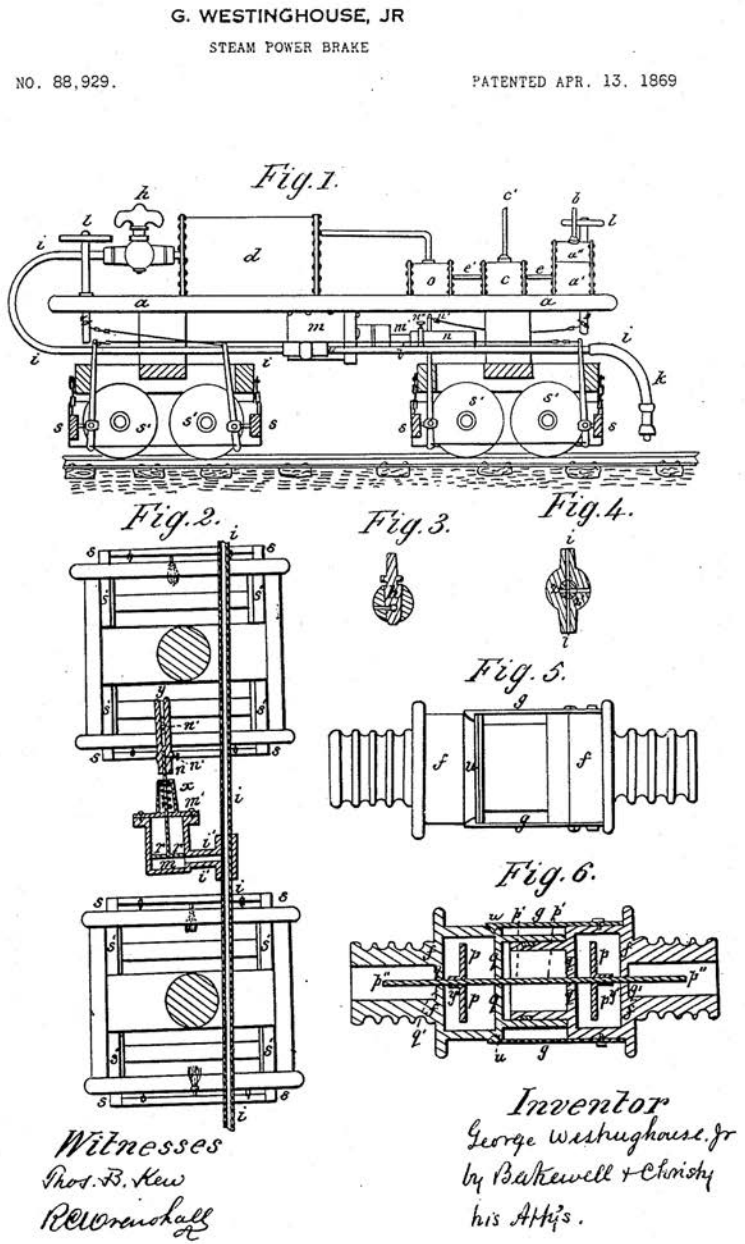
- Momentum of railroad cars preclude braking a train with the engine alone.
- In the early days, brakemen ran along the roofs of the cars, applying individual handbrakes.



The American Railway, New York 1889

Westinghouse Air Brake

- Patented by George Westinghouse in 1868 (Age 22)
- Widely adopted in US by 1900
- Compressed air from the locomotive is distributed to cars through continuous brake pipe
- Failsafe – brakes applied when air is reduced or vented
- Modern brakes substantially similar


G. WESTINGHOUSE, JR

STEAM POWER BRAKE

NO. 88,929. PATENTED APR. 13. 1869

Fig. 1.

Fig. 2. Fig. 3. Fig. 4.

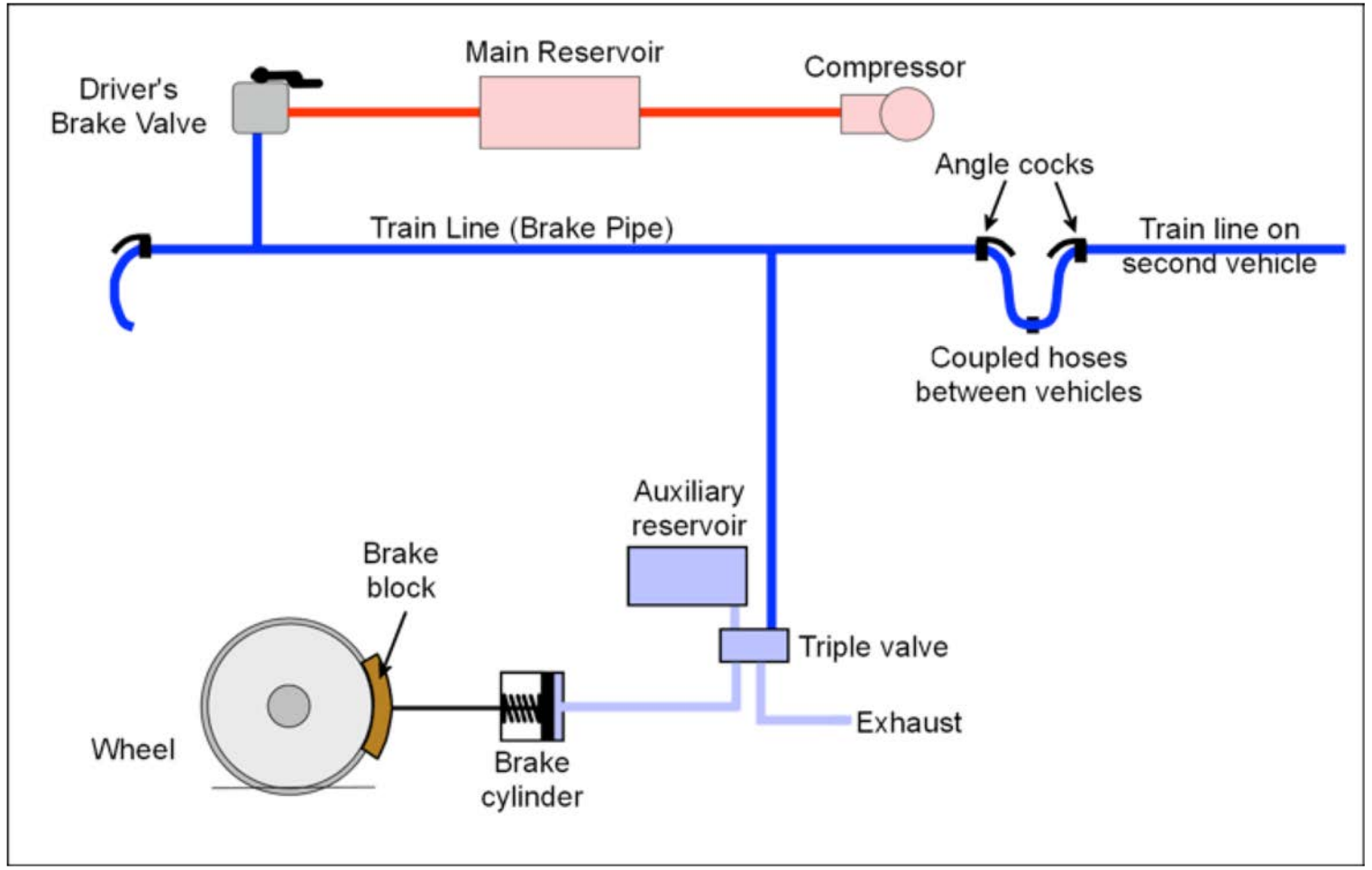
Fig. 5.

Fig. 6.

Witnesses
Thos. B. Kew
R. Wrenshall

Inventor
George Westinghouse Jr
by Bakewell & Christy
his Att'ys.

Westinghouse Air Brake



Courtesy www.railway-technical.com

Air Brake Operation

- **Release** – Reservoirs on cars are charged to 90 psi from brake pipe
- **Service Application** – Brake pipe pressure gradually reduced, reservoir air fed to brake cylinders
- **Emergency Application** – Brake pipe vented (“dumped”) to atmosphere, resulting in rapid application of brakes

Air Brake Operation

- Propagation rates
  - Service Application: 600-700 fps
  - Emergency Application: 900 fps

- Important for engineer to know the pressure at the rear of the train:
  - Has change of pressure reached the rear of the train?
  - Is the pipe disconnected somewhere?
  - Has the train pulled apart?

Monitoring Brake Pressure

1870 – 1980: Crew in caboose

Sean Lamb Photo – Creative Commons License

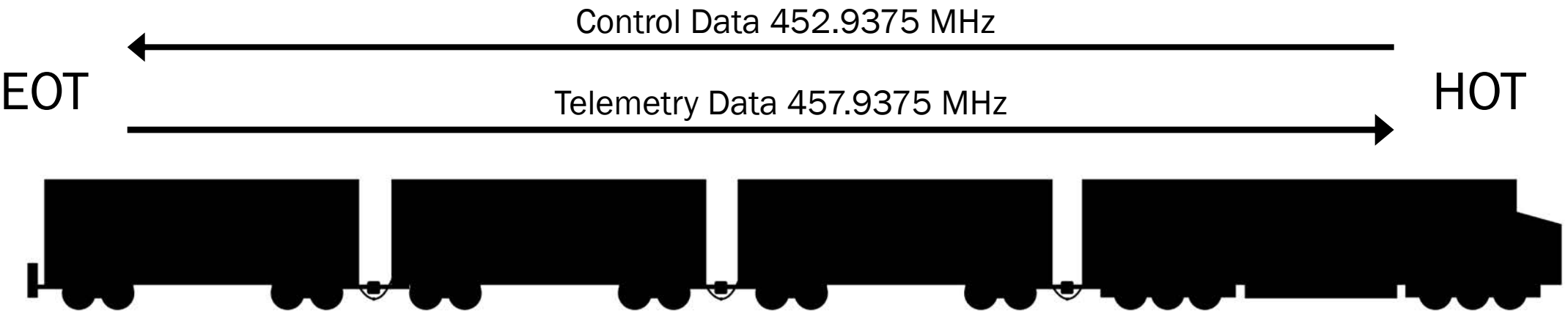
1980-Present: EOT

EOT/HOT

Two-part system:

- End-of-Train Device (EOT) attached to last car
- Head-of-Train Device (HOT) in locomotive

Modern units allow full-duplex two-way communication



EOT

Functions:

- Monitor brake pressure and relay data to Head-of-Train (HOT) device in locomotive (periodic or on request)
- Provide flashing red light, required for night operations
- On two-way systems, dump brake pipe in emergency brake applications (on request of HOT)

HOT

Commands:

- Status Request
- Emergency

EOT/HOT Specifications

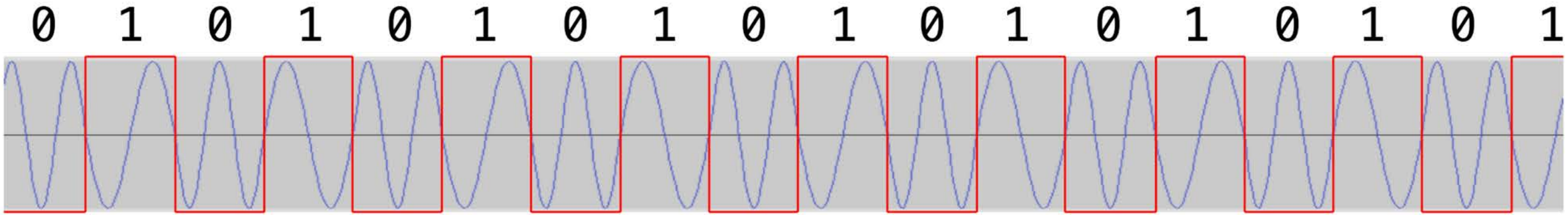
- Frequency: 457.9375/452.9375 MHz (full duplex)
- RF Power: 2 W/10+ W (variable)
- Peak Deviation: ± 3 kHz
- Modulation: Continuous Phase Fast Frequency Shift Keying (FFSK)
- Data Rate: 1200 baud
- Modulating Frequencies:

  Space: 1800 Hz

  Mark: 1200 Hz

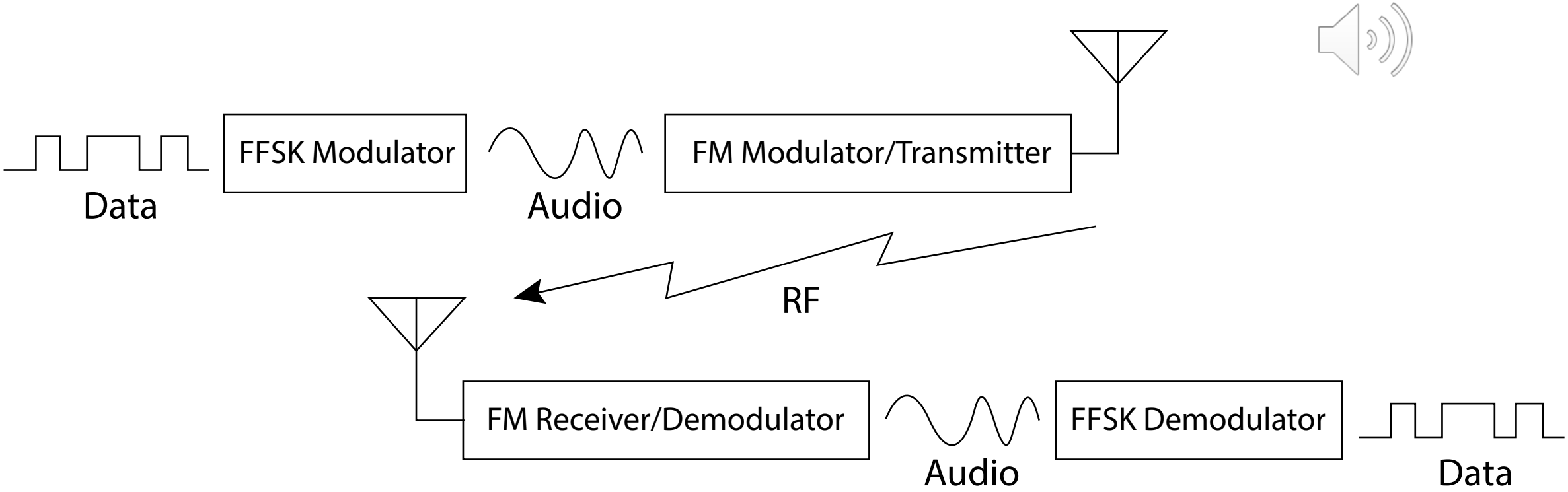
Fast Frequency Shift Keying (FFSK)
(a.k.a. MSK)

- Form of Audio Frequency Shift Keying (AFSK)
- Continuous phase
- Modulation Index = 0.5 ($f_{space} - f_{mark}$ = ½ bit rate)
- Mark (1200 Hz) = 1 cycle at 1200 baud
- Space (1800 Hz) – 1.5 cycles at 1200 baud



Fast Frequency Shift Keying (FFSK)

Two-stage Modulation & Demodulation



EOT

Message Format

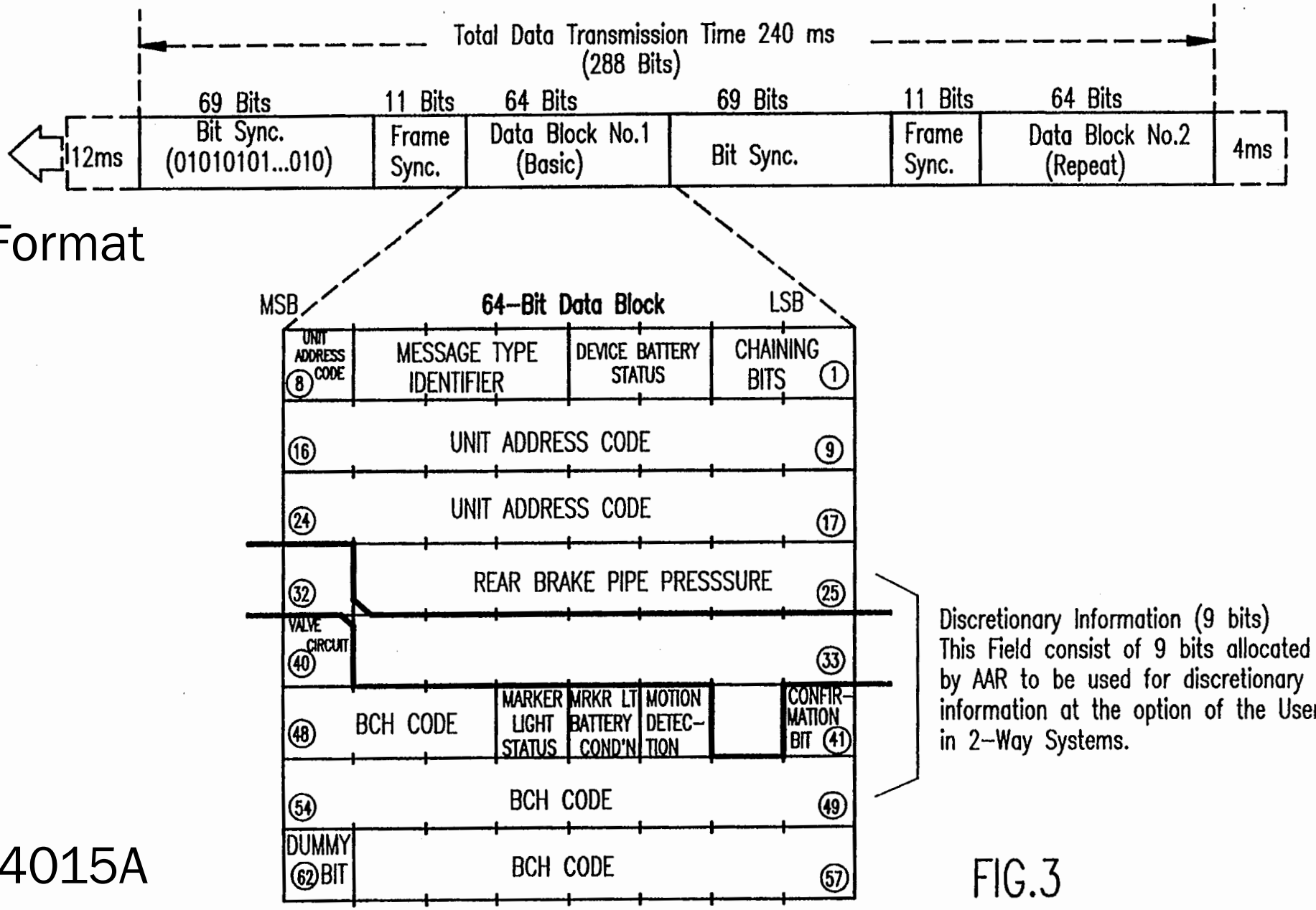


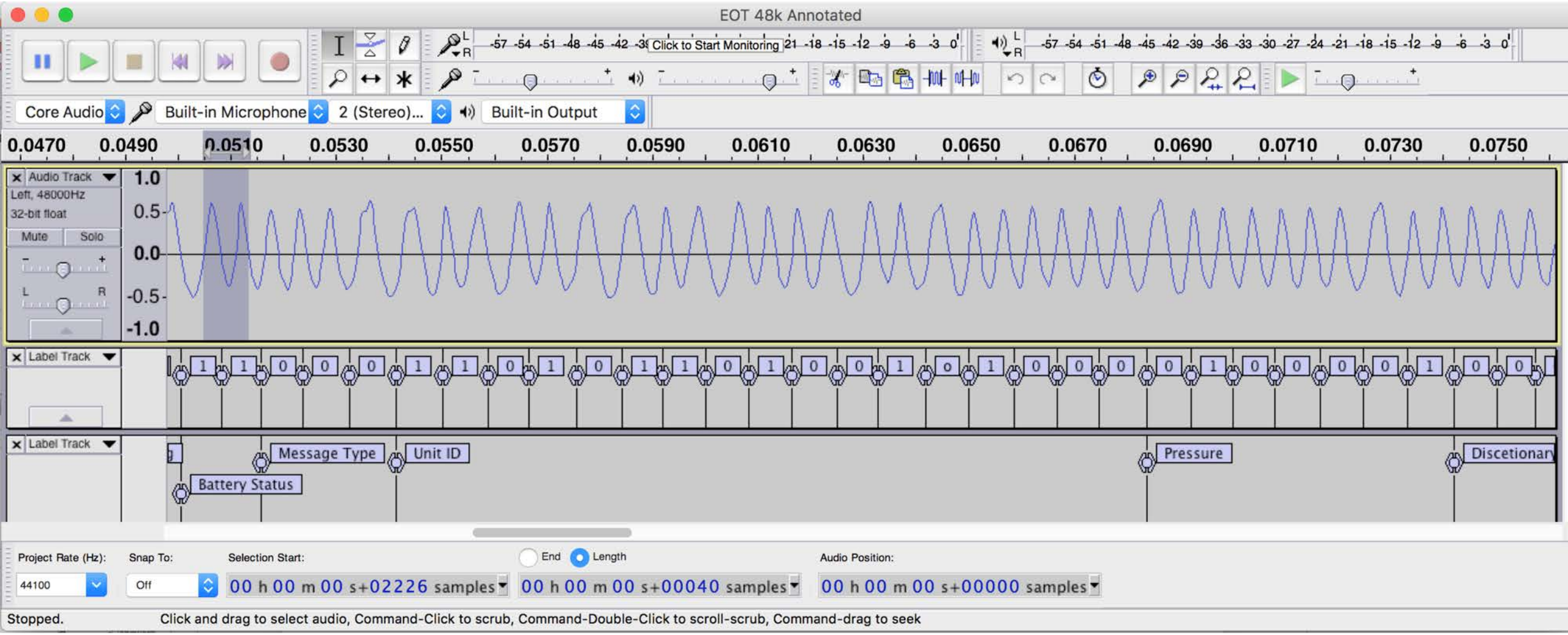
FIG.3

US Patent 5374015A

EOT Message Format

Bit Sync	69 bits	0101010101010101010101010101010…
Frame Sync	11 bits	01001000111
Chaining Bits	2 bits	11 for single block
Message Type	3 bits	000 or 111
Device Battery Cond	2 bits	Ok, weak, very weak, not monitored
Rear Unit Address	17 bits	Unique to each EOT (hard coded)
Brake Pipe Pressure	7 bits	0–125 psig
Battery Charge	7 bits	0–127
Discretionary	1 bits	
Valve Circuit Status	1 bit	0 = failed, 1 = operational
Confirmation Indicator	1 bit	1 when responding to HOT
Turbine Status	1 bit	
Motion Detection	1 bit	1 = car in motion
Marker Light Battery	1 bit	0 = OK or not monitored, 1 = weak
Marker Light Status	1 bit	0 = off, 1 = on
BCH Code	18 bits	BCH checkbits
Trailing Bit	1 bit	Always 1 (filler)

Decoding – First Attempt



Decoding – First Attempt

Little-endian

Unit ID

1101011010010100 0

Pressure

0100001

1000010 = **66 psig**

0001010010110101 1 = **10603**

BCH Code

- Cyclic multiple-error-correcting code (CRC)
- Codeword consists of data to be encoded concatenated with string of checkbits
- Example: (7, 4) code
  - Codeword length: 7 bits
  - Data length: 4 bits

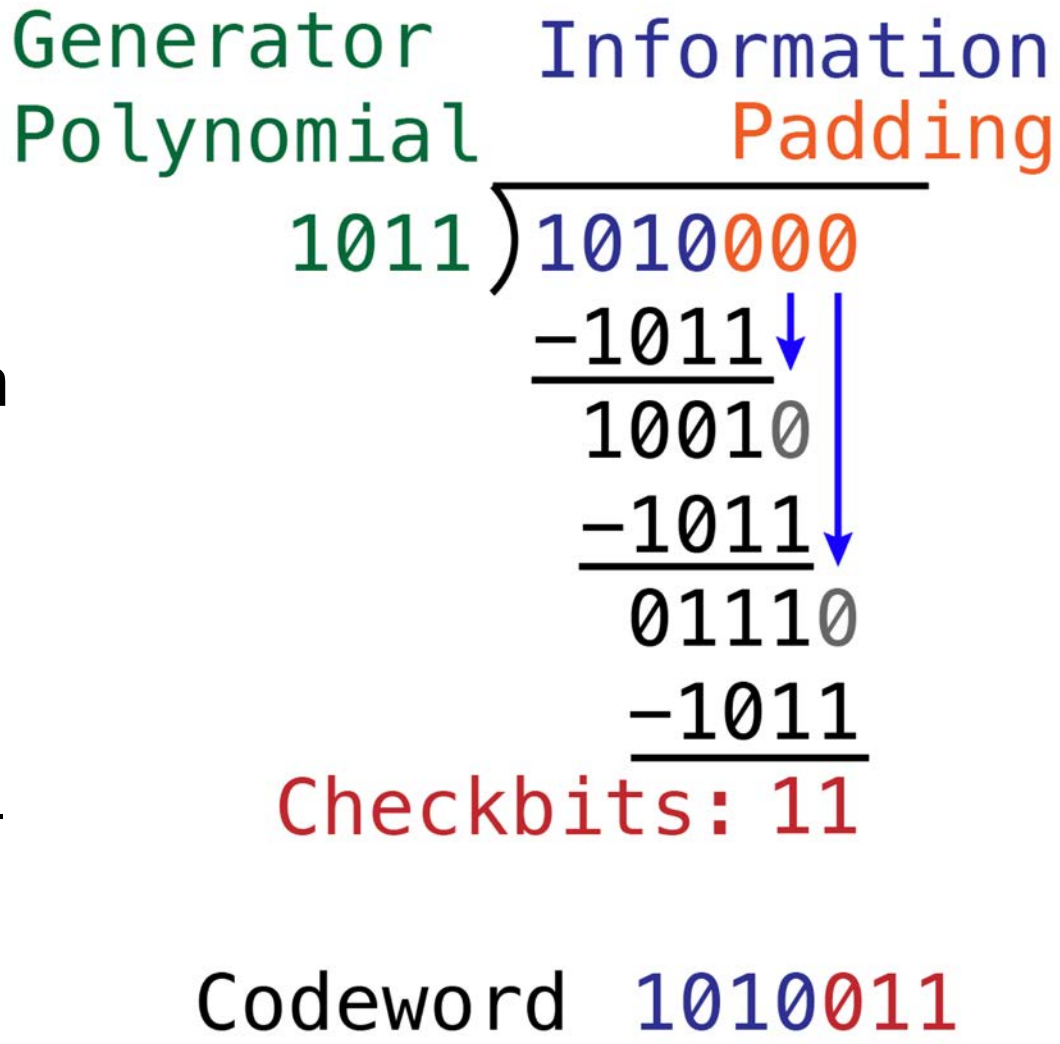
```
1 0 1 0 0 1 1
Data   Checkbits
```

BCH Code: Example

Generator Polynomial for (7, 4) code:

$g(x) = (x^3+x+1) = 1011$ in binary form

Information bits padded with zeros equal in number to the degree of the generator polynomial – 2^3 in this case.



BCH Code: Example

- Possible 7-bit words: $2^7 = 128$
- Valid codewords: $2^4 = 16$
- Invalid codewords: 112
- Can correct $t = 1$ error
- Minimum Hamming Distance: $d \geq 2t + 1$

0000000
0001011
0010110
0011101
0100111
0101100
0110001
0111010
1000101
1001110
1010011
1011000
1100010
1101001
1110100
1111111

0001011
0001100
0001101
0010110
0010111
0011000
0011001
0011010
0011011
0011100
0011101
0011110
0011111
0100001
0100010
0100011
0100100
0100101
0100111
0101000
0101001
0101010
0101011
0101100

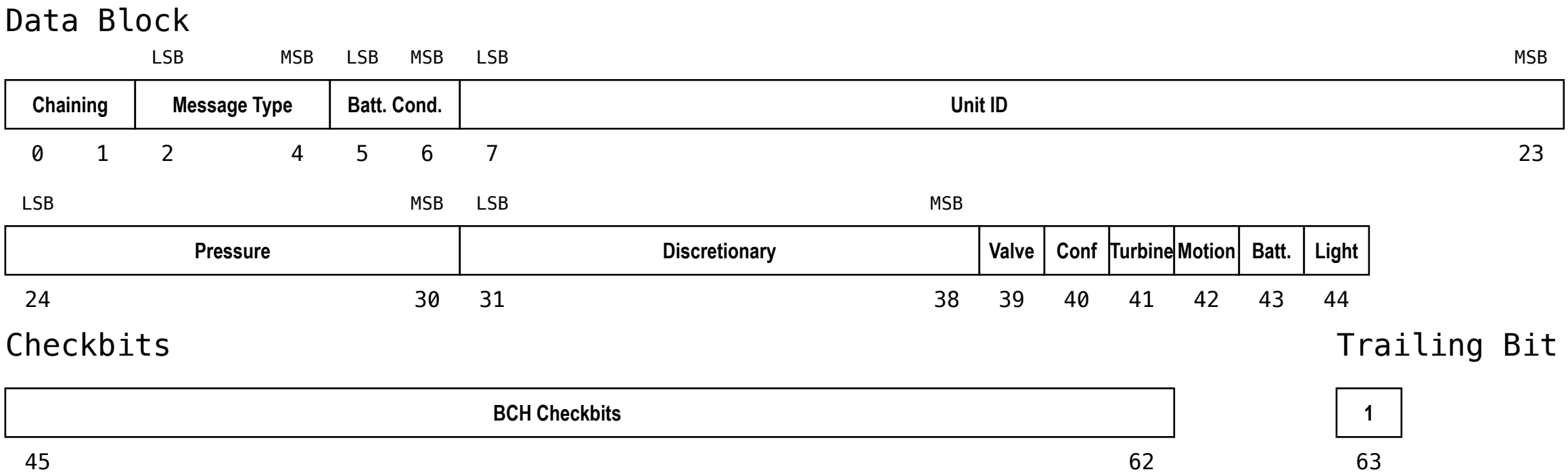
Packet Validation

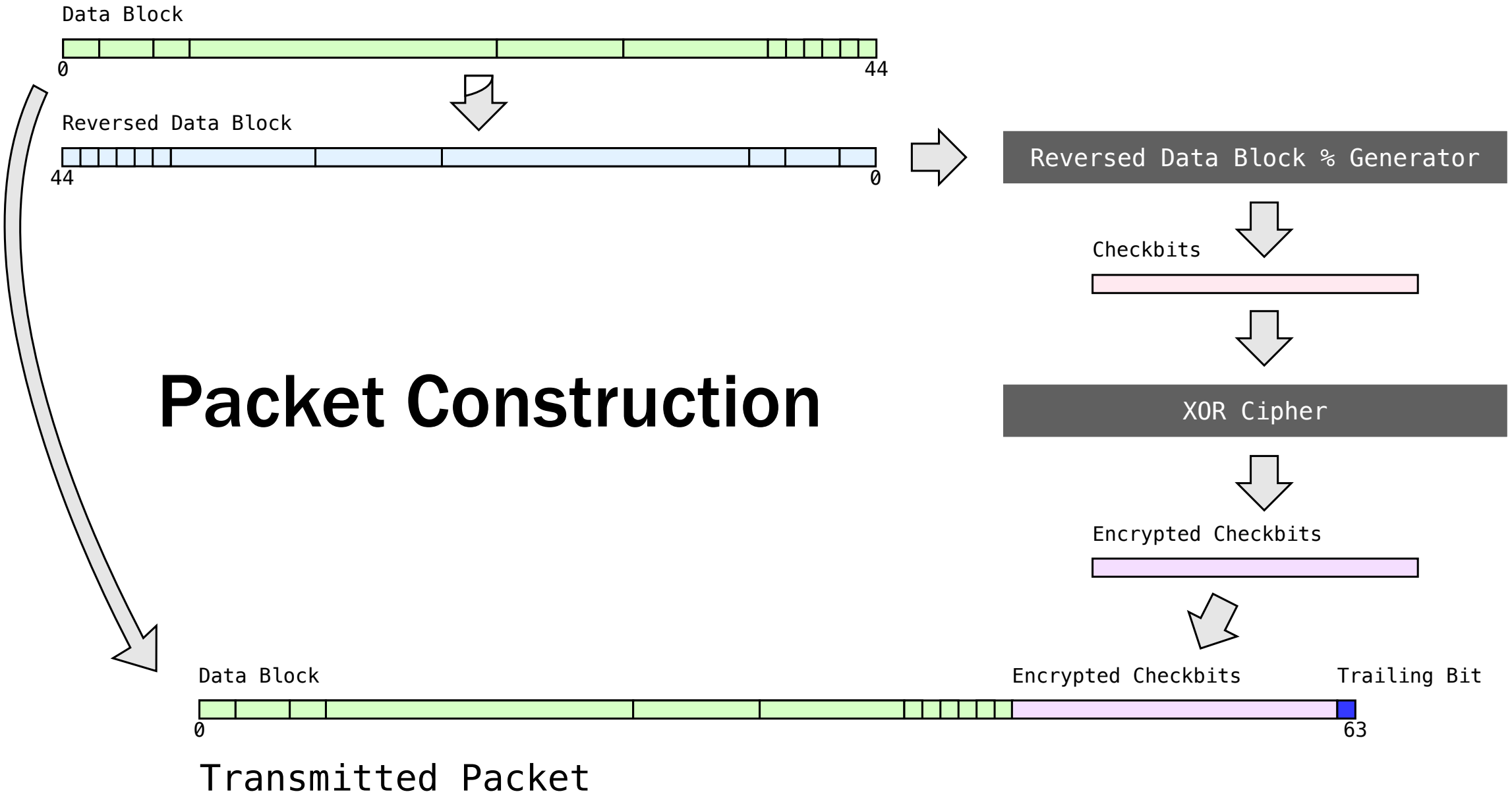
- Option 1: Decode BCH code to correct errors (harder)
- Option 2: Recalculate BCH checkbits and compare to those received (easier, but no error correction)

BCH Code

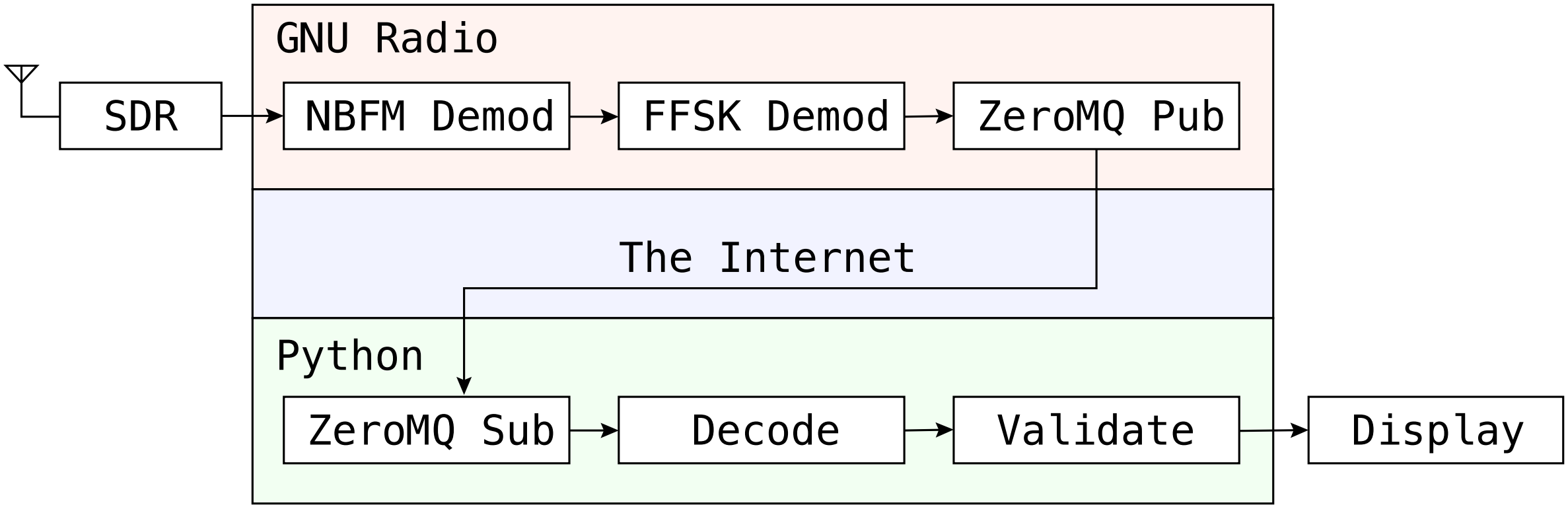
BCH code is of (63, 45) Type

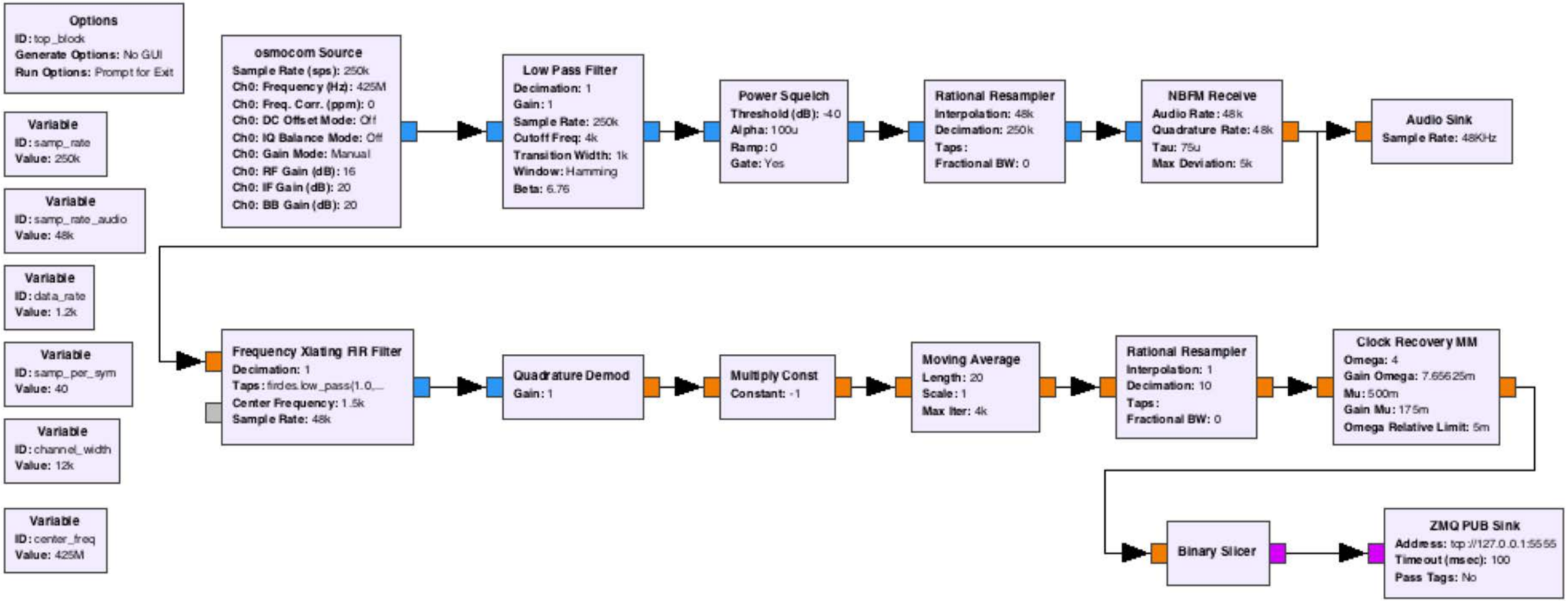
$g(x) = (x^{18}+x^{17}+x^{16}+x^{15}+x^{9}+x^{7}+x^{6}+x^{3}+x^{2}+x+1) = 111100000101100111$

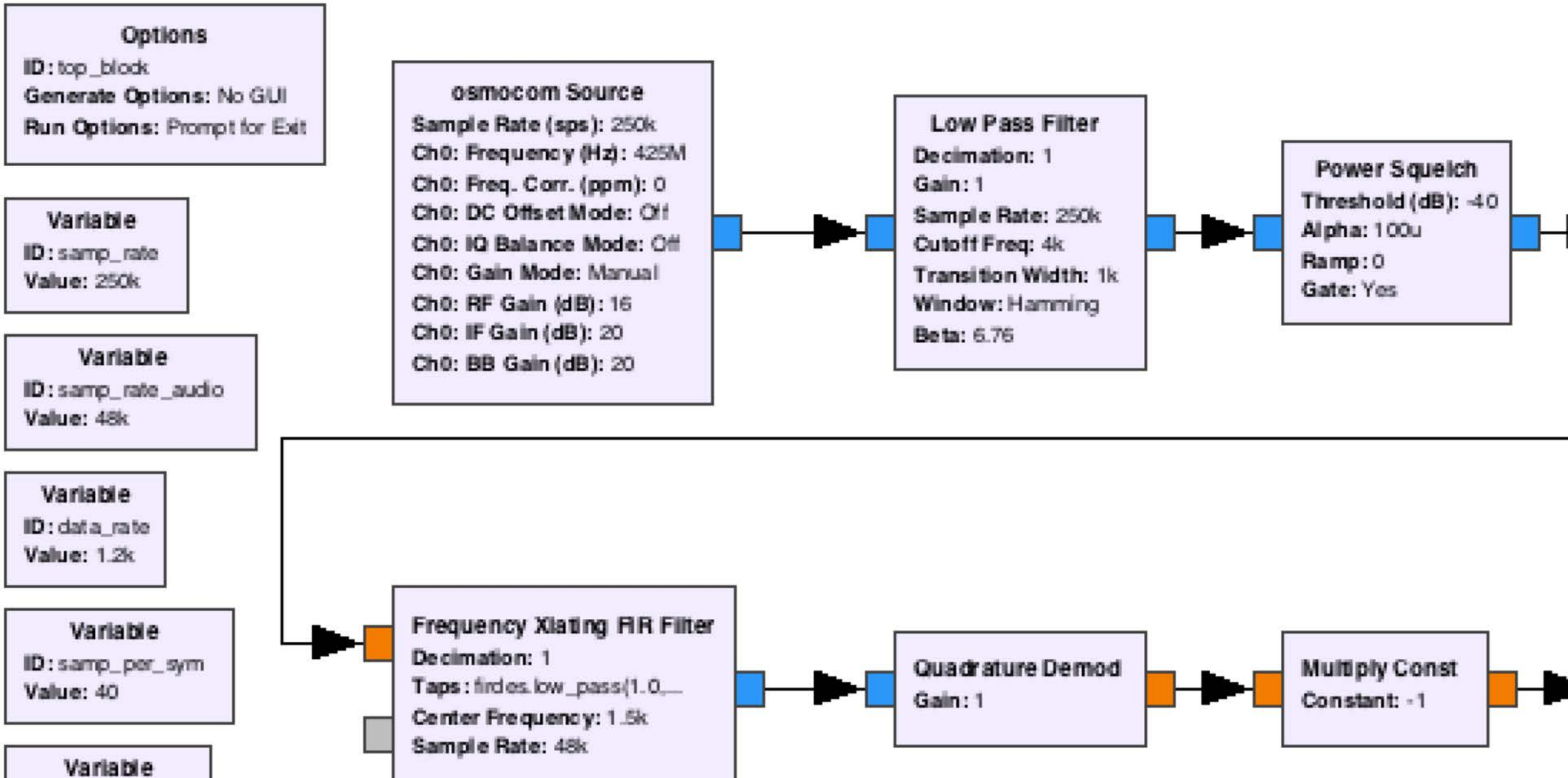


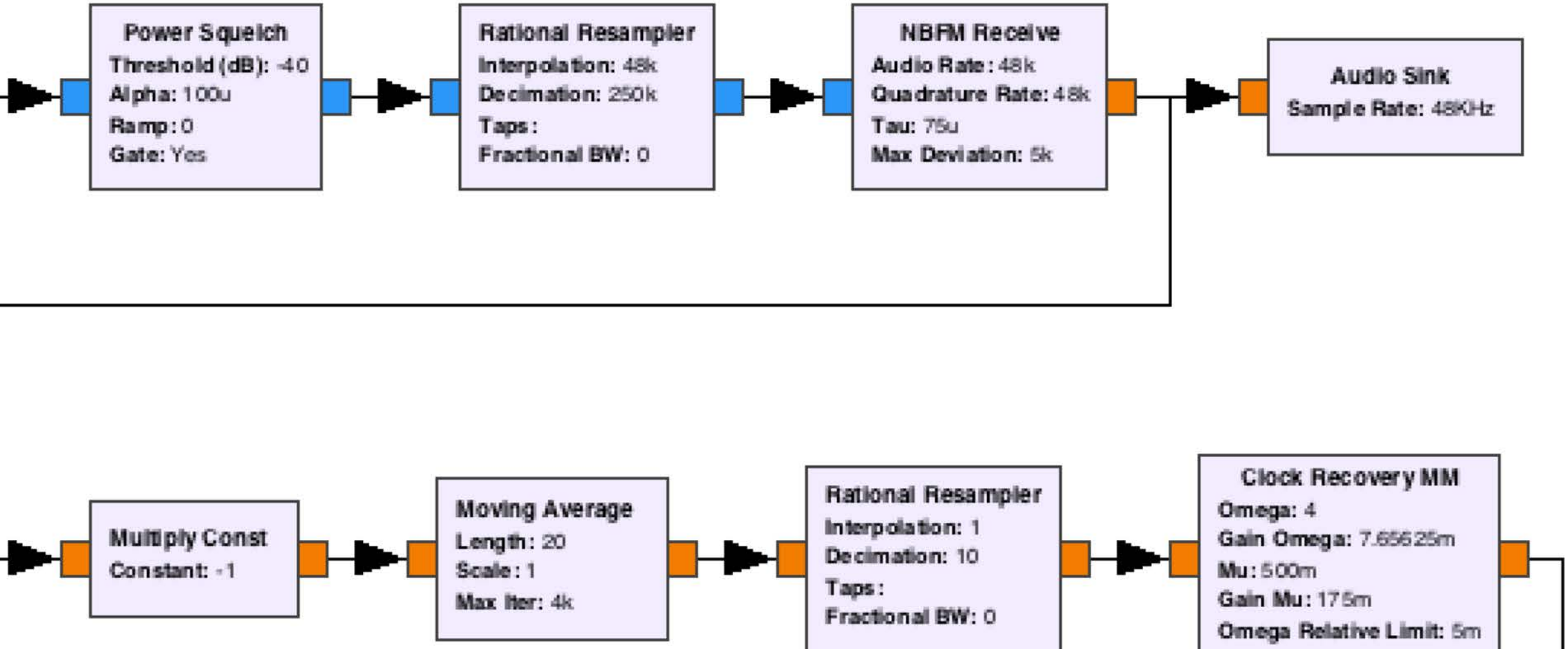

Data Block
0
44
Reversed Data Block
44
0
Reversed Data Block % Generator
Checkbits
XOR Cipher
Encrypted Checkbits
Packet Construction
Data Block
Encrypted Checkbits
Trailing Bit
0
63
Transmitted Packet

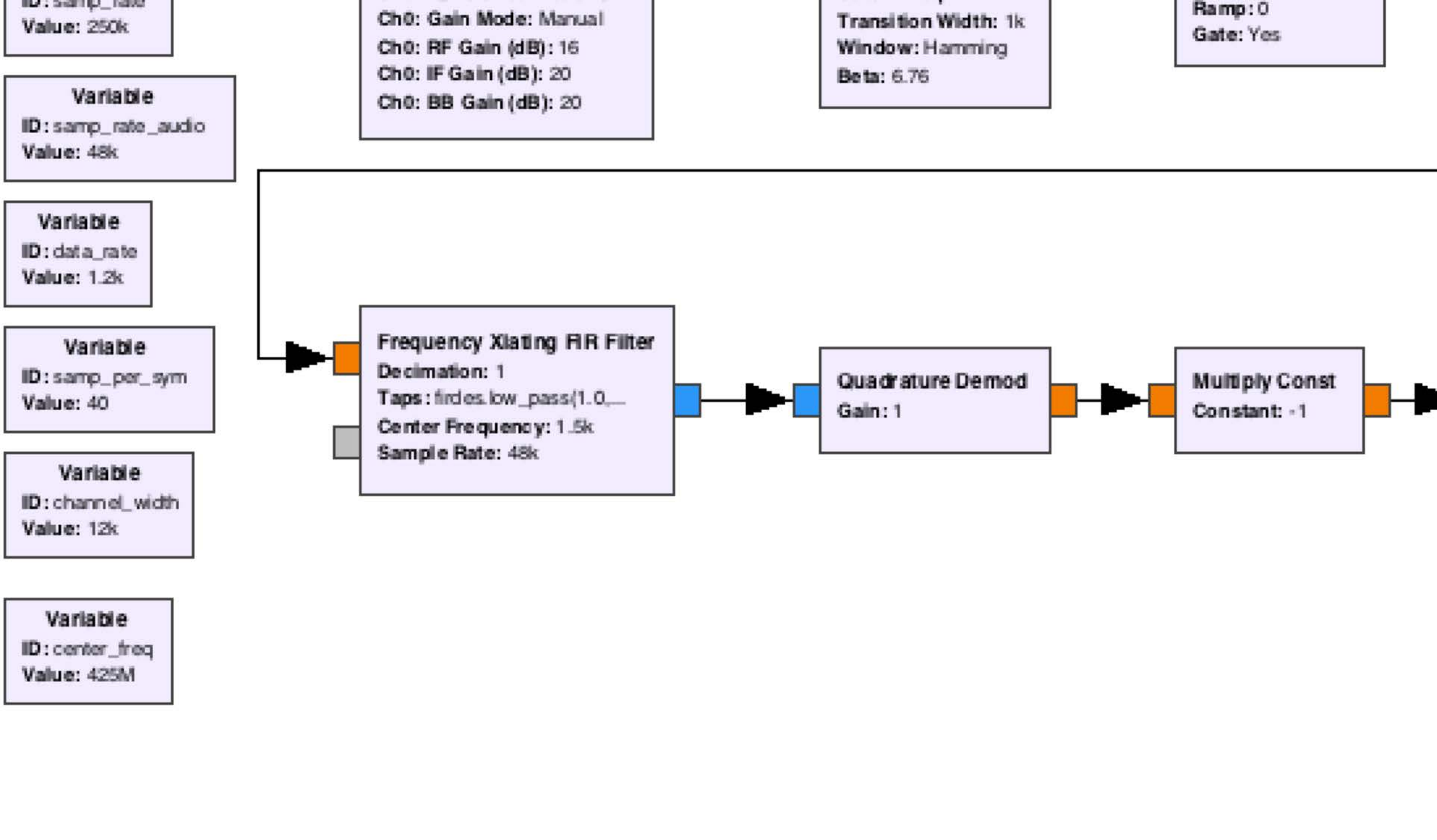
EOT Receiver/Decoder

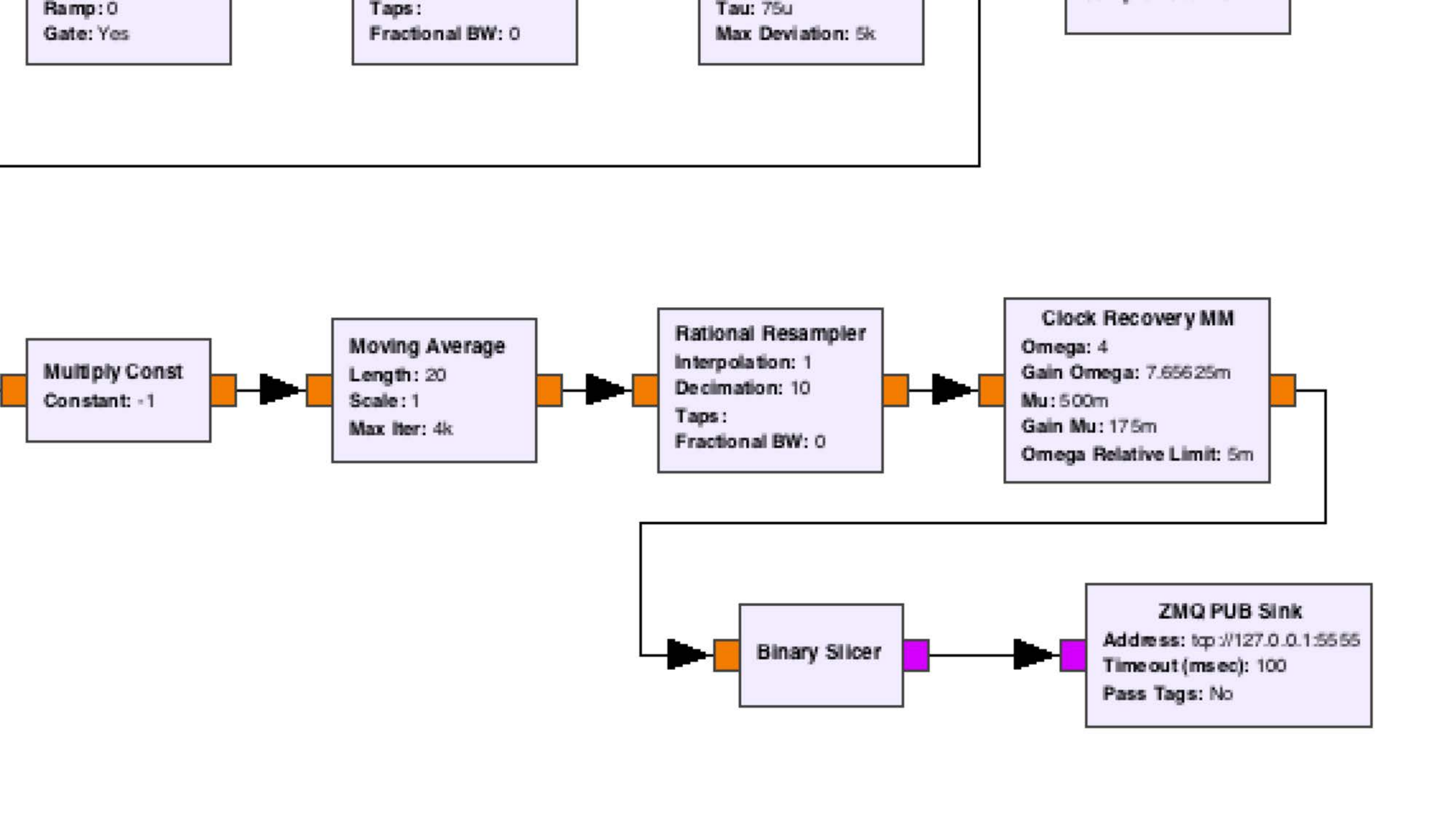


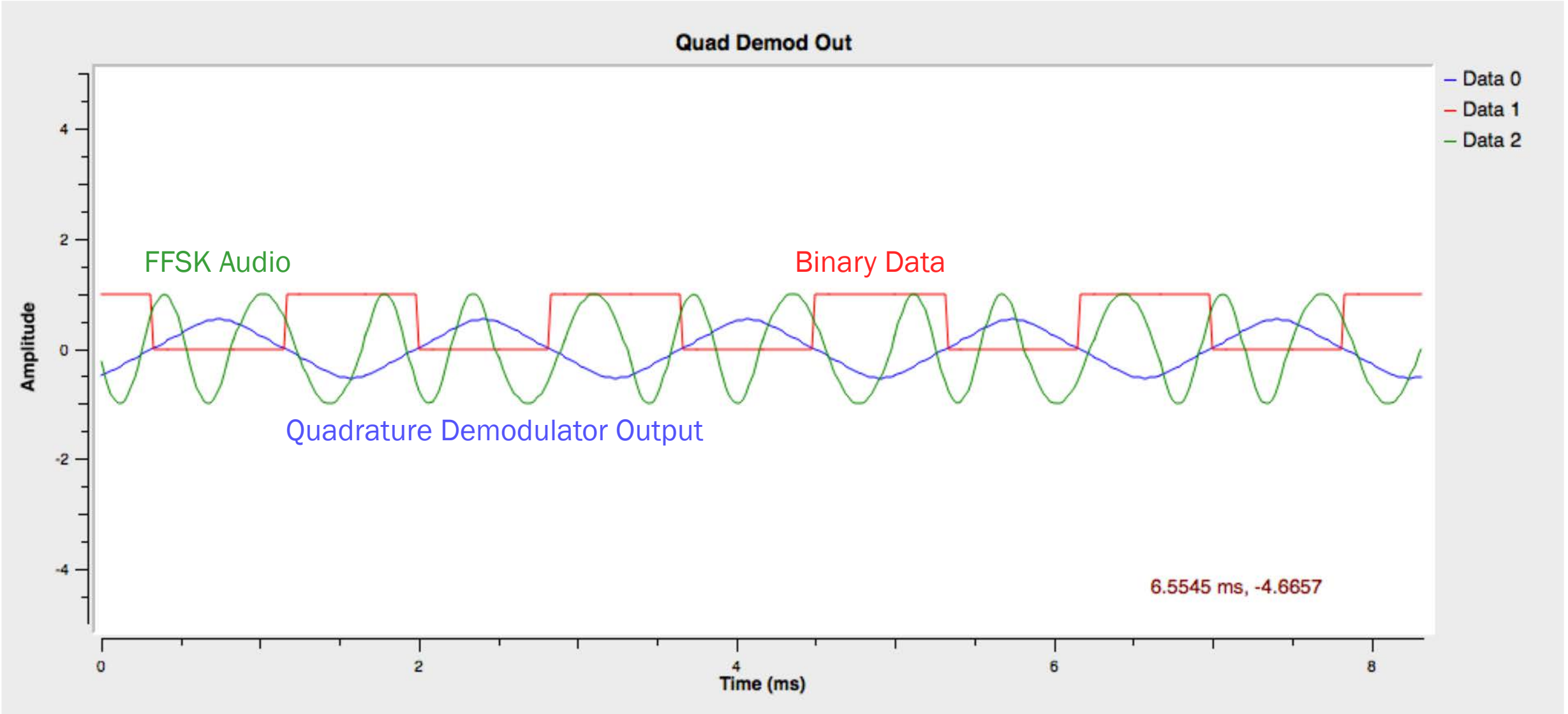

Options
ID: top_block
Generate Options: No GUI
Run Options: Prompt for Exit
Variable
ID: samp_rate
Value: 250k
Variable
ID: samp_rate_audio
Value: 48k
Variable
ID: data_rate
Value: 1.2k
Variable
ID: samp_per_sym
Value: 40
Variable
ID: channel_width
Value: 12k
Variable
ID: center_freq
Value: 425M
osmocom Source
Sample Rate (sps): 250k
Ch0: Frequency (Hz): 425M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 16
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 250k
Cutoff Freq: 4k
Transition Width: 1k
Window: Hamming
Beta: 6.76
Power Squelch
Threshold (dB): -40
Alpha: 100u
Ramp: 0
Gate: Yes
Rational Resampler
Interpolation: 48k
Decimation: 250k
Taps:
Fractional BW: 0
NBFM Receive
Audio Rate: 48k
Quadrature Rate: 48k
Tau: 75u
Max Deviation: 5k
Audio Sink
Sample Rate: 48KHz
Frequency Xlating FIR Filter
Decimation: 1
Taps: firdes.low_pass(1.0,...
Center Frequency: 1.5k
Sample Rate: 48k
Quadrature Demod
Gain: 1
Multiply Const
Constant: -1
Moving Average
Length: 20
Scale: 1
Max Iter: 4k
Rational Resampler
Interpolation: 1
Decimation: 10
Taps:
Fractional BW: 0
Clock Recovery MM
Omega: 4
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m
Binary Slicer
ZMQ PUB Sink
Address: tcp://127.0.0.1:5555
Timeout (msec): 100
Pass Tags: No


Options
ID: top_block
Generate Options: No GUI
Run Options: Prompt for Exit
Variable
ID: samp_rate
Value: 250k
Variable
ID: samp_rate_audio
Value: 48k
Variable
ID: data_rate
Value: 1.2k
Variable
ID: samp_per_sym
Value: 40
Variable
osmocom Source
Sample Rate (sps): 250k
Ch0: Frequency (Hz): 425M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 16
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 250k
Cutoff Freq: 4k
Transition Width: 1k
Window: Hamming
Beta: 6.76
Power Squelch
Threshold (dB): -40
Alpha: 100u
Ramp: 0
Gate: Yes
Frequency Xlating FIR Filter
Decimation: 1
Taps: firdes.low_pass(1.0,...
Center Frequency: 1.5k
Sample Rate: 48k
Quadrature Demod
Gain: 1
Multiply Const
Constant: -1


Power Squelch
Threshold (dB): -40
Alpha: 100u
Ramp: 0
Gate: Yes
Rational Resampler
Interpolation: 48k
Decimation: 250k
Taps:
Fractional BW: 0
NBFM Receive
Audio Rate: 48k
Quadrature Rate: 48k
Tau: 75u
Max Deviation: 5k
Audio Sink
Sample Rate: 48KHz
Multiply Const
Constant: -1
Moving Average
Length: 20
Scale: 1
Max iter: 4k
Rational Resampler
Interpolation: 1
Decimation: 10
Taps:
Fractional BW: 0
Clock Recovery MM
Omega: 4
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m


Value: 250k
Variable
ID: samp_rate_audio
Value: 48k
Variable
ID: data_rate
Value: 1.2k
Variable
ID: samp_per_sym
Value: 40
Variable
ID: channel_width
Value: 12k
Variable
ID: center_freq
Value: 425M
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 16
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Transition Width: 1k
Window: Hamming
Beta: 6.76
Ramp: 0
Gate: Yes
Frequency Xlating FIR Filter
Decimation: 1
Taps: firdes.low_pass(1.0,...
Center Frequency: 1.5k
Sample Rate: 48k
Quadrature Demod
Gain: 1
Multiply Const
Constant: -1


Ramp: 0
Gate: Yes
Taps:
Fractional BW: 0
Tau: 75u
Max Deviation: 5k
Multiply Const
Constant: -1
Moving Average
Length: 20
Scale: 1
Max Iter: 4k
Rational Resampler
Interpolation: 1
Decimation: 10
Taps:
Fractional BW: 0
Clock Recovery MM
Omega: 4
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m
Binary Slicer
ZMQ PUB Sink
Address: tcp://127.0.0.1:5555
Timeout (msec): 100
Pass Tags: No


Quad Demod Out
Data 0
Data 1
Data 2
FFSK Audio
Binary Data
Quadrature Demodulator Output
6.5545 ms, -4.6657
Amplitude
4
2
0
-2
-4
0
2
4
6
8
Time (ms)

```python
# Socket to talk to server
context = zmq.Context()
sock = context.socket(zmq.SUB)

# create fixed length queue
queue = collections.deque(maxlen=256)


def main():
    #  Connect to GNU Radio and subscribe to stream
    sock.connect("tcp://localhost:5555")
    sock.setsockopt(zmq.SUBSCRIBE, b'')

    while True:
        newData = sock.recv()  # get whatever data are available
        for byte in newData:
            queue.append(str(byte))  # append each new symbol to deque

            buffer = ''  # clear buffer
            for bit in queue:  # move deque contents into buffer
                buffer += bit

            if (buffer.find('10101011100010010') == 0):  # look for frame sync
                EOT = EOT_decode(buffer[6:])  # first 6 bits are bit sync
                if (EOT.valid):
                    printEOT(EOT)
```

```
class EOT_decode():
    def __init__(self, buffer):
        self.packet = buffer[0:74]
        self.frame_sync = self.packet[0:11]
        self.data_block = self.packet[11:56]
        self.batt_cond = (self.packet[13:15][::-1])
        self.message_type = self.packet[15:18]
        self.unit_addr = int((self.packet[18:35][::-1]), 2)
        self.pressure = int((self.packet[35:42][::-1]), 2)
        self.batt_charge = \
            ("{}%".format(int(int((self.packet[42:49][::-1]), 2) / 127 * 100)))
        self.spare = self.packet[49]
        self.valve_ckt_stat = self.packet[50]
        self.conf_ind = self.packet[51]
        self.turbine = self.packet[52]
        self.motion = self.packet[53]
        self.mkr_batt = self.packet[54]
        self.mkr_light = self.packet[55]
        self.checkbitsRx = self.packet[56:74]

        self.batt_cond_dict = {"11": "OK",
                               "10": "Low",
                               "01": "Very Low",
                               "00": "Not Monitored"}
        self.batt_cond_text = self.batt_cond_dict[self.batt_cond]

        if (self.message_type == "111"):
            if (self.conf_ind == "0"):
                self.arm_status = "Arming"
            else:
                self.arm_status = "Armed"
        else:
            self.arm_status = "Normal"

        self.generator = '1111001101000001111'  # BCH generator polynomial
        self.cipher_key = '101011011101110000'  # XOR cipher key
        self.data_block = helpers.reverse(self.data_block)
        self.checkbits = helpers.checkbits(self.data_block, self.generator)
        self.checkbits_cipher = helpers.xor(self.checkbits, self.cipher_key)
        self.valid = (self.checkbits_cipher == self.checkbitsRx)  # a match?
```


Quad Demod Out
Falken:pyEOT ereuter$ python3 eot_zmq.py

Live Demo

Security

Arming sequence prevents rogue HOT from sending emergency command to EOT:

1. HOT number dials set to match EOT ID
2. *Test* button pressed on EOT, sends ARMING packet to HOT
3. *Arm* button pressed on HOT, sends STATUS REQUEST packet to EOT
4. EOT confirms with ARMED packet

Unarmed HOT will transmit status request but not emergency command.

Security

- No way for EOT to authenticate transmission
- Replay attack impractical, as emergency events are rare
- Presumed to be vulnerable to spoofing

Paul V. Craven & Steven Craven (2005) **Security of Railway EOT Systems**, *Proceedings of JRC2005 (*pp. 199-204)

- Notes vulnerability to spoofing
- Proposed use of Diffie-Hellman key exchange for HOT to EOT transmissions

Automatic Equipment Identification (AEI)

Harvey Henkelmann Photo

Automatic Equipment Identification (AEI)

Beginning in 1960s, the US railroad industry sought a way to automatically identify equipment (cars, locomotives, etc.)

Automatic Equipment Identification (AEI)

- KarTrak (1967) used color barcodes
- Poor read rates due to physical damage and dirt accumulation
- Abandoned in 1977

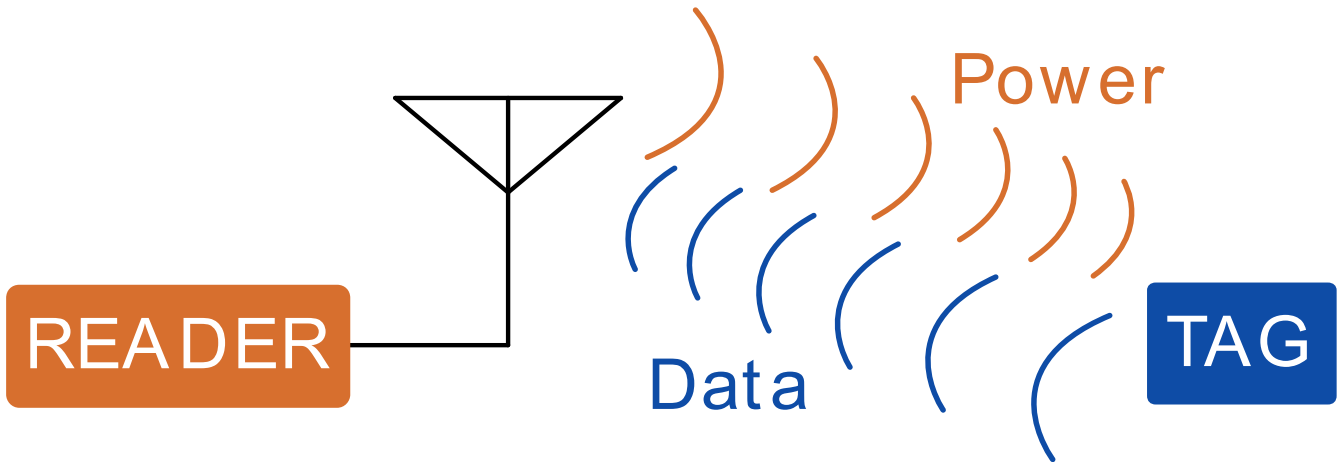
Automatic Equipment Identification (AEI)

- AEI implemented beginning in 1991
- RFID tags affixed to all rail equipment in US by 1994

Radio Frequency Identification (RFID)

Two principal components:

- **Reader** – Active radio transceiver
- **Tag** – Passive (usually) device that transmits data when powered by an RF carrier



Types of RFID

Inductive Coupling (coil)
- 125 kHz (LF) & 13.56 Mhz (HF)
- Short range (< 1 m)
- Building access, inventory control, animal ID, ticketing, etc.

Long Range (antenna)
- 433 Mhz, **900 Mhz (UHF)** & Microwave
- Range up to 12 m
- Vehicle tracking, inventory control
- Some tags are active

AEI Standards

Air Interface: ISO 10374

- Defines physical interface and communication protocol (900 Mhz ISM Band)
- Tag returns a 128-bit ID when interrogated
- Used by other industries in the past (tolling, intermodal), but mostly obsolete.

Data Format: AAR S-918

- Defines data format of 128-bit ID
- Provides guidance on tag placement
- Apparently superseded by S-9203 and then S-6009

AEI Tag Data

Tag contains these fields:

- Equipment type (car, locomotive, etc.)
- Reporting Mark (owner’s initials)
- Car/Locomotive Number
- Side (left or right)
- Length
- Number of axles
- Bearing type
- Additional equipment-specific parameters

AEI Performance

- Range depends on effective radiated power (ERP) and train speed.

- Maximum performance observed was 20 ft (12 m) for 70 mph (112 km/h) train at 32 W ERP.

AEI Performance

ERP Calculatation

ERP = transmitter power + antenna gain

Example:
2 W transmitter power into
9 dBi antenna (dB re: isotropic)

ERP = 2 W * $10^{0.9}$ = <u>15.9 W</u>

AEI Performance

Conventions for Reader RF Power

- Low-power reader
  - 1 W + 9 dBiC (Circular polarization: -3 dB)
  - ERP = 5 W
  - FCC Part 15 (no license required)
  - Not practical unless on RR property
- High-power reader
  - 2 W + any antenna gain
  - ERP may be as high as 32 W
  - FCC Part 90 (site license required)

Experimental Setup

Sirit ID5100

- Obsolete – eBay purchase
- Supports ISO 10374
- Used with 9 dBi Yagi
- Part 90 Device
- Operating under my amateur radio license
- ID by togging carrier to send CW

Experimental Setup

Sirit ID5100

- Runs Linux on internal microprocessor
- Communicates via TCP over ethernet
- Provides raw tag data
- Can log tag data internally

Software

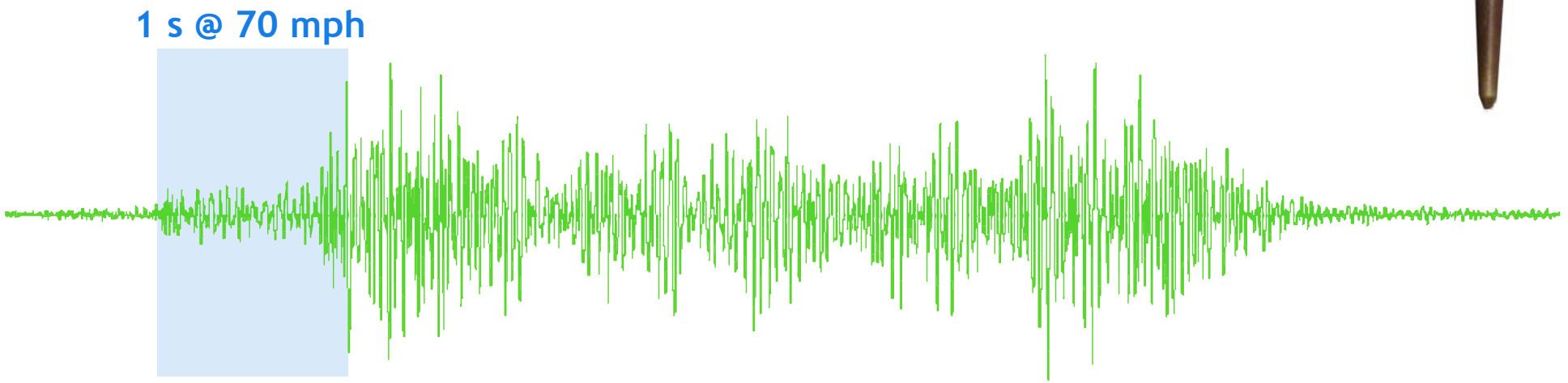
- Developed Python class decode and analyze tags
- Can be used for real-time processing or post-processing of log files.

Realtime processing test – 70 mph

Experimental Setup

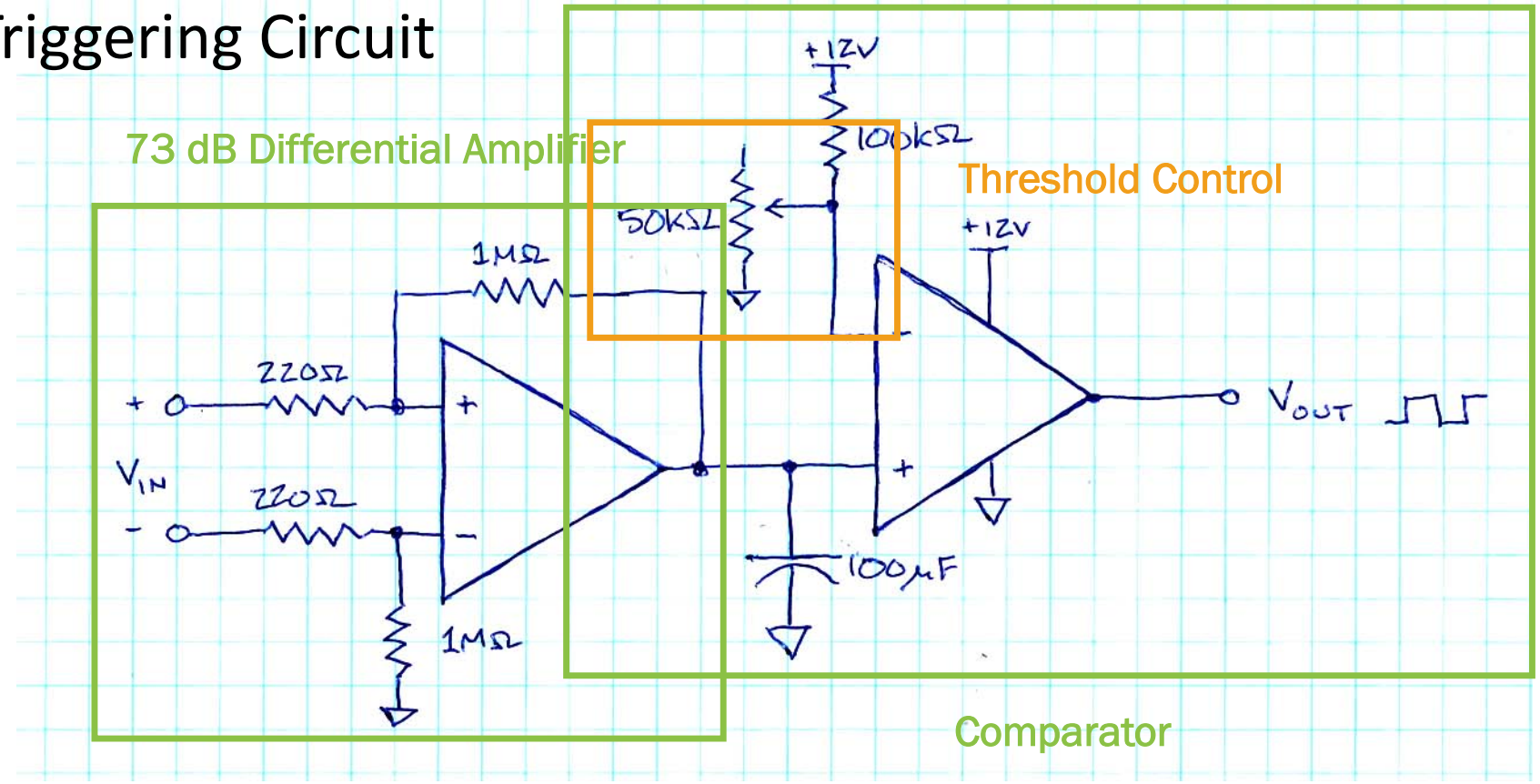
Triggering

- Triggering reduces power consumption, heat, and RF pollution.
- Railroads use wheel flange detectors
- I used ground vibration
- Seismic geophone

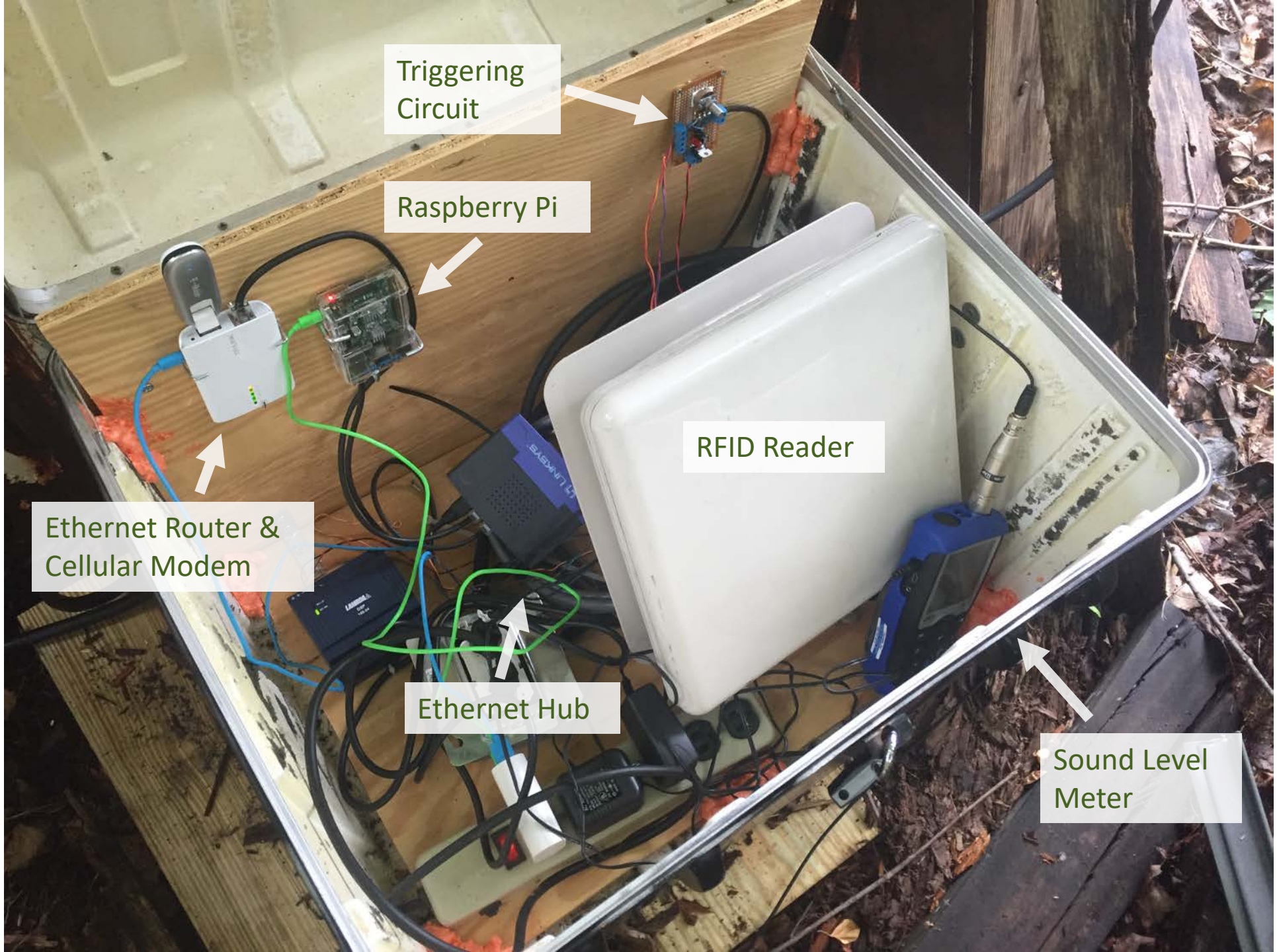


Experimental Setup

Triggering Circuit

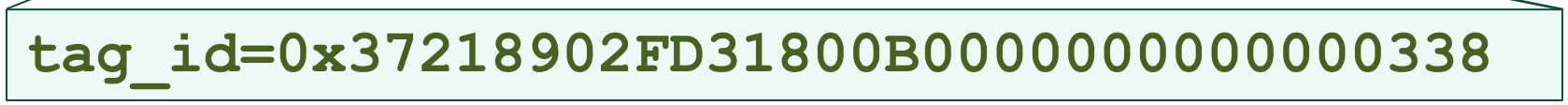



reader is in standby mode
dio.in.1: 'e - reader is in active mode
timer popped, switching to standby
reader is in standby mode
Connected to Channel 27


Triggering Circuit
Raspberry Pi
RFID Reader
Ethernet Router & Cellular Modem
Ethernet Hub
Sound Level Meter

Data : Raw

```
event.tag.arrive tag_id=0x2AEA4C8004F6B413D9F00144002E8730, first=2016-08-18T23:04:20.334
event.tag.arrive tag_id=0x9E44F94421AED01B90000000002E8738, first=2016-08-18T23:04:21.631
event.tag.arrive tag_id=0x9F7C28C51B8CCF1F90000000002E8738, first=2016-08-18T23:04:23.245
event.tag.arrive tag_id=0x9F6710C2110AD21390000000002E8738, first=2016-08-18T23:04:24.993
event.tag.arrive tag_id=0x9F73F5929B4CD21790000000002E8738, first=2016-08-18T23:04:26.704
event.tag.arrive tag_id=0x9E286B2FACD4C8179000000000000330, first=2016-08-18T23:04:28.461
event.tag.arrive tag_id=0x9E286B2FAE34C8179000000000000330, first=2016-08-18T23:04:30.165
event.tag.arrive tag_id=0x9F6710C21104D21F90000000002E8738, first=2016-08-18T23:04:31.938
event.tag.arrive tag_id=0x9F73F5929DC0D21390000000002E8738, first=2016-08-18T23:04:33.752
event.tag.arrive tag_id=0x9F73F5929B1AD21790000000002E8738, first=2016-08-18T23:04:35.666
event.tag.arrive tag_id=0x9E286B2FAD5EC81F900000000000330, first=2016-08-18T23:04:37.485
event.tag.arrive tag_id=0x9B73F59270C2D21B90000000002E8738, first=2016-08-18T23:04:39.419
event.tag.arrive tag_id=0x9F6710C21156D21B90000000002E8738, first=2016-08-18T23:04:41.367
event.tag.arrive tag_id=0x37218902FD31800B0000000000000338, first=2016-08-18T23:04:43.106
```



32 digits (hex) = 128 bits

Data : Formatted

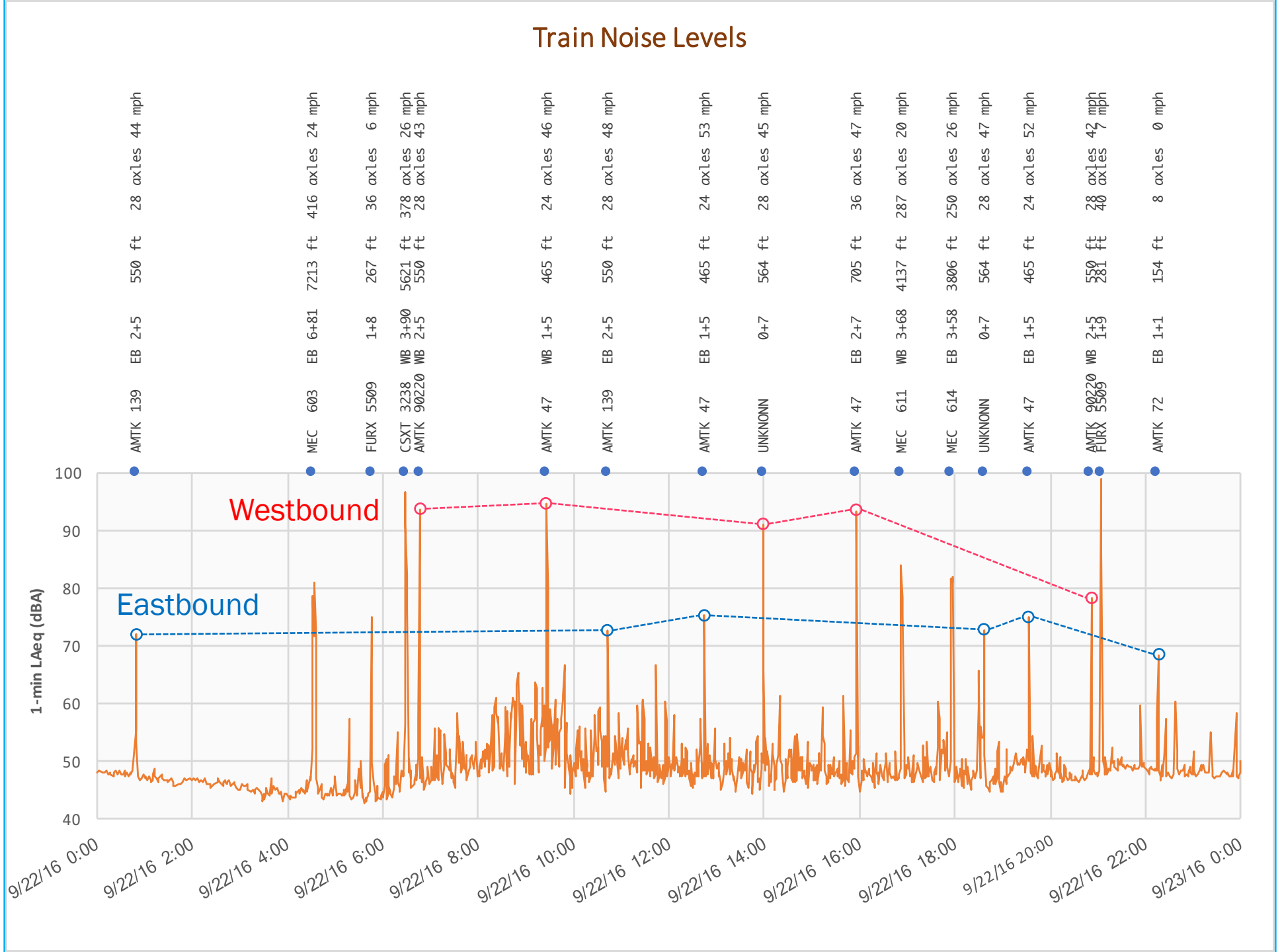
```
Train 51 2016-08-18 23:04:20.334000

1   LOC MEC 317   R 59.0 ft 4 axles 23:04:20.334
2   CAR DOWX 67691  R 68.2 ft 4 axles 23:04:21.631
3   CAR TTZX 83683  L 67.9 ft 4 axles 23:04:23.245
4   CAR SRIX 33858  R 68.9 ft 4 axles 23:04:24.993
5   CAR TILX 304851 L 68.9 ft 4 axles 23:04:26.704
6   CAR CBTX 781109 L 65.6 ft 4 axles 23:04:28.461
7   CAR CBTX 781197 L 65.6 ft 4 axles 23:04:30.165
8   CAR SRIX 33857  L 68.9 ft 4 axles 23:04:31.938
9   CAR TILX 305008 L 68.9 ft 4 axles 23:04:33.752
10  CAR TILX 304838 R 68.9 ft 4 axles 23:04:35.666
11  CAR CBTX 781143 R 65.6 ft 4 axles 23:04:37.485
12  CAR TILX 302128 R 68.9 ft 4 axles 23:04:39.419
13  CAR SRIX 33877  R 68.9 ft 4 axles 23:04:41.367
14  EOT PARX 48972

Total Length = 874.1 ft
Total Time = 22.772 s
Speed = 26.2 mph
```

Data : Formatted

```
2016-09-17 16:32:57 AMTK 22    WB 1+6    564 ft  28 axles 43 mph
2016-09-17 18:31:48 AMTK 139   EB 2+5    550 ft  28 axles 47 mph
2016-09-17 20:42:30 AMTK 140   WB 1+6    550 ft  28 axles 44 mph
2016-09-17 20:59:58 CSXT 5107  EB 3+102 6085 ft 426 axles 23 mph
2016-09-17 21:28:17 AMTK 22    EB 1+6    564 ft  28 axles 49 mph
2016-09-17 22:17:20 MEC  611   WB 3+101 5751 ft 420 axles 21 mph CSXE 44973
2016-09-18 00:43:03 AMTK 140   EB 1+6    550 ft  28 axles 46 mph
2016-09-18 07:06:47 CSXT 3285  WB 4+110 8361 ft 510 axles  8 mph CSXE 58355
2016-09-18 09:35:16 AMTK 22    WB 1+6    564 ft  28 axles 52 mph
2016-09-18 10:56:02 AMTK 821   EB 1+6    550 ft  28 axles 49 mph
2016-09-18 13:16:31 AMTK 22    EB 1+6    564 ft  28 axles 47 mph
2016-09-18 14:24:41 AMTK 90220 WB 2+5    550 ft  28 axles 37 mph
2016-09-18 16:25:02 AMTK 22    WB 1+6    564 ft  28 axles 44 mph
2016-09-18 16:43:53 CSXT 931   EB 2+95  5986 ft 392 axles 26 mph CSXE 48521
2016-09-18 18:28:11 AMTK 139   EB 2+5    550 ft  28 axles 46 mph
2016-09-18 20:43:20 AMTK 821   WB 1+6    550 ft  28 axles 43 mph
2016-09-18 21:28:20 AMTK 22    EB 1+6    564 ft  28 axles 49 mph
2016-09-19 00:42:38 AMTK 821   EB 1+6    550 ft  28 axles 45 mph
2016-09-19 02:57:14 MEC  614   EB 4+61  2994 ft 268 axles 26 mph
2016-09-19 06:10:14 CSXT 3068  WB 2+69  5633 ft 328 axles 20 mph WRE  97812
2016-09-19 06:31:42 AMTK 22    WB 1+6    564 ft  28 axles 51 mph
2016-09-19 09:14:08 AMTK 821   WB 1+5    465 ft  24 axles 47 mph
2016-09-19 10:42:24 AMTK 22    EB 1+6    564 ft  28 axles 48 mph
2016-09-19 13:00:19 AMTK 821   EB 1+5    465 ft  24 axles 51 mph
2016-09-19 14:06:43 AMTK 90220 WB 2+5    550 ft  28 axles 48 mph
2016-09-19 15:03:57 MEC  610   EB 2+54  3386 ft 228 axles 26 mph
```


Train Noise Levels
AMTK 139 28 axles 44 mph
EB 2+5 550 ft
MEC 603 416 axles 24 mph
EB 6+81 7213 ft
FURX 5509 36 axles 6 mph
1+8 267 ft
CSXT 3238 378 axles 26 mph
AMTK 90220 28 axles 43 mph
WB 3+90 5621 ft
WB 2+5 550 ft
AMTK 47 24 axles 46 mph
WB 1+5 465 ft
AMTK 139 28 axles 48 mph
EB 2+5 550 ft
AMTK 47 24 axles 53 mph
EB 1+5 465 ft
UNKNONN 28 axles 45 mph
0+7 564 ft
AMTK 47 36 axles 47 mph
EB 2+7 705 ft
MEC 611 287 axles 20 mph
WB 3+68 4137 ft
MEC 614 250 axles 26 mph
EB 3+58 3806 ft
UNKNONN 28 axles 47 mph
0+7 564 ft
AMTK 47 24 axles 52 mph
EB 1+5 465 ft
AMTK 90220 28 axles 42 mph
FURX 5509 40 axles 7 mph
WB 2+5 550 ft
1+9 281 ft
AMTK 72 8 axles 0 mph
EB 1+1 154 ft
Westbound
Eastbound
1-min LAeq (dBA)
100
90
80
70
60
50
40
9/22/16 0:00
9/22/16 2:00
9/22/16 4:00
9/22/16 6:00
9/22/16 8:00
9/22/16 10:00
9/22/16 12:00
9/22/16 14:00
9/22/16 16:00
9/22/16 18:00
9/22/16 20:00
9/22/16 22:00
9/23/16 0:00

Security

- Tags don’t contain any information that isn’t painted on the car
- No indication of type or presence of load
- Sensitivity to private readers exists in industry
- Be certain that
  1. You have a proper license
  2. You are not on railroad property

Trains

Login or Register | Customer Service

Home / News / News Wire / FCC fines clandestine train tracking company

14

FCC fines clandestine train tracking company

Federal communications agency says company had incorrect licensing to install special readers along railroad rights-of-way

By Justin Franz | September 27, 2016

RELATED TOPICS: CLASS 1 FREIGHT RAILROADS | COMMODITIES | SECURITY | CRIME

WASHINGTON D.C. — A New York-based data-company must pay a $195,000 fine a year after it was caught setting up unauthorized freight car trackers near rights-of-way across the country.

On Aug. 29, the enforcement bureau of the Federal Communications Commission announced it had entered into an agreement with ClipperData LLC for violating federal communications law because it was installing and operating automatic equipment identification, or AEI, readers without acquiring the correct license.

What's in a picture? In this one, a brown-painted automatic equipment identification, or AEI, reader barely pokes out of a slope along a Conrail right-of-way in New Jersey. A company that installed this reader received a $195,000 fine from the Federal Communications Commissions for having incorrect licensing for tracking freight trains near rights-of-way.

In September 2015, *Trains* News Wire first reported that the Association of American Railroads’ Railway Alert

Resources

- **PyEOT** (shown in this talk): https://github.com/ereuter/PyEOT

- **SoftEOT** (Windows): https://groups.yahoo.com/neo/groups/SoftEOT
- **ATCS Monitor** (Windows): https://groups.yahoo.com/neo/groups/ATCS-Monitor

- **BCH Code**: https://en.wikipedia.org/wiki/BCH_code

@EricReuter