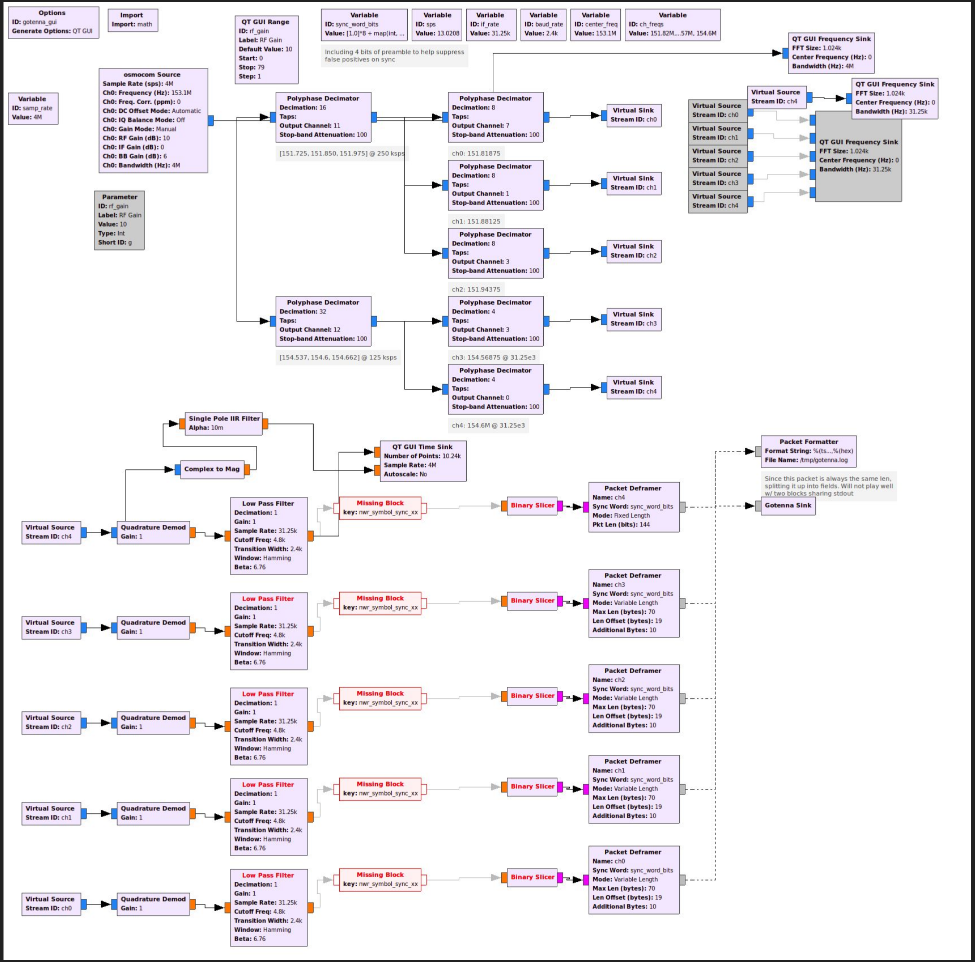
gr-reveng

Packet Tools for GNU Radio

Tim K (@bjt2n3904)



Options
ID: gotenna_gui
Generate Options: QT GUI

Import
Import: math

QT GUI Range
ID: rf_gain
Label: RF Gain
Default Value: 10
Start: 0
Stop: 79
Step: 1

Variable
ID: sync_word_bits
Value: [1,0]*8 + map(int, ...

Including 4 bits of preamble to help suppress false positives on sync

Variable
ID: sps
Value: 13.0208

Variable
ID: if_rate
Value: 31.25k

Variable
ID: baud_rate
Value: 2.4k

Variable
ID: center_freq
Value: 153.1M

Variable
ID: ch_freqs
Value: 151.82M,...57M, 154.6M

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 4M

Variable
ID: samp_rate
Value: 4M

osmocom Source
Sample Rate (sps): 4M
Ch0: Frequency (Hz): 153.1M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Automatic
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 0
Ch0: BB Gain (dB): 6
Ch0: Bandwidth (Hz): 4M

Parameter
ID: rf_gain
Label: RF Gain
Value: 10
Type: Int
Short ID: g

Polyphase Decimator
Decimation: 16
Taps:
Output Channel: 11
Stop-band Attenuation: 100

[151.725, 151.850, 151.975] @ 250 ksps

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 7
Stop-band Attenuation: 100

ch0: 151.81875

Virtual Sink
Stream ID: ch0

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 1
Stop-band Attenuation: 100

ch1: 151.88125

Virtual Sink
Stream ID: ch1

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 3
Stop-band Attenuation: 100

ch2: 151.94375

Virtual Sink
Stream ID: ch2

Polyphase Decimator
Decimation: 32
Taps:
Output Channel: 12
Stop-band Attenuation: 100

[154.537, 154.6, 154.662] @ 125 ksps

Polyphase Decimator
Decimation: 4
Taps:
Output Channel: 3
Stop-band Attenuation: 100

ch3: 154.56875 @ 31.25e3

Virtual Sink
Stream ID: ch3

Polyphase Decimator
Decimation: 4
Taps:
Output Channel: 0
Stop-band Attenuation: 100

ch4: 154.6M @ 31.25e3

Virtual Sink
Stream ID: ch4

Virtual Source
Stream ID: ch4

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 31.25k

Virtual Source
Stream ID: ch0

Virtual Source
Stream ID: ch1

Virtual Source
Stream ID: ch2

Virtual Source
Stream ID: ch3

Virtual Source
Stream ID: ch4

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 31.25k

Single Pole IIR Filter
Alpha: 10m

Complex to Mag

QT GUI Time Sink
Number of Points: 10.24k
Sample Rate: 4M
Autoscale: No

Packet Formatter
Format String: %(ts...,%(hex)
File Name: /tmp/gotenna.log

Since this packet is always the same len, splitting it up into fields. Will not play well w/ two blocks sharing stdout

Gotenna Sink

Virtual Source
Stream ID: ch4

Quadrature Demod
Gain: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 31.25k
Cutoff Freq: 4.8k
Transition Width: 2.4k
Window: Hamming
Beta: 6.76

Missing Block
key: nwr_symbol_sync_xx

Binary Slicer

Packet Deframer
Name: ch4
Sync Word: sync_word_bits
Mode: Fixed Length
Pkt Len (bits): 144

Virtual Source
Stream ID: ch3

Quadrature Demod
Gain: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 31.25k
Cutoff Freq: 4.8k
Transition Width: 2.4k
Window: Hamming
Beta: 6.76

Missing Block
key: nwr_symbol_sync_xx

Binary Slicer

Packet Deframer
Name: ch3
Sync Word: sync_word_bits
Mode: Variable Length
Max Len (bytes): 70
Len Offset (bytes): 19
Additional Bytes: 10

Virtual Source
Stream ID: ch2

Quadrature Demod
Gain: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 31.25k
Cutoff Freq: 4.8k
Transition Width: 2.4k
Window: Hamming
Beta: 6.76

Missing Block
key: nwr_symbol_sync_xx

Binary Slicer

Packet Deframer
Name: ch2
Sync Word: sync_word_bits
Mode: Variable Length
Max Len (bytes): 70
Len Offset (bytes): 19
Additional Bytes: 10

Virtual Source
Stream ID: ch1

Quadrature Demod
Gain: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 31.25k
Cutoff Freq: 4.8k
Transition Width: 2.4k
Window: Hamming
Beta: 6.76

Missing Block
key: nwr_symbol_sync_xx

Binary Slicer

Packet Deframer
Name: ch1
Sync Word: sync_word_bits
Mode: Variable Length
Max Len (bytes): 70
Len Offset (bytes): 19
Additional Bytes: 10

Virtual Source
Stream ID: ch0

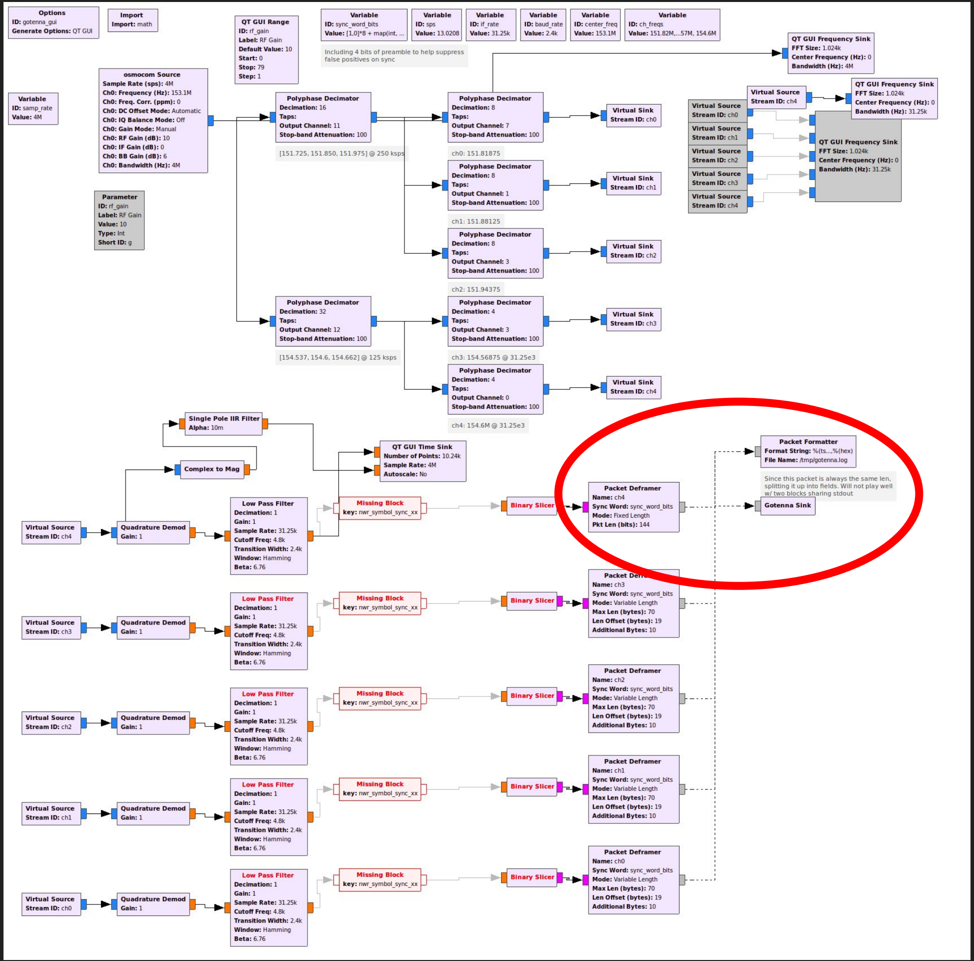
Quadrature Demod
Gain: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 31.25k
Cutoff Freq: 4.8k
Transition Width: 2.4k
Window: Hamming
Beta: 6.76

Missing Block
key: nwr_symbol_sync_xx

Binary Slicer

Packet Deframer
Name: ch0
Sync Word: sync_word_bits
Mode: Variable Length
Max Len (bytes): 70
Len Offset (bytes): 19
Additional Bytes: 10
)



Options
ID: gotenna_gui
Generate Options: QT GUI

Import
Import: math

QT GUI Range
ID: rf_gain
Label: RF Gain
Default Value: 10
Start: 0
Stop: 79
Step: 1

Variable
ID: sync_word_bits
Value: [1,0]*8 + map(int, ...

Including 4 bits of preamble to help suppress false positives on sync

Variable
ID: sps
Value: 13.0208

Variable
ID: if_rate
Value: 31.25k

Variable
ID: baud_rate
Value: 2.4k

Variable
ID: center_freq
Value: 153.1M

Variable
ID: ch_freqs
Value: 151.82M,...57M, 154.6M

Variable
ID: samp_rate
Value: 4M

osmocom Source
Sample Rate (sps): 4M
Ch0: Frequency (Hz): 153.1M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Automatic
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 0
Ch0: BB Gain (dB): 6
Ch0: Bandwidth (Hz): 4M

Parameter
ID: rf_gain
Label: RF Gain
Value: 10
Type: Int
Short ID: g

Polyphase Decimator
Decimation: 16
Taps:
Output Channel: 11
Stop-band Attenuation: 100

[151.725, 151.850, 151.975] @ 250 ksps

Polyphase Decimator
Decimation: 32
Taps:
Output Channel: 12
Stop-band Attenuation: 100

[154.537, 154.6, 154.662] @ 125 ksps

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 7
Stop-band Attenuation: 100

ch0: 151.81875

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 1
Stop-band Attenuation: 100

ch1: 151.88125

Polyphase Decimator
Decimation: 8
Taps:
Output Channel: 3
Stop-band Attenuation: 100

ch2: 151.94375

Polyphase Decimator
Decimation: 4
Taps:
Output Channel: 3
Stop-band Attenuation: 100

ch3: 154.56875 @ 31.25e3

Polyphase Decimator
Decimation: 4
Taps:
Output Channel: 0
Stop-band Attenuation: 100

ch4: 154.6M @ 31.25e3

Virtual Sink **Stream ID:** ch0
Virtual Sink **Stream ID:** ch1
Virtual Sink **Stream ID:** ch2
Virtual Sink **Stream ID:** ch3
Virtual Sink **Stream ID:** ch4

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 4M

Virtual Source **Stream ID:** ch4

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 31.25k

Virtual Source **Stream ID:** ch0
Virtual Source **Stream ID:** ch1
Virtual Source **Stream ID:** ch2
Virtual Source **Stream ID:** ch3
Virtual Source **Stream ID:** ch4

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 31.25k

Single Pole IIR Filter
Alpha: 10m

Complex to Mag

QT GUI Time Sink
Number of Points: 10.24k
Sample Rate: 4M
Autoscale: No

Virtual Source **Stream ID:** ch4 → **Quadrature Demod** **Gain:** 1 → **Low Pass Filter** **Decimation:** 1 **Gain:** 1 **Sample Rate:** 31.25k **Cutoff Freq:** 4.8k **Transition Width:** 2.4k **Window:** Hamming **Beta:** 6.76 → **Missing Block** **key:** nwr_symbol_sync_xx → **Binary Slicer** → **Packet Deframer** **Name:** ch4 **Sync Word:** sync_word_bits **Mode:** Fixed Length **Pkt Len (bits):** 144

Virtual Source **Stream ID:** ch3 → **Quadrature Demod** **Gain:** 1 → **Low Pass Filter** **Decimation:** 1 **Gain:** 1 **Sample Rate:** 31.25k **Cutoff Freq:** 4.8k **Transition Width:** 2.4k **Window:** Hamming **Beta:** 6.76 → **Missing Block** **key:** nwr_symbol_sync_xx → **Binary Slicer** → **Packet Deframer** **Name:** ch3 **Sync Word:** sync_word_bits **Mode:** Variable Length **Max Len (bytes):** 70 **Len Offset (bytes):** 19 **Additional Bytes:** 10

Virtual Source **Stream ID:** ch2 → **Quadrature Demod** **Gain:** 1 → **Low Pass Filter** **Decimation:** 1 **Gain:** 1 **Sample Rate:** 31.25k **Cutoff Freq:** 4.8k **Transition Width:** 2.4k **Window:** Hamming **Beta:** 6.76 → **Missing Block** **key:** nwr_symbol_sync_xx → **Binary Slicer** → **Packet Deframer** **Name:** ch2 **Sync Word:** sync_word_bits **Mode:** Variable Length **Max Len (bytes):** 70 **Len Offset (bytes):** 19 **Additional Bytes:** 10

Virtual Source **Stream ID:** ch1 → **Quadrature Demod** **Gain:** 1 → **Low Pass Filter** **Decimation:** 1 **Gain:** 1 **Sample Rate:** 31.25k **Cutoff Freq:** 4.8k **Transition Width:** 2.4k **Window:** Hamming **Beta:** 6.76 → **Missing Block** **key:** nwr_symbol_sync_xx → **Binary Slicer** → **Packet Deframer** **Name:** ch1 **Sync Word:** sync_word_bits **Mode:** Variable Length **Max Len (bytes):** 70 **Len Offset (bytes):** 19 **Additional Bytes:** 10

Virtual Source **Stream ID:** ch0 → **Quadrature Demod** **Gain:** 1 → **Low Pass Filter** **Decimation:** 1 **Gain:** 1 **Sample Rate:** 31.25k **Cutoff Freq:** 4.8k **Transition Width:** 2.4k **Window:** Hamming **Beta:** 6.76 → **Missing Block** **key:** nwr_symbol_sync_xx → **Binary Slicer** → **Packet Deframer** **Name:** ch0 **Sync Word:** sync_word_bits **Mode:** Variable Length **Max Len (bytes):** 70 **Len Offset (bytes):** 19 **Additional Bytes:** 10

Packet Formatter
Format String: %(ts...,%(hex)
File Name: /tmp/gotenna.log

Since this packet is always the same len, splitting it up into fields. Will not play well w/ two blocks sharing stdout

Gotenna Sink

From Packets to Serial Data Streams

Packet Structure:

- Transmitter ID: **`0x1234`**
- Sensor Value: **`0xFF`**
- Checksum: **`0xD9`**

From Packets to Serial Data Streams

Packet Structure:

- Transmitter ID: **`0x1234`**
- Sensor Value: **`0xFF`**
- Checksum: **`0xD9`**

--	**--**	**12**	**34**	**FF**	**D9**

From Packets to Serial Data Streams

Packet Structure:

- Transmitter ID: `0x1234`
- Sensor Value: `0xFF`
- Checksum: `0xD9`

--	--	12	34	FF	D9
--	--	0001 0010	0011 0100	1111 1111	1101 1001

Then, magic happens!

PROBLEM: How to find packet?

0001 0010 0011 0100 1111

1111 1101 1001

PROBLEM: How to find packet?

1110 1101 1010 0100 1000 0001 0011 0110 1101 1010 0100 1000 0000 0001 0011 0110 1100 1001 0010 0101 1010 0100 1001 0010 0100 1000 0000 0000 0000 0001 0011 0111 1110 1100 1000 0001 0010 **0001 0010 0011 0100 1111 1111 1101 1001** 1011 0111 1110 1101 1010 0101 1010 0101 1011 0110 1101 1010 0100 1001 0010 0101 1011 0110 1100 1000 0001 0011 0110 1101 1011 0110 1101 1010 0101 1011 0110 1100 1001 0010 0100 1000 0000 0000 0000 0001 0011 0110 1100 1000 0001 0010 0100 1001 0011 0111 1110 1100 1001 0010 0100 1001 0011 0110 1101 1011 0111 1111 1110 1101 1011 0111 1110 1101 1011 0111 1110 1101 1011 0110 1101 1011 0111 1110 1100 1000 0001 0010 0100 1000 0000 0000 0000 0001 0011 1101

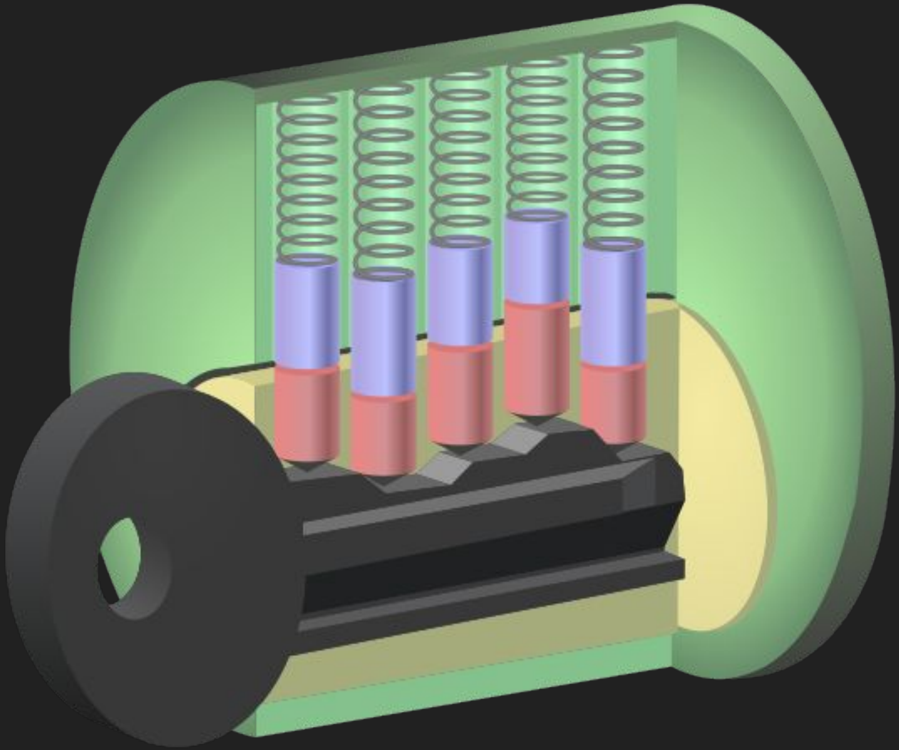
PROBLEM: How to find packet?

1110110110100100100000001001101101101101001001000000000001001101101
1001001001001011010010010010010010010000000000000000000001001101111 1
1011001000000010010**0001001000110100111111111110110011**01101111110110
1101001011010010110110110110110100100100100100101101101101 1001000
00010011011011011011011011011010010110110110110010010010010010000
0000000000000001001101101100100000010010010010011011111110110010
0100100100100100110110110110110111111111101101101101111111101101101
10111111101101101101101101101101111111011001000000100100100000000
0000000000000100111101

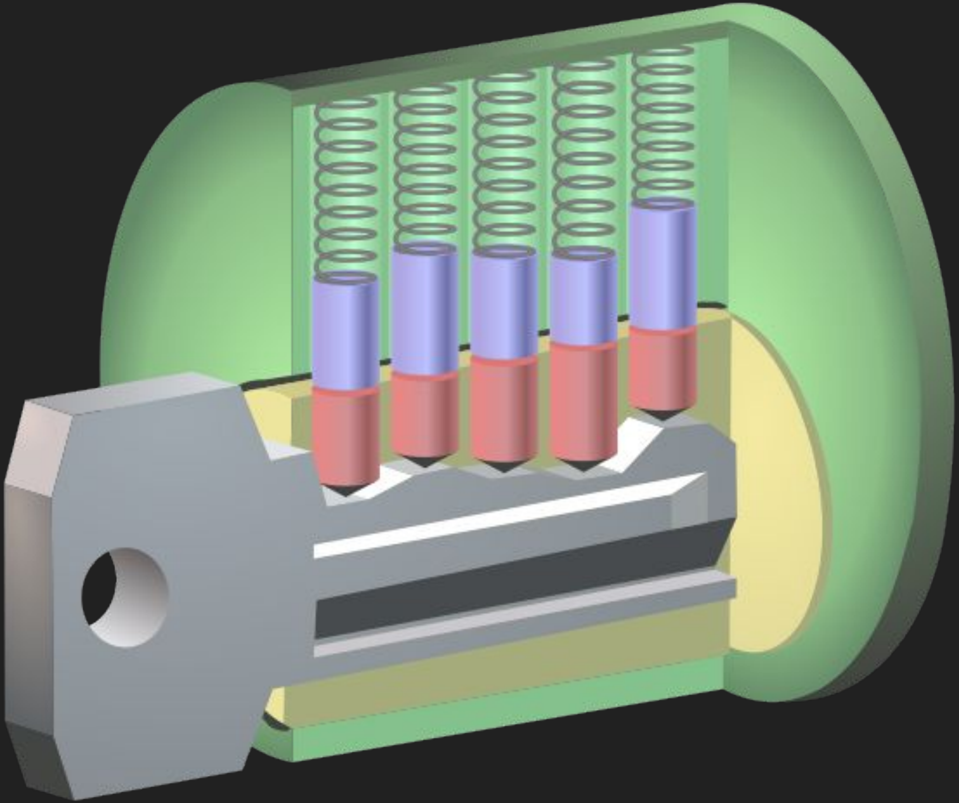
PROBLEM: How to find packet?

```
11101101101001001000000100110110110100100100000000001001101101
```

SOLUTION: Sync Words

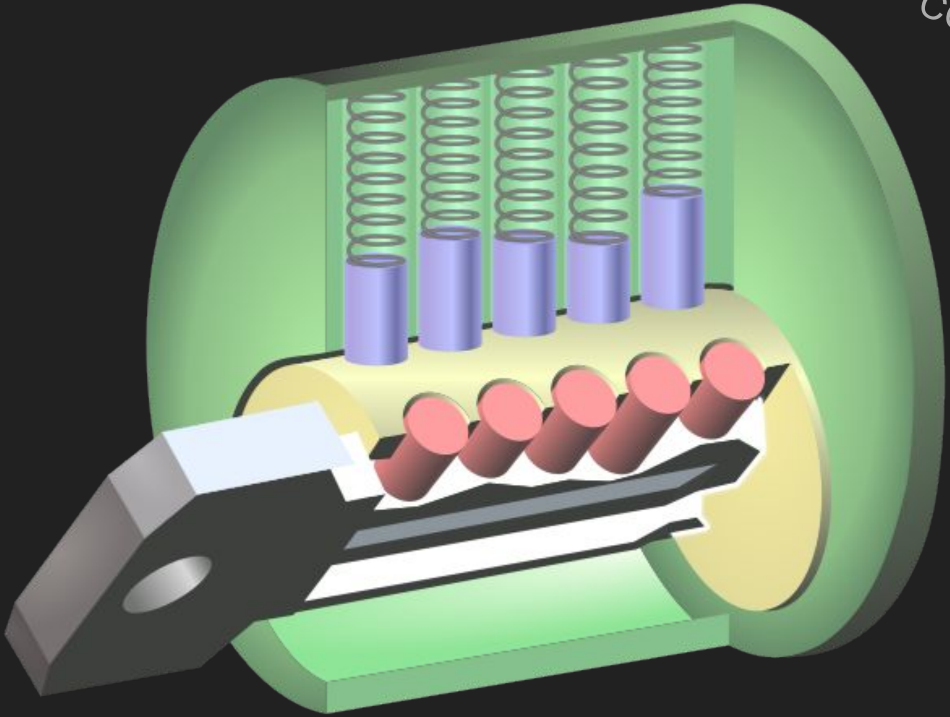


SOLUTION: Sync Words



SOLUTION: Sync Words

Thanks pbroks13 from Wikipedia!
CC-BY-SA 3.0



SOLUTION: Sync Words

```
01011101010010010101110110110110
0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
 0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
  0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
   0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
    0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
          0100100
```

SOLUTION: Sync Words

```
0101110101001001010111011011011O
0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
        0100100
```

SOLUTION: Sync Words

```
01011101010010010101110110110110
        0100100
```

SOLUTION: Sync Words

```
010111010100100|10101110|11011011|0
        0100100|   AE   |   DA   |
```

What makes a good sync word?

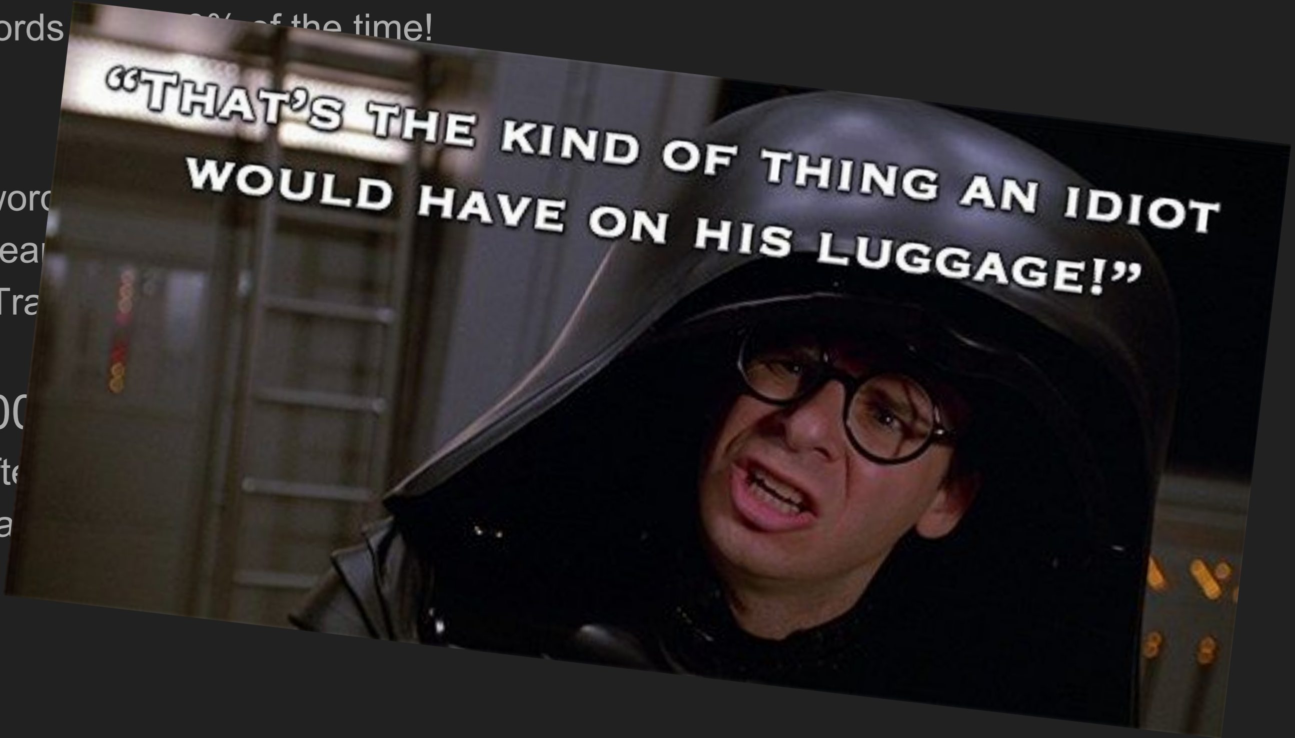
- Not too short!
  - One bit sync words latch 50% of the time!

- Not too long!
  - 2048-bit sync words take up too much air time
  - One bit error means you could lose the entire packet!
  - Special Case: Transmit nothing but sync words? (Mossman’s DSSS talk last year!)

- Not too simple: 0000, 1111, or 0101
  - Happens too often in nature
  - Leads to bad packet sync

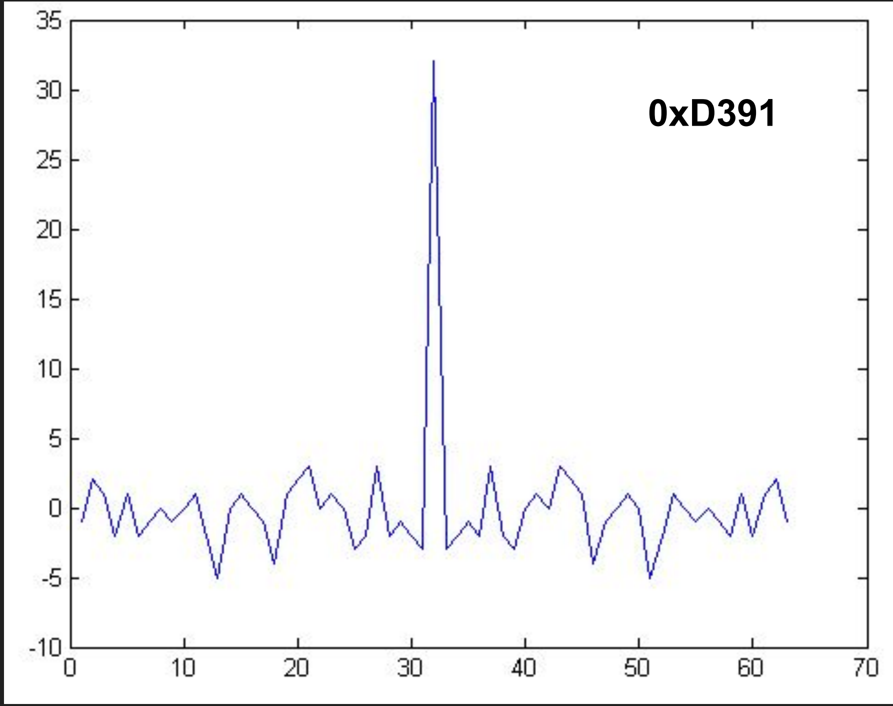
What makes a good sync word?

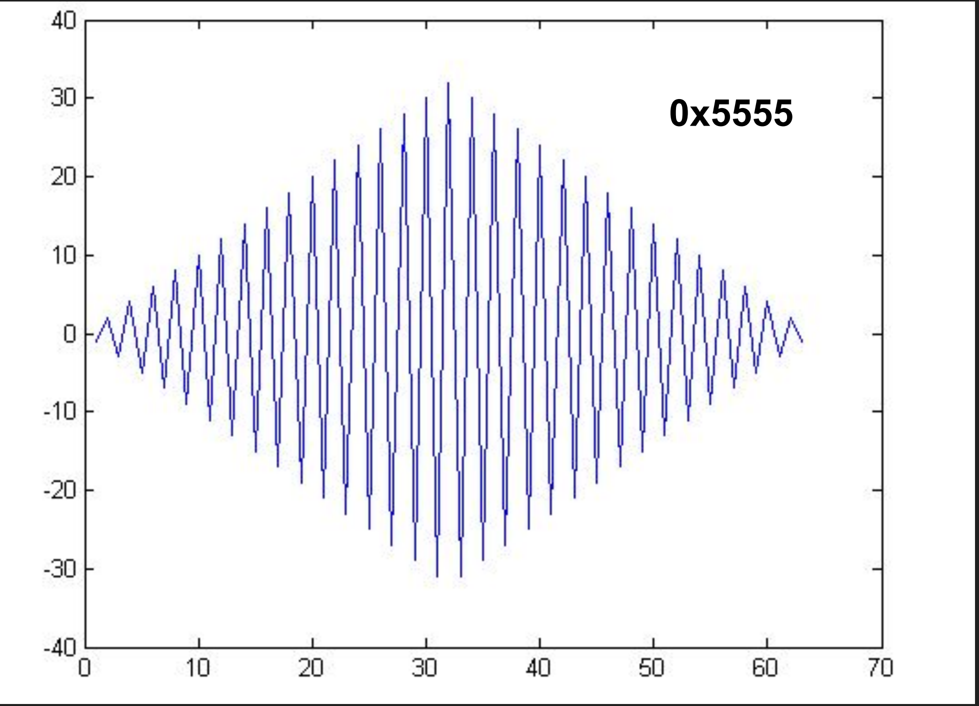
- Not too short!
  - One bit sync words ... of the time!
- Not too long!
  - 2048-bit sync word
  - One bit error mea
  - Special Case: Tra
- Not too simple: 00
  - Happens too ofte
  - Leads to bad pa


"THAT'S THE KIND OF THING AN IDIOT WOULD HAVE ON HIS LUGGAGE!"

What makes a good sync word?

Thanks Texas Instruments!





Revised Packet

Packet Structure:

- Sync Word: `0xD391`
- Transmitter ID: `0x1234`
- Sensor Value: `0xFF`
- Checksum: `0xD9`

D3	91	12	34	FF	D9
1101 0011	1001 0001	0001 0010	0011 0100	1111 1111	1101 1001

Revised Packet

Packet Structure:

- Sync Word: `0xD391`
- Transmitter ID: `0x1234`
- Sensor Value: `0xFF`
- Checksum: `0xD9`

Deframer Settings:

- Sync Word: `reveng.hex2bits('d391')`
- Mode: `Fixed Length`
- Packet Length: `8 * 4`

D3	91	12	34	FF	D9
1101 0011	1001 0001	0001 0010	0011 0100	1111 1111	1101 1001

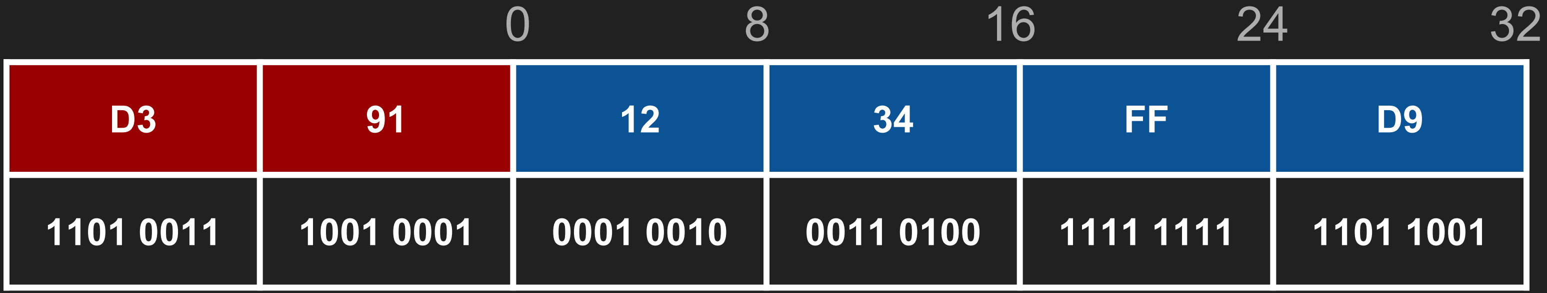
Revised Packet

Packet Structure:

- Sync Word: **`0xD391`**
- Transmitter ID: **`0x1234`**
- Sensor Value: **`0xFF`**
- Checksum: **`0xD9`**

Format String:

```
TX ID: %(hex[0:16])\n
Value: %(int[16:24])\n
Checksum: %(bits[24:])
```



D3	91	12	34	FF	D9
1101 0011	1001 0001	0001 0010	0011 0100	1111 1111	1101 1001

Demo Time

More Complicated Packets?

Packet Structure:

- Sync Word: `0xD391`
- Transmitter ID: `0x1234`
- Length: `5`
- Message: `Hello`
- Checksum: `0xD9`

Demo Time

EVEN MORE Complicated Packets?

Packet Structure:

- Sync Word: **`0xD391`**
- Transmitter ID: **`0x1234`**
- Msg Sequence: **`0`**
- Length: **`5`**
- Message: **`Hello`**
- Checksum: **`0xD9`**

Packet Structure:

- Sync Word: **`0xD391`**
- Transmitter ID: **`0x1234`**
- Msg Sequence: **`1`**
- Length: **`5`**
- Message: **`World`**
- Checksum: **`0xD9`**

Conclusion

Gr-reveng:

- Fixed Length Packets
- Variable Length Packets
- Easy integration w/ scapy
- Custom blocks for complicated packet flows!

Thanks!

Tim K (@bjt2n3904)

https://github.com/tkuester/gr-reveng