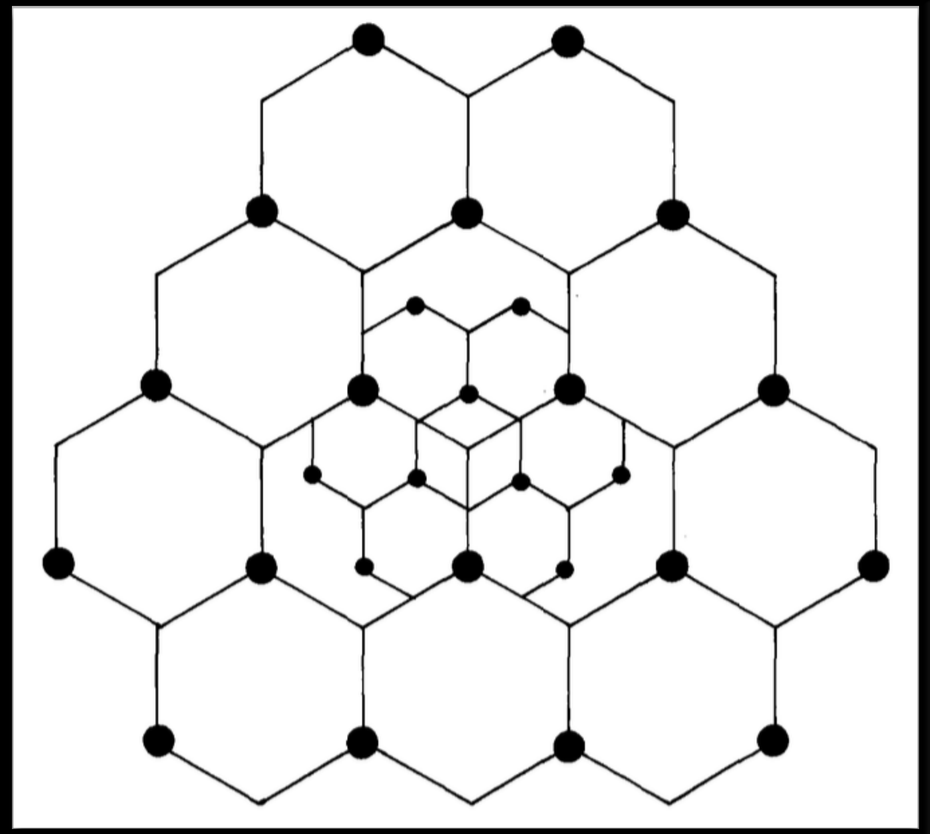
Dig Out Your Brick Phone!
Bringing AMPS Back with GNU Radio

cstone a/k/a Brandon Creighton
Veracode / Ninja Networks
bjc@pobox.com

ShmooCon 2017

Goals

- Provide a quick tour of AMPS (**A**dvanced **M**obile **P**hone **S**ystem)
- Show the steps in creating GNU Radio blocks to prototype parts of a base station (gr-amps)
- Have fun!
- Demonstrate that TX > RX



Why AMPS?

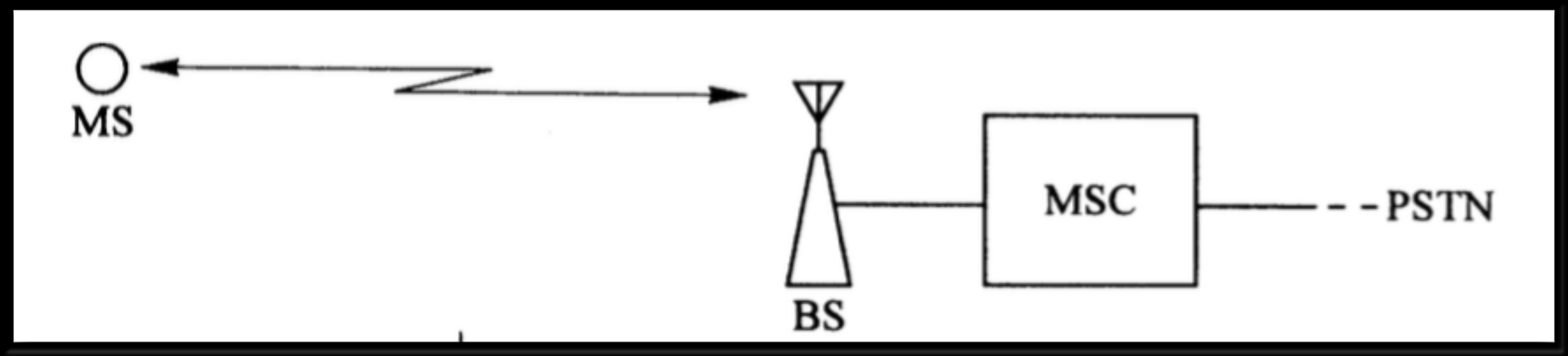
- Equipment easily accessible (ebay)
- Simple enough to be a good target
  - Final BS-MS standard (TIA/EIA-553, 1999): 136 pages
- Historical significance
- Anything still out there?

About AMPS

- First cellular network in the US
  - Designed in the 70s; deployed in 1983
- Extremely dumb phones, smarter base stations
- 824-849 MHz (MS->BS); 869->894
- Tradeoffs: power, spectrum use, cost, security
- Looks like a trunking system
- Often nicknamed “analog” since voice is FM

A bit about GNU Radio

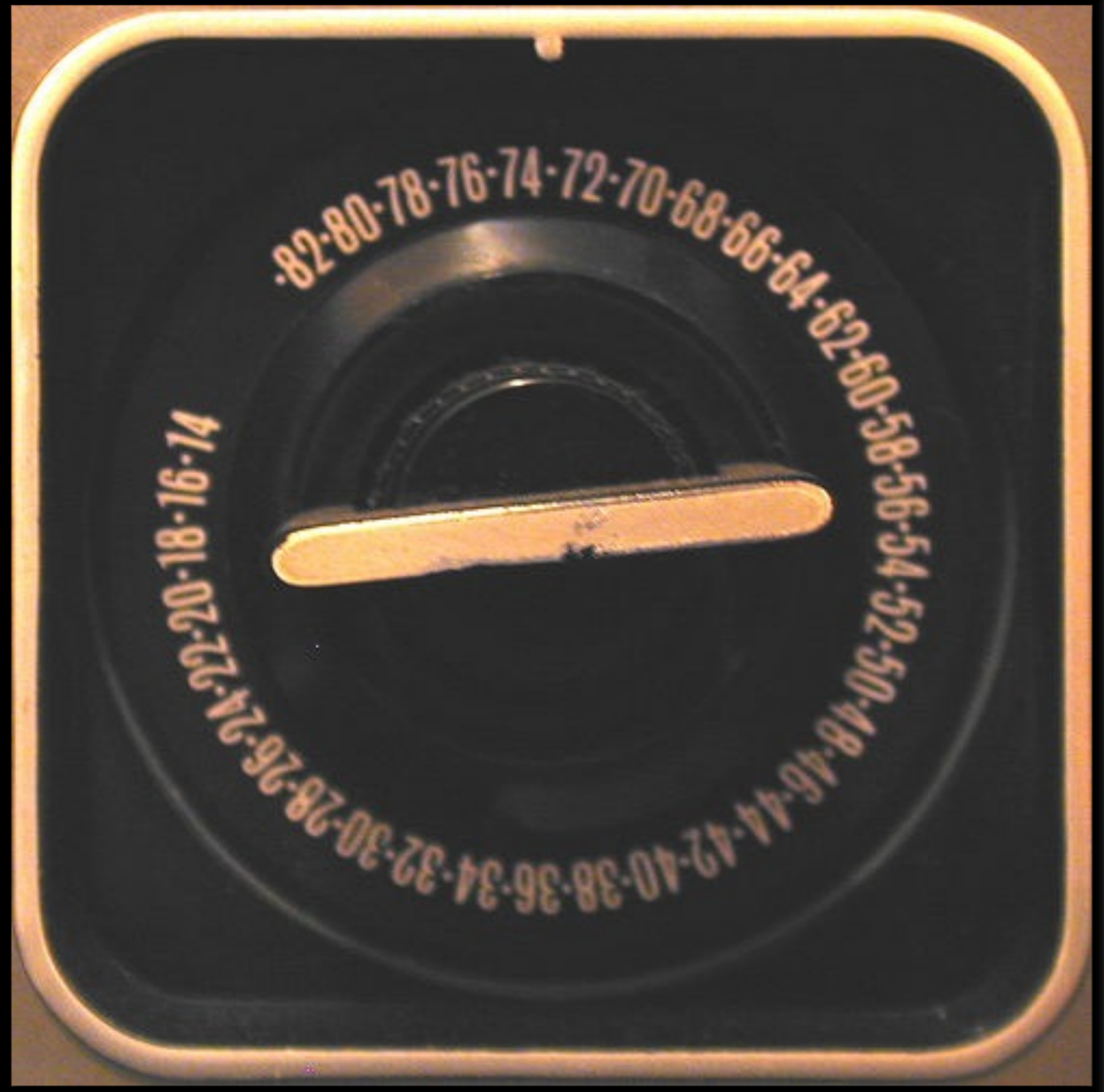
- GR is a library of DSP/RF components (“***blocks***”) mapped to Python interfaces
- Blocks can be native code (C++ API) or Python
- GUI tool (gnuradio-companion, aka GRC) to let you connect them together in ***flowgraphs***, change variables, and generate Python code to run them
- Watch Mike Ossmann’s video series to learn more!



First Steps..

Research!

System	Channel Numbers	Forward Center Freq (BS: Base Station)	Reverse Center Freq (MS: Mobile Station)
A’’	991–1023 (33 channels)	824.040 – 825.000	869.040 – 870.000
A	1–333 (333 channels)	825.030 – 834.990	870.030 – 879.990
B	334–666 (333 channels)	835.020 – 844.980	880.020 – 889.980
A’	667–716 (50 channels)	845.010 – 846.480	890.010 – 891.480
B’	717–799 (83 channels)	846.510 – 848.970	891.510 – 893.970


·82·80·78·76·74·72·70·68·66·64·62·60·58·56·54·52·50·48·46·44·42·40·38·36·34·32·30·28·26·24·22·20·18·16·14


32
34
36
38
40
43
46
49
51
54
56
59
62
66
70
14
15
16
18
20
22
24
26
28
30
RETROTHING.COM

AMPS Channel Types

Type	Forward (BS -> MS)	Reverse (MS -> BS)
Control (Data only) *Channels 313-333 (A), 334-354 (B)*	**FOCC** (Forward Control Channel)	**RECC** (Reverse Control Channel)
Voice (Data/audio mixed)	**FVC** (Forward Voice Channel)	**RVC** (Reverse Voice Channel)

Critical Parameters

Base Station

- **SID:** The 15-bit network ID that the base station belongs to
  - Assigned to telcos; manually programmed in to the device
    - If it doesn’t match, the phone is roaming.

Mobile Station

- **ESN:** 32-bit (!) serial number, globally unique to each device
- **MIN**: Phone number of the device
- This pair of IDs uniquely IDs an AMPS phone!

Control Channels

What does “connected” mean?

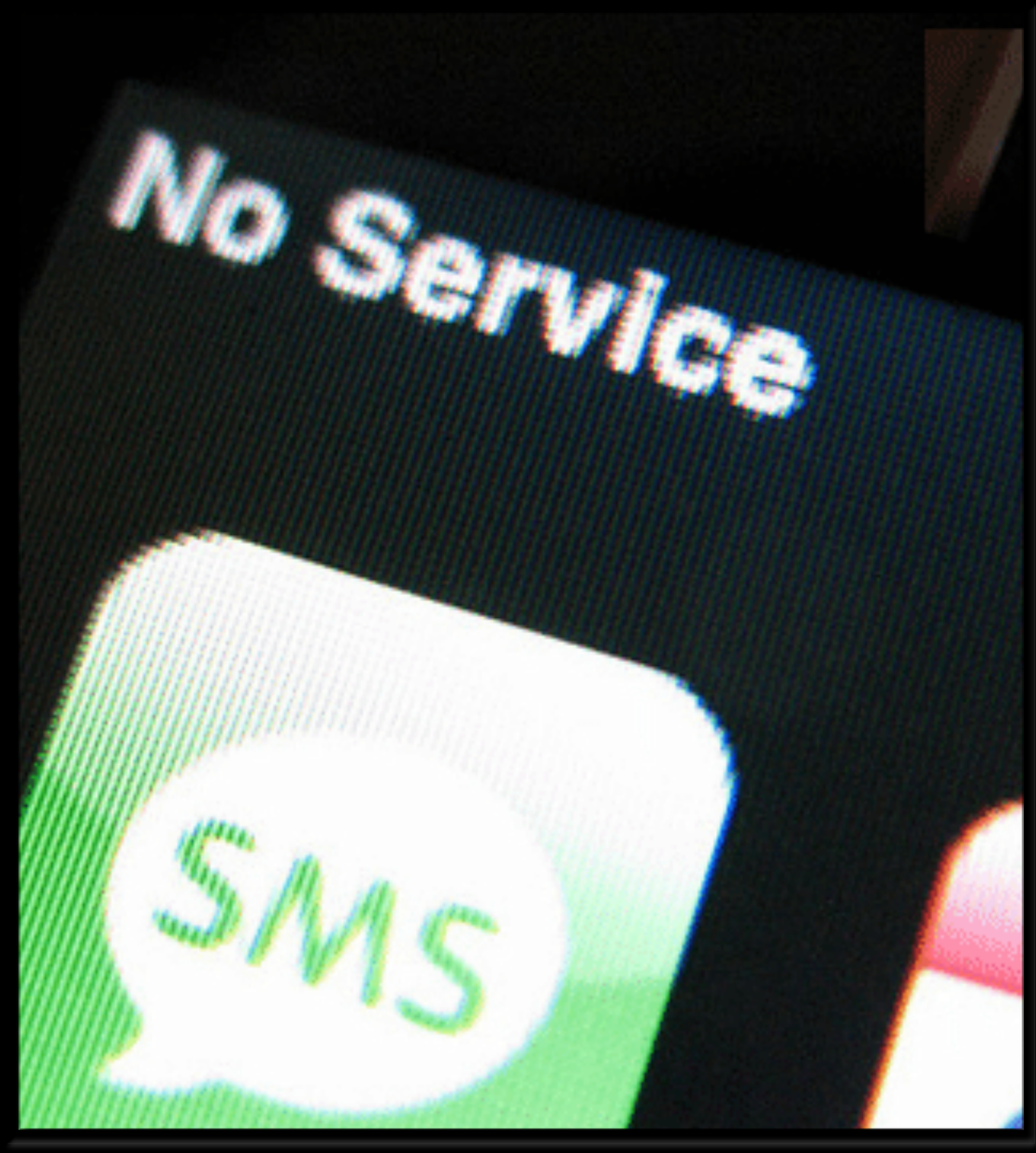
GSM/CDMA/LTE: Registered; acknowledged; authenticated

(exception: emergency calls)

AMPS: Valid control channel received recently, not explicitly disallowing the MS

Registration is unidirectional; acknowledgement not required

(ref: Section 2.6.3.3: “Response to mobile station messages”)



Control Channels

- **Forward (FOCC) transmits constantly:**

1. Broadcast info about the system (Overhead Message)
   - **“Yooo! You’re listening to SID 31337. Here are my settings”**
2. Direct **all** MSes to do something (Overhead Message)
   - e.g. “**Hey, I’m busy, try only 3 times to get a hold of me before waiting”**
3. Direct a **specific MS** to do something (Mobile Station Control Message)
   - e.g. “**MS 6668675309! Tune to FVC 42”**
4. Filler words (Control-Filler Message) — for blank space

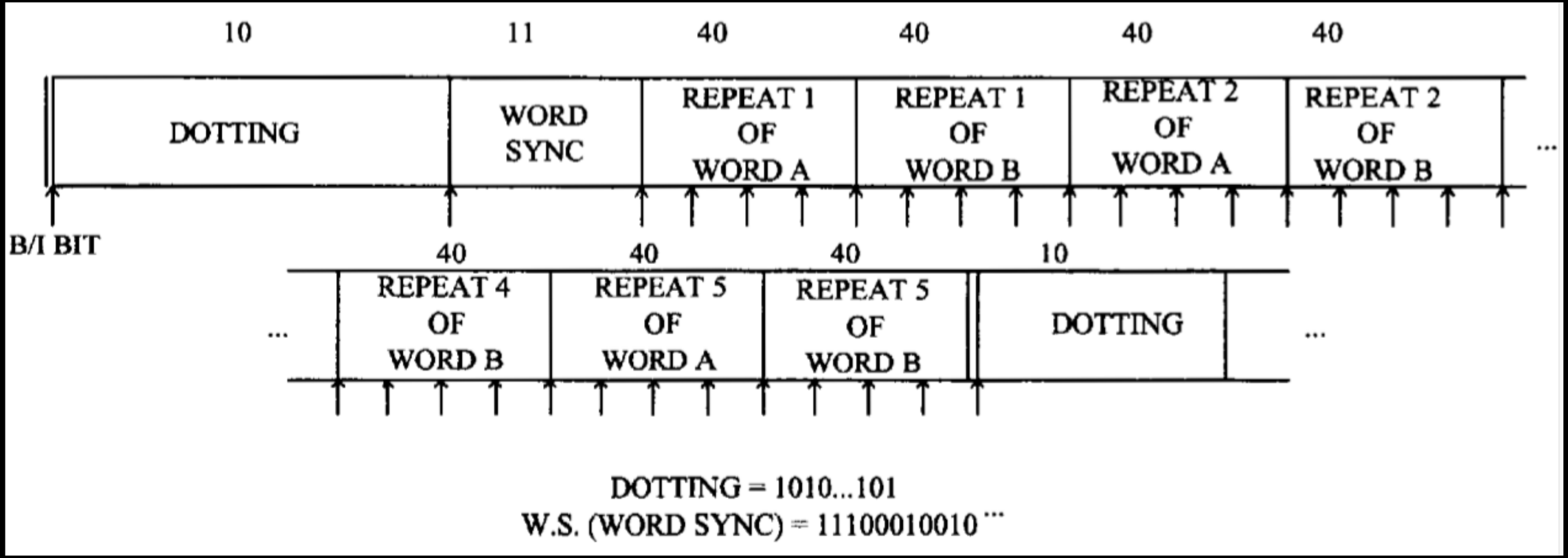
Building an FOCC Stream

All messages are broken up into 28-bit **words**

Every word includes an additional 12 bits of error correction

Truncated BCH(63, 51); protects against 2 bit flips

FOCC Stream Format (Frame)



Phones where LSB(MIN) = 0 listen to word A

Busy/Idle bit sent before dotting/sync, after sync, and every 10 bits after

Building an FOCC Stream

Word 1

T1T2 = 11	DCC	SID1	EP	AUTH	PCI	NAWC	OHD = 110	P
2	2	14	1	1	1	4	3	12

Word 2

T1T2=11	DCC	S	E	REGH	REGR	DTX	
2	2	1	1	1	1	2	

	N-1	RCF	CPA	CMAX-1	END	OHD=111	P
	5	1	1	7	1	3	12

Building an FOCC Stream

- First step: We want to build a static FOCC saying that there's a network out there; everything else can be filler
  - The System Parameter Overhead Message (two words) does this
- Ordering them
  - TIA/EIA-553 3.7.1.2 says that this message must be sent every 0.8 +/- 0.3s
    - Data is sent at 10 kbit/s, which means if we repeat every 18 or 19 words, we're OK:
      - 18 x 463 bits = 8334 bits (0.8334s)
      - 19 x 463 bits = 8797 bits (0.8797s)

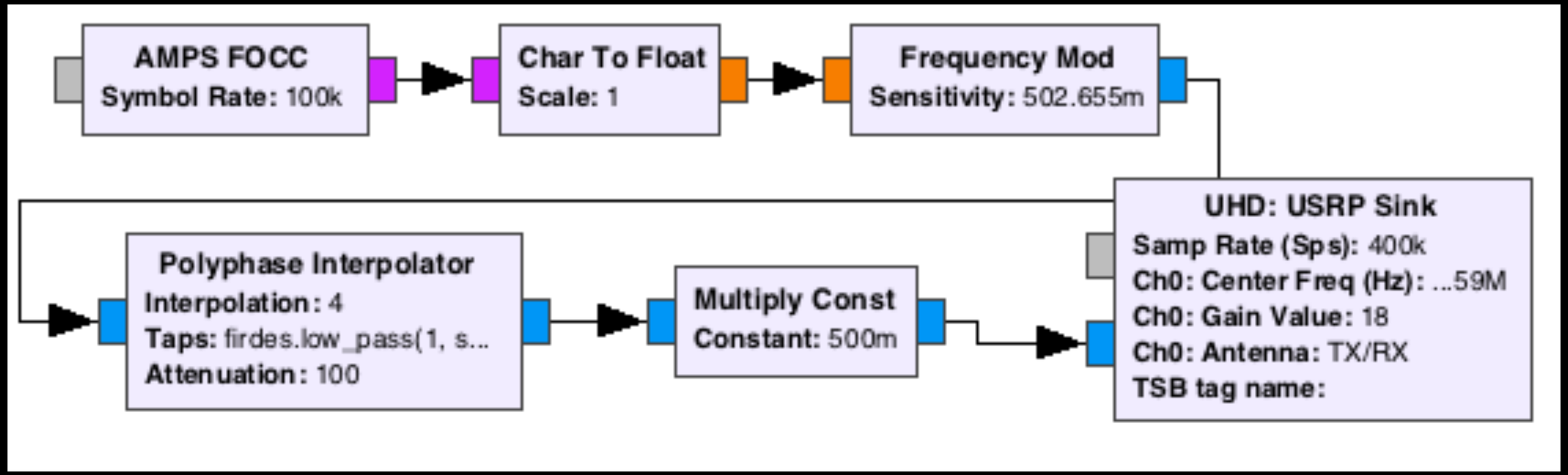
0	SYSTEM PARAMETER OVERHEAD MSG, WORD 1
1	SYSTEM PARAMETER OVERHEAD MSG, WORD 2
2	filler
3	filler
4	filler
5	filler
6	filler
7	filler
8	filler
9	filler
10	filler
11	filler
12	filler
13	filler
14	filler
15	filler
16	filler
17	filler

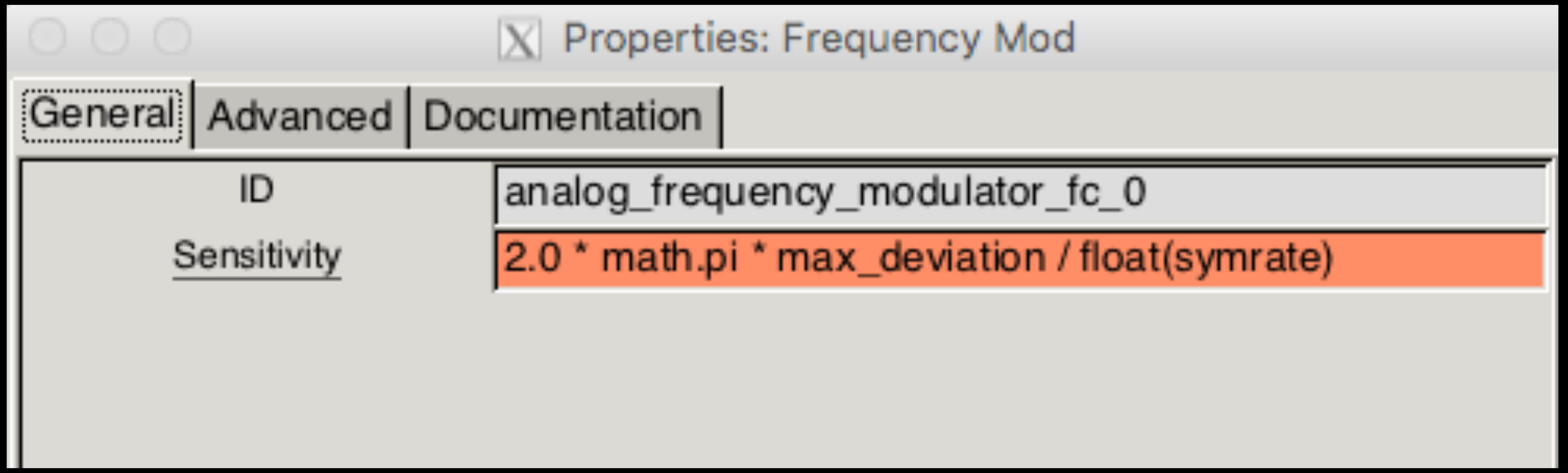
FOCC - Modulation

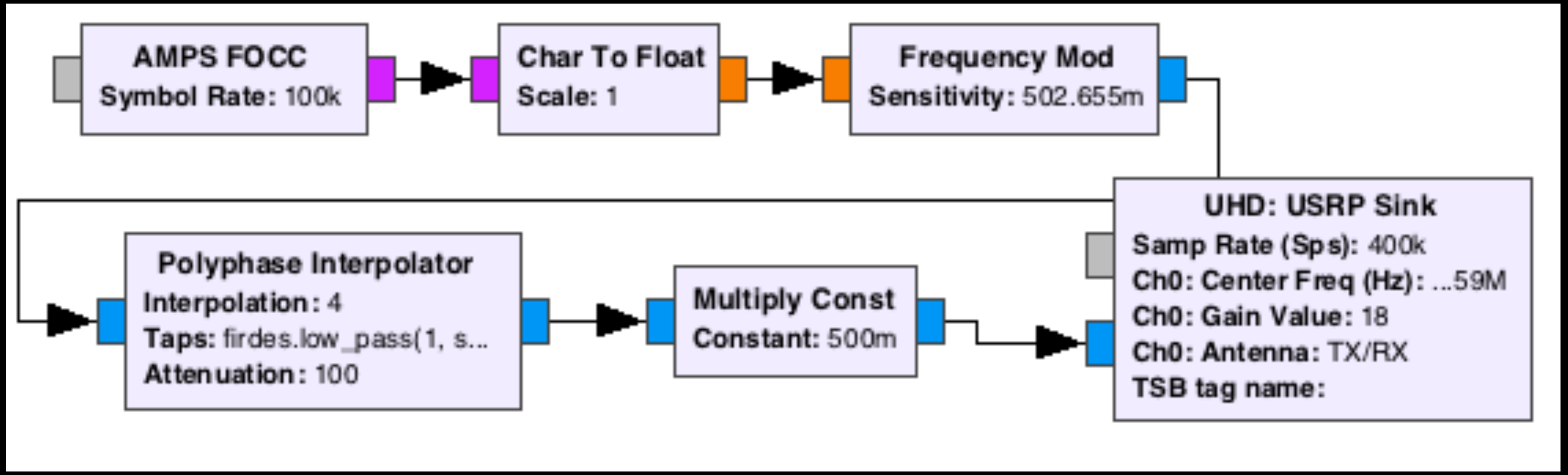
- **Forward (FOCC):**
  - **Repeated broadcast of BFSK-modulated data**
    - **8kHz deviation**
    - **Phase doesn’t matter (not CPFSK)**
    - **10kbit/sec, but Manchester encoded(!), so 20k symbols/sec**
      - **To send a 0: send 1, then 0**
      - **To send a 1: send 0, then 1**

Modulating BFSK (aka 2FSK) in GNU Radio

- **Create a source block that emits values of -1 or 1, each representing a 0 or 1 symbol**
- **Multiply those values by the deviation**
- **Either emit (`sample_rate / 10000`) symbols from the block, or interpolate**




Properties: Frequency Mod
General
Advanced
Documentation
ID
analog_frequency_modulator_fc_0
Sensitivity
2.0 * math.pi * max_deviation / float(symrate)


AMPS FOCC
Symbol Rate: 100k
Char To Float
Scale: 1
Frequency Mod
Sensitivity: 502.655m
UHD: USRP Sink
Samp Rate (Sps): 400k
Ch0: Center Freq (Hz): ...59M
Ch0: Gain Value: 18
Ch0: Antenna: TX/RX
TSB tag name:
Polyphase Interpolator
Interpolation: 4
Taps: firdes.low_pass(1, s...
Attenuation: 100
Multiply Const
Constant: 500m

Demo - FOCC only

RECC - Reverse Control Channel

Shared channel - phones send bursts of data when they need to

1. Send Acknowledgements (Order Confirmation / Page Response messages)
2. Request something from the BS (Order message)
   - Primarily: registration/authentication
3. Placing a call (Origination messages)

RECC Burst Format

DOTTING	WORD SYNC	CODED DCC*	FIRST WORD REPEATED 5 TIMES	SECOND WORD REPEATED 5 TIMES	THIRD WORD REPEATED 5 TIMES	...
30	11	7	240	240	240	

Seizure Precursor

DOTTING = 1010...010

WORD SYNC = 11100010010

* DIGITAL COLOR CODE - Coded per Table 2.7.1-1.

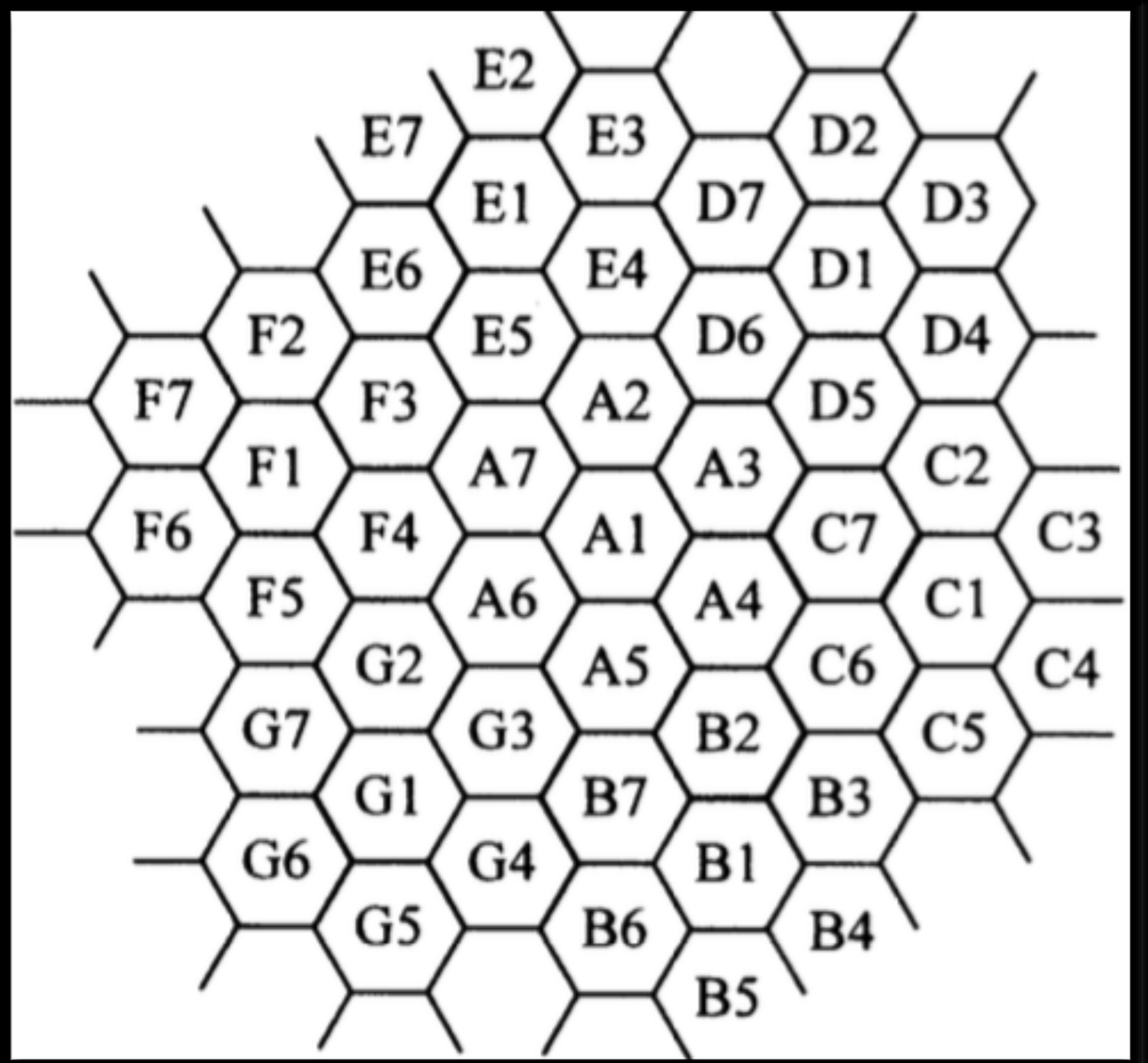
Same basic idea as the FOCC — repetition

Color Codes?

AMPS is a *cellular* network; there are meant to be other BSes around (on the same channel)!

Each BS sends its DCC in the System Parameter Overhead Message; MSes use that to talk to them

Neighboring BSes have *different* DCC values, and ignore

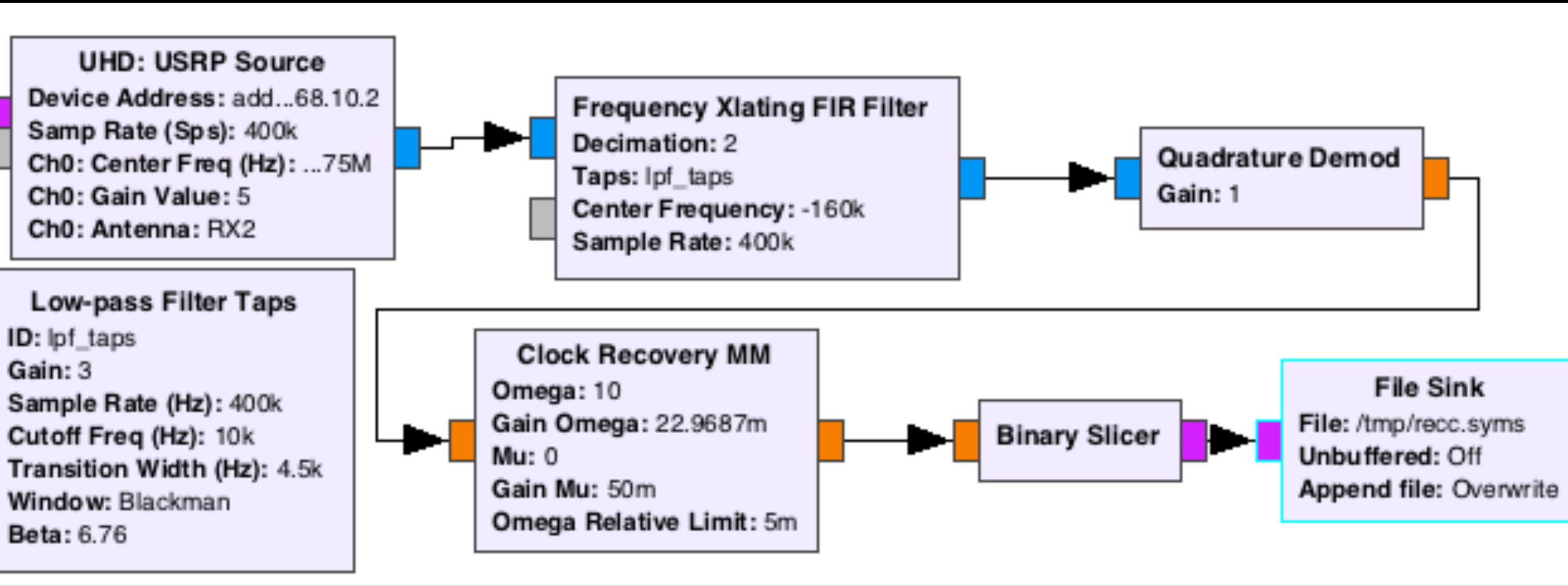


Looking For a RECC Burst with GR

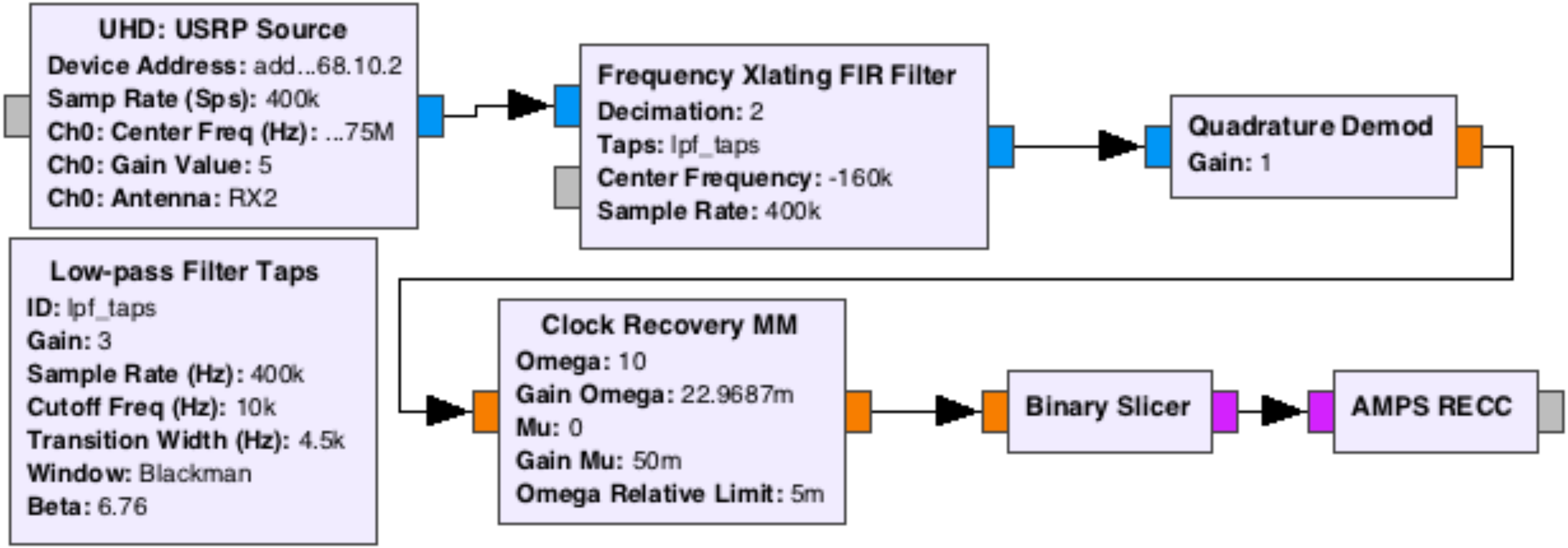
1. Start with an RF source (USRP Source, Osmocon Source, etc).
   - (To avoid hassles, use a frequency offset to avoid the DC spike)
2. Isolate the signal with a filter / demodulate as necessary
3. Demodulate, do clock recovery to align samples with the bitstream
   - Fairly easy if you have a good idea of the symbol rate!
4. Extract bits, collect bits in a custom block

Clock Recovery Visualization

RECC Demod Flowgraph



RECC Demod Flowgraph



RECC Burst Format

DOTTING	WORD SYNC	CODED DCC*	FIRST WORD REPEATED 5 TIMES	SECOND WORD REPEATED 5 TIMES	THIRD WORD REPEATED 5 TIMES	...
30	11	7	240	240	240	

Seizure Precursor

DOTTING = 1010...010

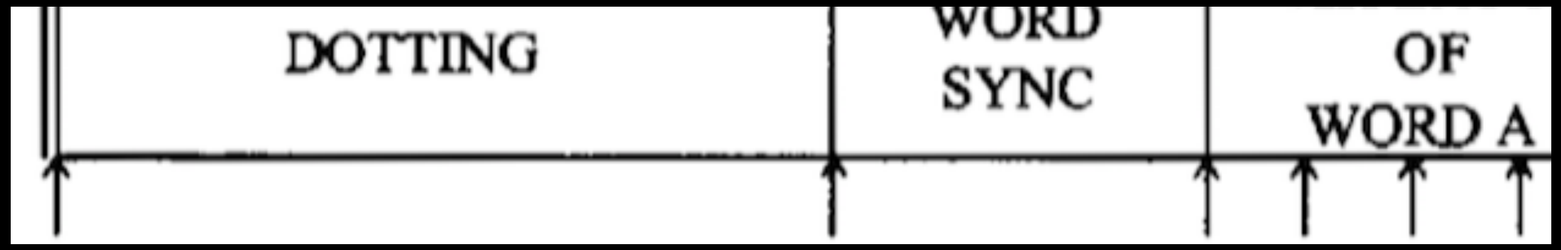
WORD SYNC = 11100010010

* DIGITAL COLOR CODE - Coded per Table 2.7.1-1.

Same basic idea as the FOCC — repetition

Seizing the RECC

To avoid collisions, MSes monitor the busy-idle bits sent every ~10 bits in the FOCC



MSes transmit when they see idle bits, but continue listening while TXing

- If MS sees that the channel is busy before the first 56 bits are sent, stop
- If MS *fails to see* the channel go busy before the first 104 bits, stop

(Full procedure described in section 2.6.3.5)

Building a Better FOCC Block

Original FOCC block's `work()` function just emitted all the bits to a superframe every time it was called with all B/I bits set to idle)

That's not good enough!

New design stores the data as a series of segments (either messages or B/I bits), and `work()` only sends one segment per call (side effect: increased CPU)

Hypothesis: the RECC block could set a global `volatile bool` (yes, this works in GR) when the preamble is detected, and that would be close enough to meet the timing requirements

Results: NOPE

Also tried: Thresholds / AM Demod for carrier detection (but timing is tricky)

Giving Up On Doing It Right

Access Type Parameters Global Action Message

T1T2 =11	DCC	ACT= 1001	BIS	PCI HOME	PCI ROAM	BSPC	BSCA P	RSVD	END	OHD =100	P
2	2	4	1	1	1	4	3	6	1	3	12

- Turns out, you can just tell the phones to ignore the B/I bits.
- Set BIS = 0 and they won’t care.

0	SYSTEM PARAMETER OVERHEAD MSG, WORD 1
1	SYSTEM PARAMETER OVERHEAD MSG, WORD 2
2	ACCESS TYPE PARAMETERS GLOBAL MESSAGE
3	filler
4	filler
5	filler
6	filler
7	filler
8	filler
9	filler
10	filler
11	filler
12	filler
13	filler
14	filler
15	filler
16	filler
17	filler

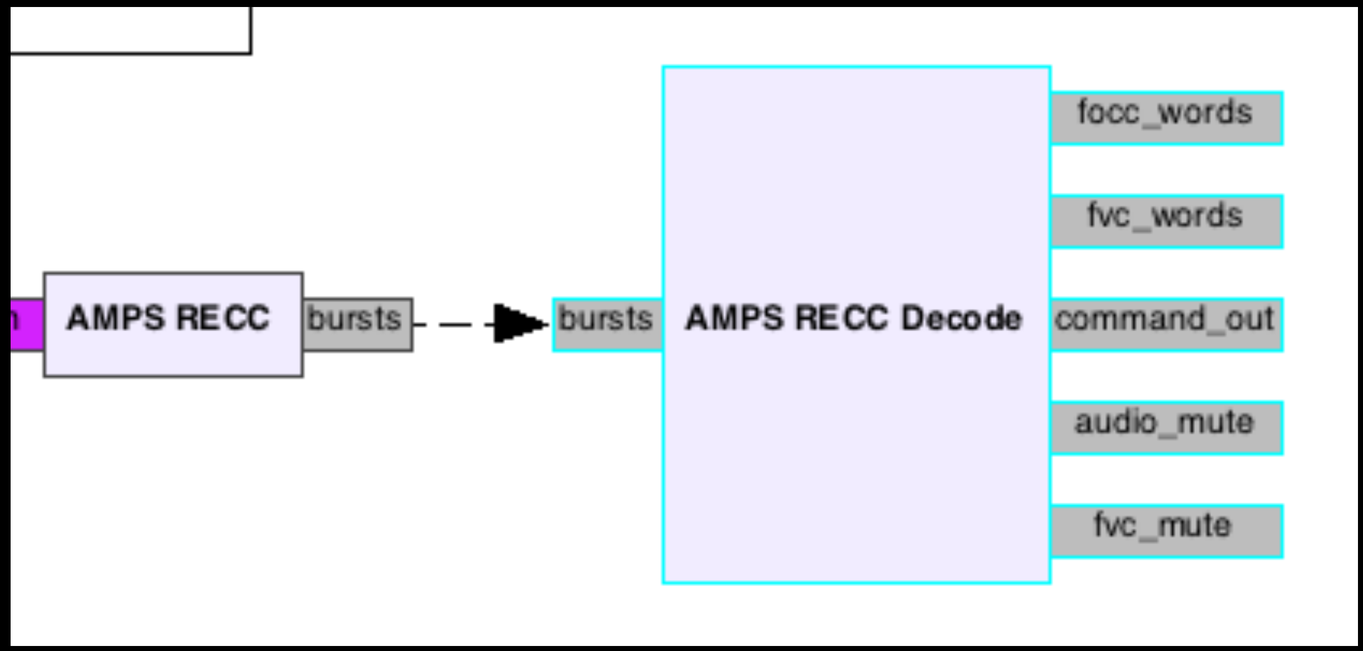
Lazy Preamble Detector

```
unsigned char *starting_ptr;

int recc_impl::work(...) {
    memcpy(&symbol_buf[curidx], input_buf, input_bytes);
    if(starting_ptr == NULL) {
        starting_ptr = memmem(symbol_buf[curidx - search_sz],
                search_sz, preamble, preamble_len);
    }
    if(starting_ptr != NULL
        && (symbol_buf_end - starting_ptr) >= BURST_SIZE) {
        // send the data off
    }
    // Rotate buffers as needed / reset starting_ptr to NULL
}
```

GNU Radio Messages

- GNU Radio allows blocks to define message ports, to send bits of data between blocks in an asynchronous way
  - Don’t tie up `work()` methods; send messages to a new block instead
  - Used for RECC bursts, outgoing data messages, etc.



Answering Calls

When a call is placed from an MS, it sends an Origination RECC message.

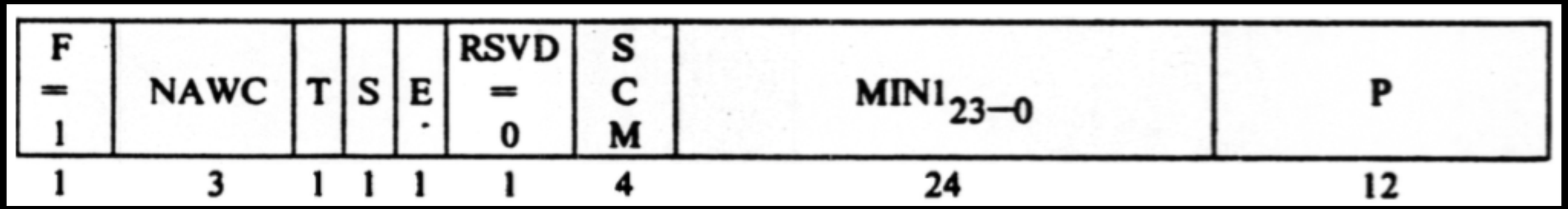
Goal: Answer it, redirect it to a voice channel playing a static looping audio file.

Mobile-Originated Call Setup Process

1. **MS sends Origination message over the RECC, including ESN/MIN and dialed number**
2. BS sends a Mobile Station Control Message with SCC != 11 and a voice channel number/RF power level
3. MS tunes to that FVC for incoming audio/messages
4. MS starts broadcasting audio on the corresponding RVC

Origination Message

All RECC messages start with Word A:



F = 1	NAWC	T	S	E	RSVD = 0	SCM	$MIN1_{23-0}$	P
1	3	1	1	1	1	4	24	12

If T=1, the MS is asking the BS for something; if T=0, it is acknowledging.

E=1 indicates that the next word is Word B

S=1 indicates that Word C is sent

NAWC: Number of Additional Words Coming

Origination Message

Word B tells us what type of order we're talking about, so it's usually there:

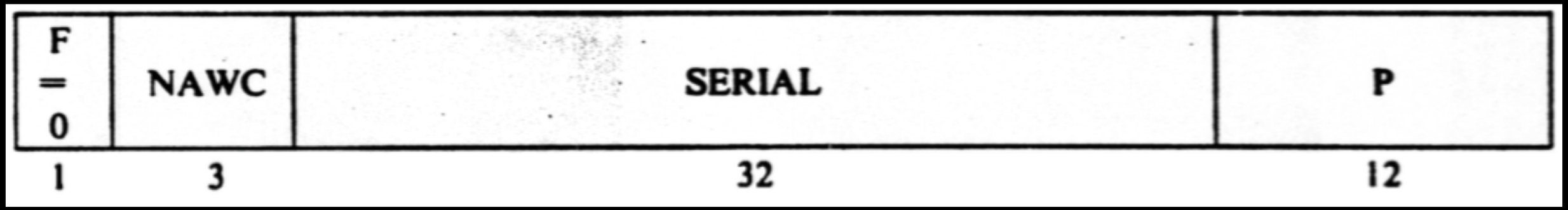
F = 0	NAWC	LOCAL	ORDQ	ORDER	LT	RSVD = 000...0	$MIN2_{33-24}$	P
1	3	5	3	5	1	8	10	12

The ORDQ/ORDER/LOCAL fields indicate order type

In the case of an Origination: all three will be 0

Origination Message

Word C (sent unless the BS says not to in the parameters) gives the ESN:



F = 0	NAWC	SERIAL	P
1	3	32	12

```
                unsigned char nawc = worda.NAWC-2;
                if(wordc.NAWC != nawc) {
                    LOG_WARNING("protocol violation!  Word C NAWC does not agree with Word
A's -- continuing anyway");
                }
```

Origination Message

For an origination, words D and beyond are just the dialed phone number:

F = 0	NAWC	1st DIGIT	2nd DIGIT	...	...	...	...	7th DIGIT	8th DIGIT	P
1	3	4	4	4	4	4	4	4	4	12

Word A (T=1, S=1, E=1)
Word B (ORDER=ORDQ=0)
Word C (ESN)
Word D = 31337

Mobile-Originated Call Setup Process

1. MS sends Origination message over the RECC, including ESN/MIN and dialed number
2. **BS sends a Mobile Station Control Message with SCC != 11 and a voice channel number/RF power level**
3. MS tunes to that FVC for incoming audio/messages
4. MS starts broadcasting audio on the corresponding RVC

Mobile Station Control Message (FOCC)

BS replies to the MS's Origination request with a two-word channel assignment message: (VMAC = power level; CHAN = channel; SCC=color)

T1T2	DCC	$MIN1_{23-0}$	P
2	2	24	12

2	2	10	1	5	3	5	12
T1T2 = 10	SCC = 11	$MIN2_{33-24}$	EF = 0	LOCAL/ MSG_TYPE	ORDQ	ORDER	P
	SCC≠11		VMAC		CHAN		
2	2	10	3		11		12

Mobile-Originated Call Setup Process

1. MS sends Origination message over the RECC, including ESN/MIN and dialed number
2. BS sends a Mobile Station Control Message with SCC != 11 and a voice channel number/RF power level
3. **MS tunes to that FVC for incoming audio/messages**
4. **MS starts broadcasting audio on the corresponding RVC**

AMPS Voice Channels

Both FVC and RVC can transmit audio or data (also 10kbit 2FSK)

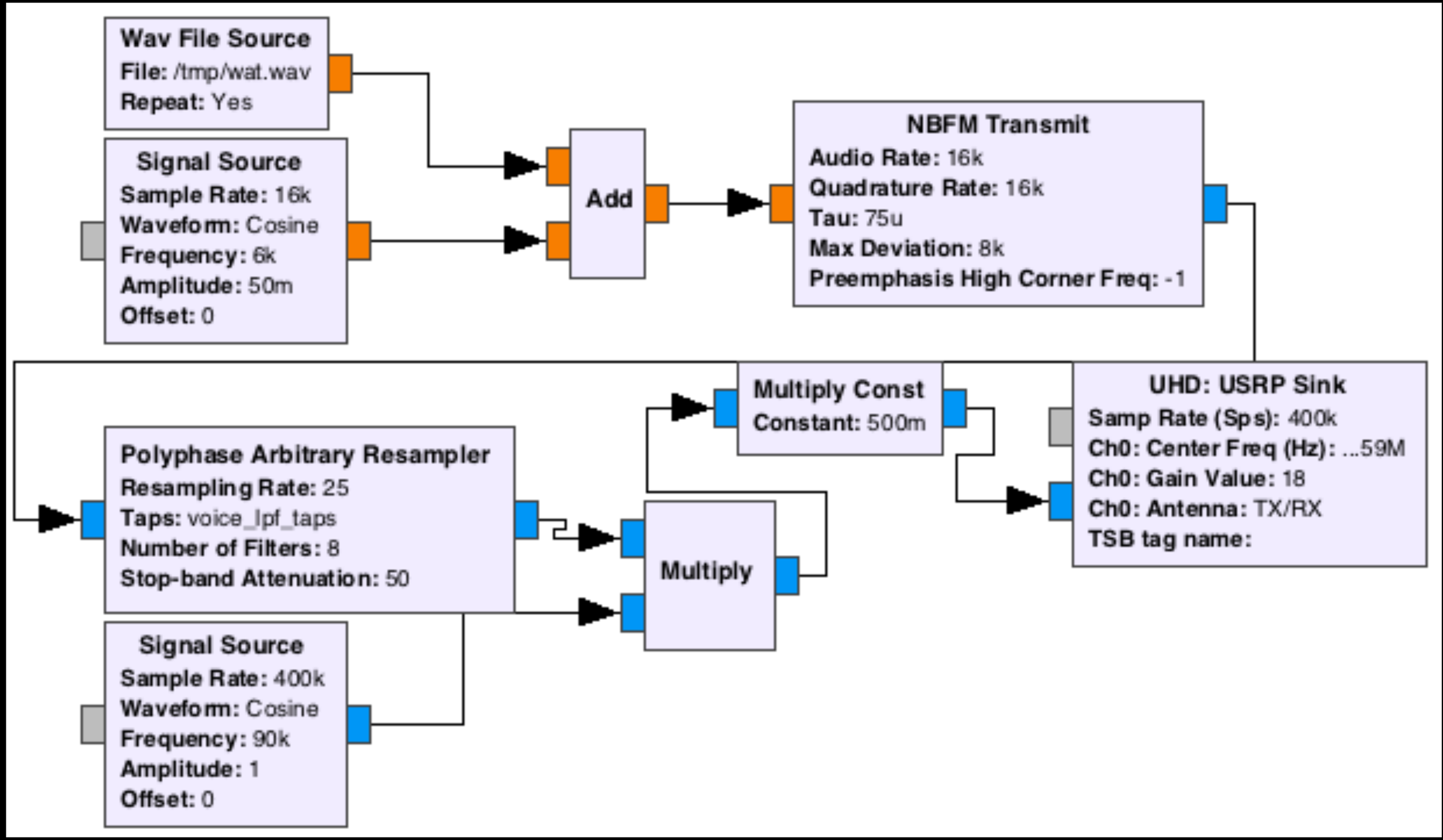
They use ***blank-and-burst*** in-band signaling

When a ~6kHz audio tone (SAT Tone) is present, then audio is being sent; when it's not, data bursts are being sent

SCC value sent from BS determines the frequency to be used

Also important for defending against audio from interfering cells!

SCC	Freq
0	5970
1	6000
2	6030

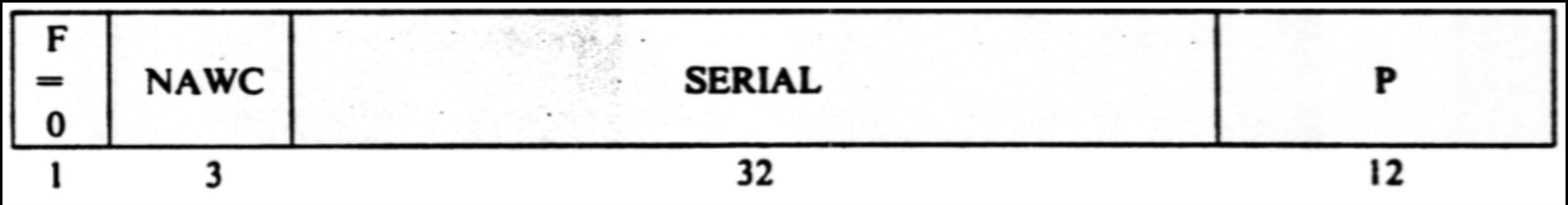

Wav File Source
File: /tmp/wat.wav
Repeat: Yes
Signal Source
Sample Rate: 16k
Waveform: Cosine
Frequency: 6k
Amplitude: 50m
Offset: 0
Add
NBFM Transmit
Audio Rate: 16k
Quadrature Rate: 16k
Tau: 75u
Max Deviation: 8k
Preemphasis High Corner Freq: -1
Multiply Const
Constant: 500m
UHD: USRP Sink
Samp Rate (Sps): 400k
Ch0: Center Freq (Hz): ...59M
Ch0: Gain Value: 18
Ch0: Antenna: TX/RX
TSB tag name:
Polyphase Arbitrary Resampler
Resampling Rate: 25
Taps: voice_lpf_taps
Number of Filters: 8
Stop-band Attenuation: 50
Multiply
Signal Source
Sample Rate: 400k
Waveform: Cosine
Frequency: 90k
Amplitude: 1
Offset: 0

Demo - FVC audio only

Channel Layout

Channel	353	354	355	356
Freq (MHz)	880.59 835.59	880.62 835.62	880.65 835.65	880.68 835.68
Function	FOCC RECC	unused	unused	FVC RVC

Cloning



F = 0	NAWC	SERIAL	P
1	3	32	12

Initially, the RECC had no protection against eavesdropping, making obtaining ESN/MIN pairs easy

Cloning a phone was as simple as changing a phone's ESN

Cloning Countermeasures

Countermeasure 1: Forbid handset manufacturers from allowing the ESN to be programmable

Some were better at it than others.

Cloning Countermeasures

Countermeasure 2: Bolt-on encryption after the fact

TDMA (IS-54 aka ANSI/TIA/EIA-627), aka D-AMPS included the Cellular Message Encryption Algorithm suite (aka CAVE)

Published in the early 90s

Kept private; document describing the algorithm was leaked to Cryptome in 1997

First severe attacks published by David Wagner, Bruce Schneier, and John Kelsey at CRYPTO 97

Cloning Countermeasures

Countermeasure 3: Make it illegal to make or buy a scanner that covered AMPS ranges

This happened.

Telephone Disclosure and Dispute Resolution Act (1992)

Enforced by the FCC

Cloning Countermeasures

Countermeasure 4: Make users type in a PIN to make a call

Some carriers did this!

It raised the cost only slightly: you had to decode the RECC request, the FOCC response, and the audio on the FVC

BS-Originated Call (Page)

1. BS sends Page Message over FOCC (Mobile Station Control Message)
2. MS sends Page Response over RECC (order ack)
3. BS sends a Mobile Station Control Message with SCC != 11 and a voice channel number/RF power level (same message as before!)
4. MS tunes to designated FVC, awaits commands; starts broadcasting silence (no SAT) on RVC
5. BS sends Alert over FVC; MS rings the phone
6. User Answers
7. MS starts broadcasting SAT and audio

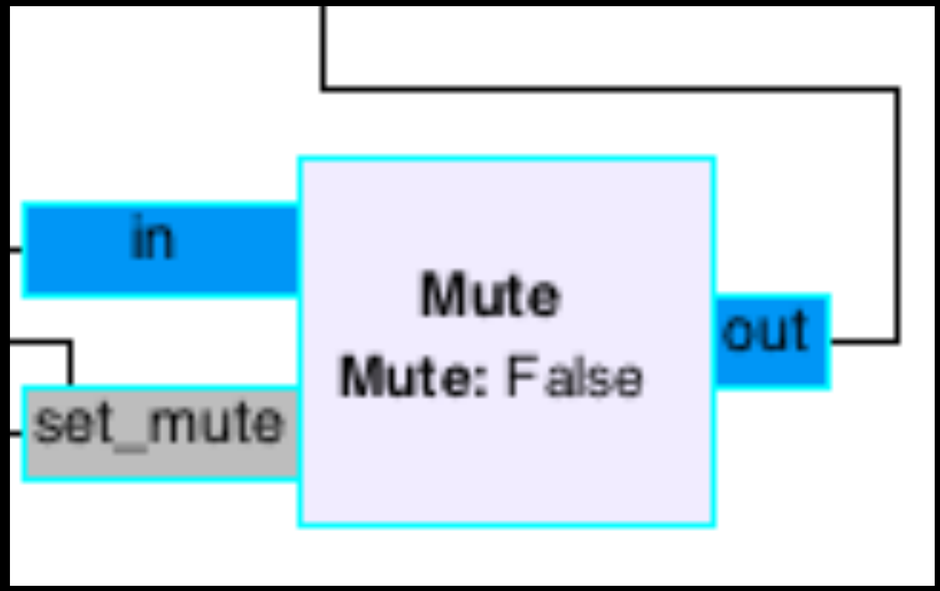
Channel Layout

Channel	353	354	355	356
Freq (MHz)	880.59 835.59	880.62 835.62	880.65 835.65	880.68 835.68
Function	FOCC RECC	unused	FVC RVC	FVC RVC

A basic FVC in GNU Radio

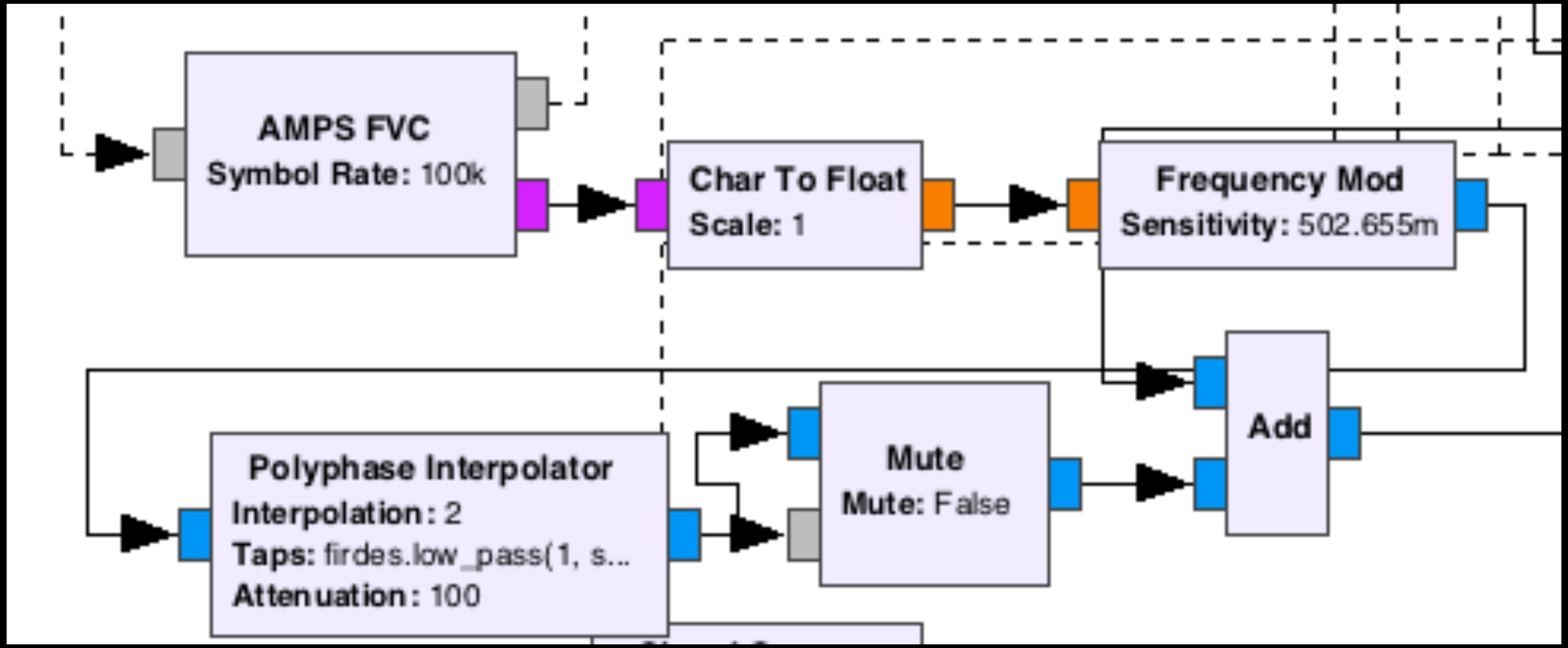
Problem: Need to transmit both data and audio, and choose

Solution: Two paths transmitting; each flows through a Mute block controlled by messages



(There are probably more elegant solutions.)

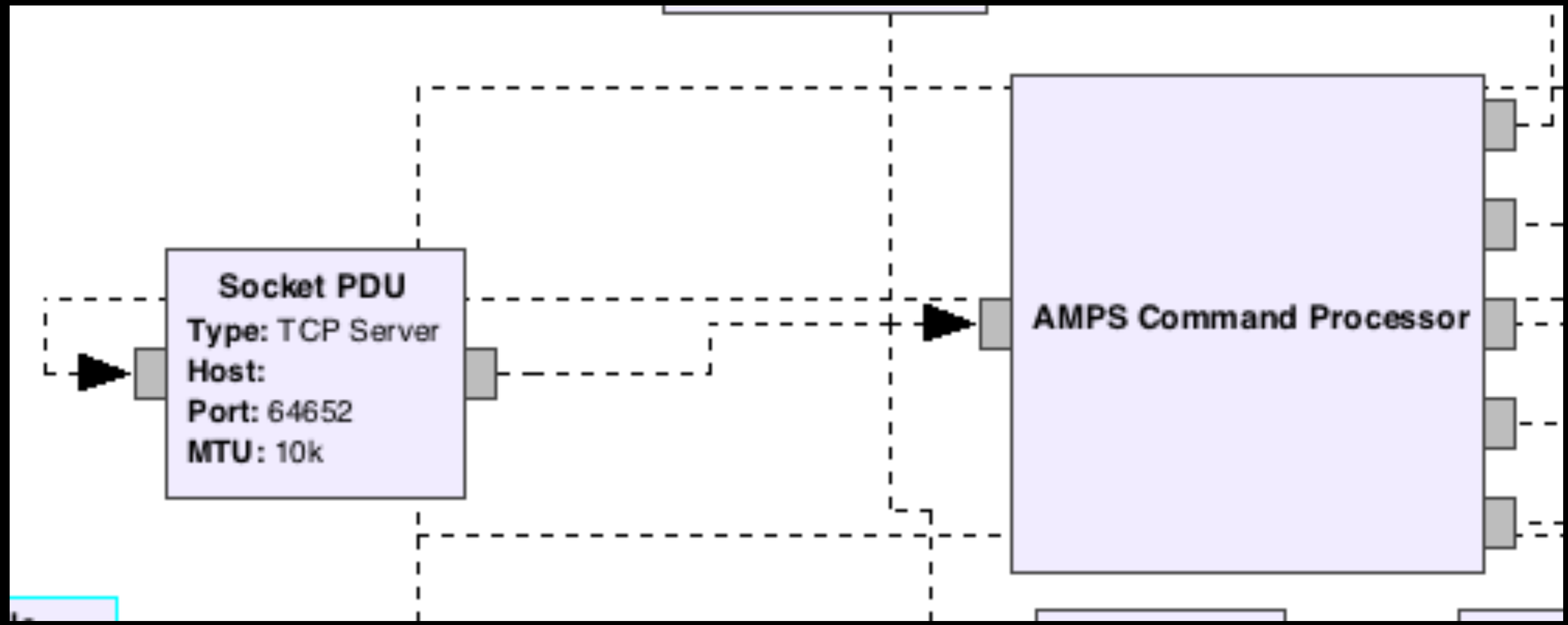
A basic FVC in GNU Radio



Build a CLI with Socket PDU

Want a quick and dirty telnet CLI?

Use Socket PDU and a lightweight parser block



Demo - FVC Page

Future work - RVC

- For mobile-mobile calls: straight sample copying doesn’t seem to work well
  - Caveat: haven’t tried everything yet

- Is it worth continuing to do this in GR, or is it time to move it out?

Packet-In-Packet Protection!

Orders to certain MINs could not be transmitted on AMPS.

Table 3.7.1-5 Troublesome central office codes

Bit pattern thousands					Central office	
T1T2	Digit DCC	NXX	X	XXX	Code	
00	ZZ	111110(0)0100	10YY	...	007	0,8,9
00	ZZ	111011(1)0001	0010	...	056	2
00	ZZ	111100(0)1001	0ZZZ	...	070	1-7
00	ZZ	000011(1)0001	0010	...	150	2
00	ZZ	000111(1)0001	0010	...	224	2
00	ZZ	000111(0)0010	010Z	...	225	4,5
00	ZZ	001011(1)0001	0010	...	288	2
00	ZZ	001110(0)0100	10YY	...	339	0,8,9
00	ZZ	001111(1)0001	0010	...	352	2
00	ZZ	001111(0)0010	010Z	...	353	4,5
00	ZZ	010011(1)0001	0010	...	416	2
00	ZZ	010111(1)0001	0010	...	470	2

BE CAREFUL

This is infrastructure - deprecated and disabled infrastructure, but nonetheless.

Don’t transmit carelessly.

Recap: Don’t Do These Things

- When RXing/decoding Manchester-encoded data, make sure you’re looking for symbols and not bits
- Read the specs carefully
- Be careful where you click in GRC. (Dragging a connection deletes it.)

Thanks!

https://github.com/unsynchronized/gr-amps

References

- The latest BS<->MS standard: TIA/EIA-553-A (1999)
  - Older version published as Canada's RSS 118 Annex A
- Avian's Blog, for practical use/caveats of the Clock Recovery MM block: https://www.tablix.org/~avian/blog/archives/2015/03/notes_on_m_m_clock_recovery/
- Keysight/HP: "Programming techniques for the HP 8920A": http://literature.cdn.keysight.com/litweb/pdf/5964-0159E.pdf

References

- Brian Oblivion: “Cellular Telephony”. Phrack 38:9
- Brian Oblivion: “Cellular Telephony II”, Phrack 40:6
- John G. van Bosse, Fabrizio U. Devetak, Signaling In Telecommunication Networks (2nd ed.)