GNU Radio, an educational tool to teach synchronization and much more

Decoding RDS signal from FM radio

Thomas Lavarenne

Lycée Jean Rostand - Villepinte

29 Mars 2023

Summary

Presentation

Thomas Lavarenne
(thomas.lavarenne@lycee-jean-rostand.fr)

Teacher of applied physics in BTS ("Brevet de technicien supérieur")

Presentation

Thomas Lavarenne
(thomas.lavarenne@lycee-jean-rostand.fr)

Teacher of applied physics in BTS ("Brevet de technicien supérieur")

Equivalent to the second year of a bachelor's degree

Presentation

Thomas Lavarenne
(thomas.lavarenne@lycee-jean-rostand.fr)

Teacher of applied physics in BTS ("Brevet de technicien supérieur")

Equivalent to the second year of a bachelor's degree

Students mainly study computer science, from the physical layer to application creation.

Why I use GNU Radio?

GNU Radio fits particularly well for the study of digital wireless communications

Why I use GNU Radio?

GNU Radio fits particularly well for the study of digital wireless communications

Many do not have the mathematical experience needed in signal processing

Why I use GNU Radio?

GNU Radio fits particularly well for the study of digital wireless communications

Many do not have the mathematical experience needed in signal processing

GNU radio is rather simple handling and it is quickly possible to do very interesting things

Why I use GNU Radio?

GNU Radio fits particularly well for the study of digital wireless communications

Many do not have the mathematical experience needed in signal processing

GNU radio is rather simple handling and it is quickly possible to do very interesting things

Practical sessions that allow you to work on a lot of interesting skills (building an antenna, FM analog demodulation, filtering, synchronization, phase modulation...etc.)

Why I use GNU Radio?

GNU Radio fits particularly well for the study of digital wireless communications

Many do not have the mathematical experience needed in signal processing

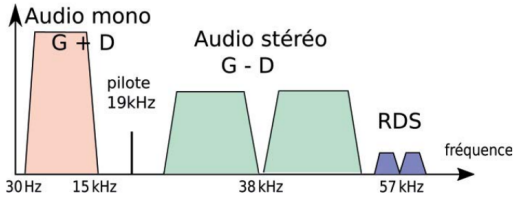
GNU radio is rather simple handling and it is quickly possible to do very interesting things

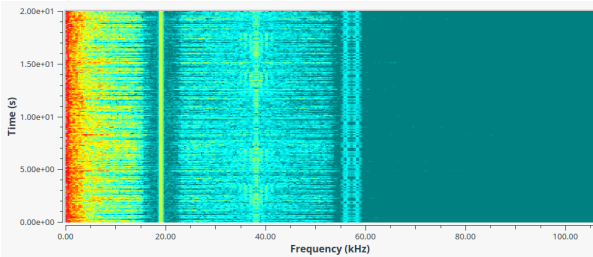
Practical sessions that allow you to work on a lot of interesting skills (building an antenna, FM analog demodulation, filtering, synchronization, phase modulation...etc.)

Isolate difficulties

1. FM Radio

Each FM station consists of a multiplex containing:

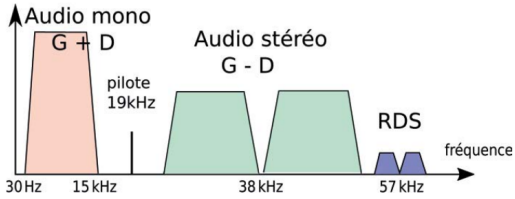


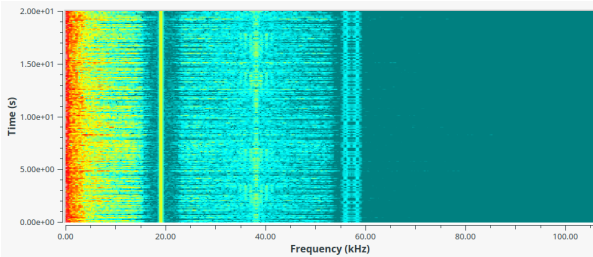


1. FM Radio

Each FM station consists of a multiplex containing:

- Mono informations in baseband

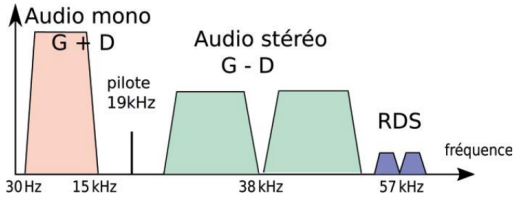


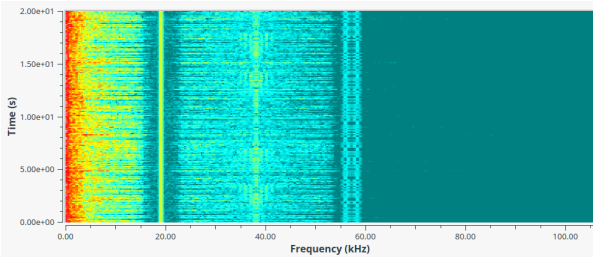


1. FM Radio

Each FM station consists of a multiplex containing:

- Mono informations in baseband
- a pilot at 19 kHz

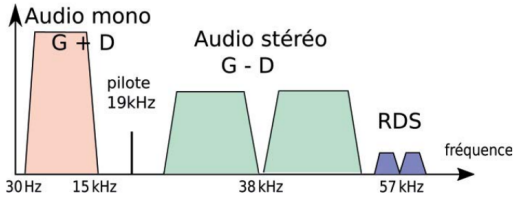


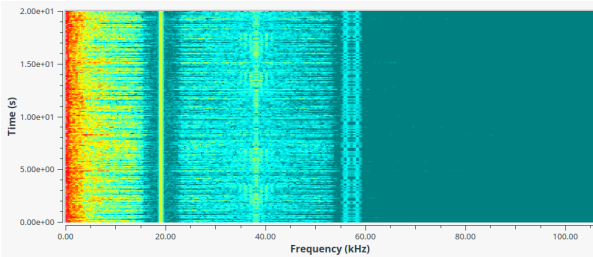


1. FM Radio

Each FM station consists of a multiplex containing:

- Mono informations in baseband
- a pilot at 19 kHz
- Stereo information around a 38 kHz sub-carrier

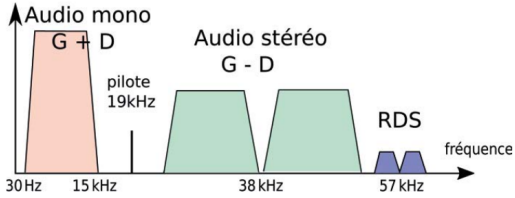


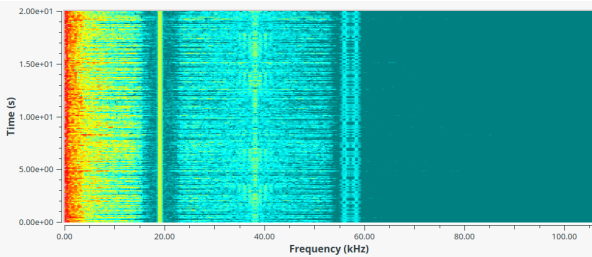


1. FM Radio

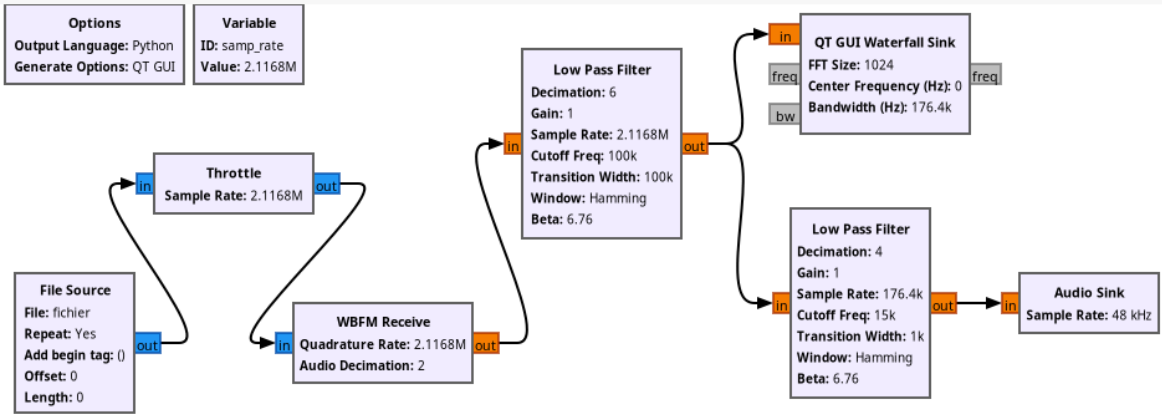
Each FM station consists of a multiplex containing:

- Mono informations in baseband
- a pilot at 19 kHz
- Stereo information around a 38 kHz sub-carrier
- the RDS signal around the subcarrier at 57 kHz



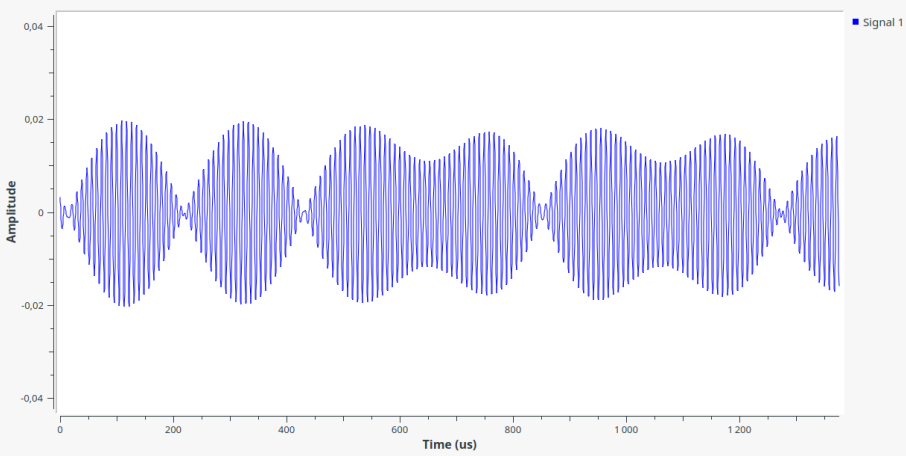


Listen to the radio:

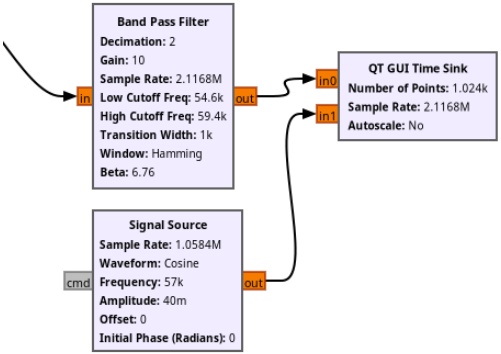


2. RDS signal extraction

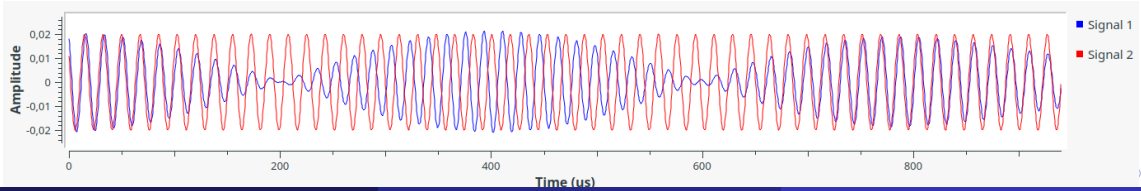
Filter around 57 kHz and get the modulated RDS signal:



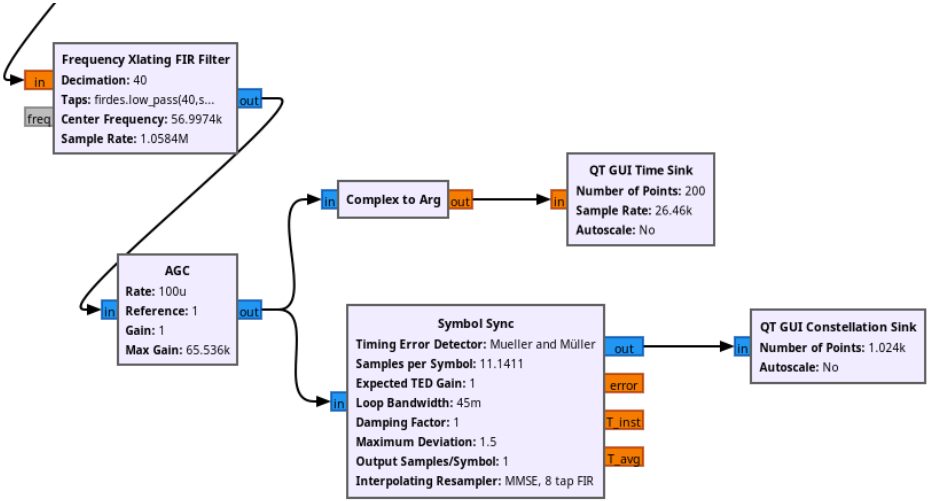
BPSK Modulation?



Superimposing a reference signal at 57kHZ with the signal source block:

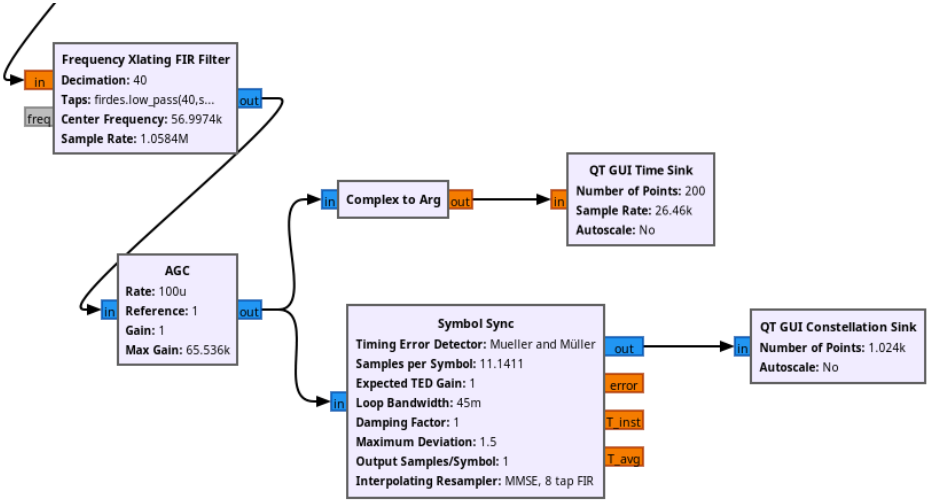


Transfer the signal to baseband: frequency translation



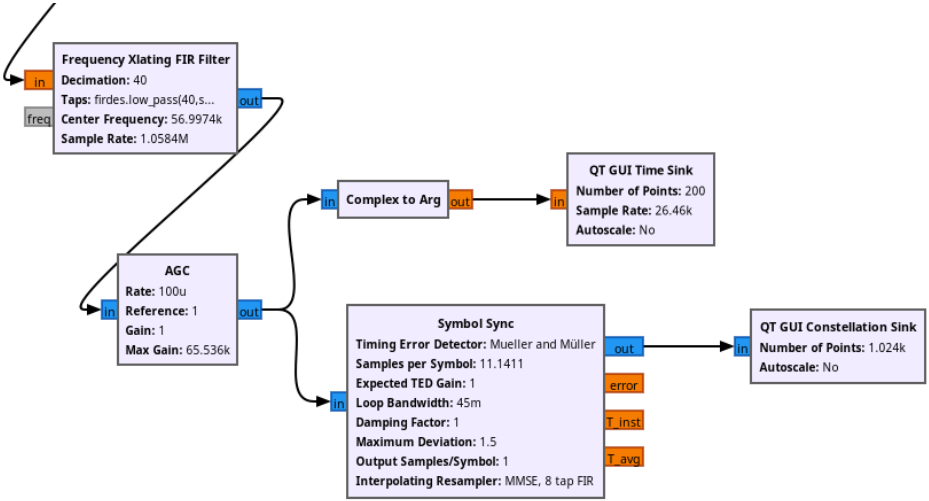
Transfer the signal to baseband: frequency translation

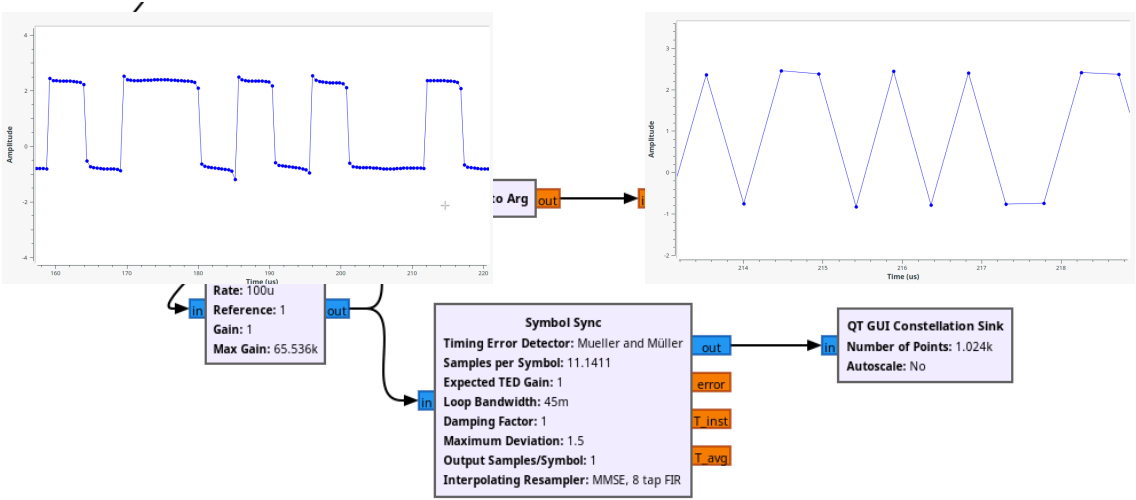
- Extract the phase with the **Complex to Arg** block

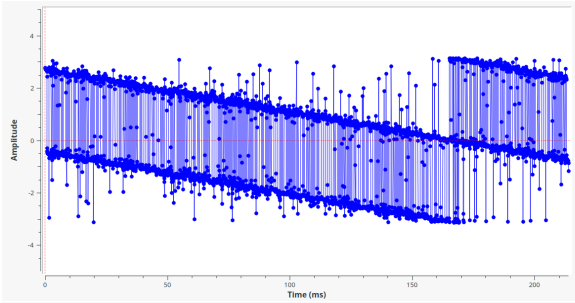


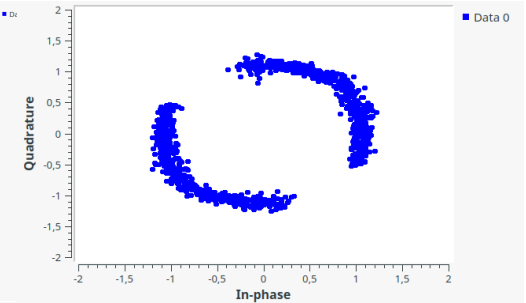
Transfer the signal to baseband: frequency translation

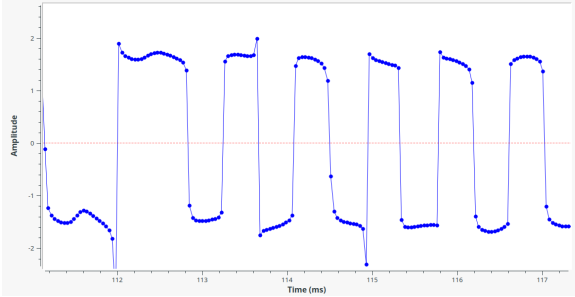
- Extract the phase with the **Complex to Arg** block
- Plot the constellation after symbol synchronization

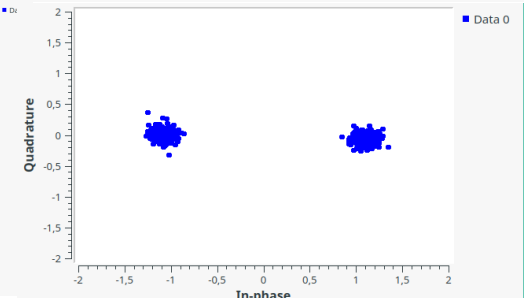



Amplitude
Time (us)
o Arg
out
Amplitude
Time (us)
Rate: 100u
Reference: 1
Gain: 1
Max Gain: 65.536k
in
out
Symbol Sync
Timing Error Detector: Mueller and Müller
Samples per Symbol: 11.1411
Expected TED Gain: 1
Loop Bandwidth: 45m
Damping Factor: 1
Maximum Deviation: 1.5
Output Samples/Symbol: 1
Interpolating Resampler: MMSE, 8 tap FIR
in
out
error
T_inst
T_avg
QT GUI Constellation Sink
Number of Points: 1.024k
Autoscale: No
in


Amplitude
Time (ms)


Data 0
Quadrature
In-phase


Amplitude
Time (ms)


Data 0
Quadrature
In-phase

Conclusion:

- Manual synchronization has only an educational purpose

Conclusion:

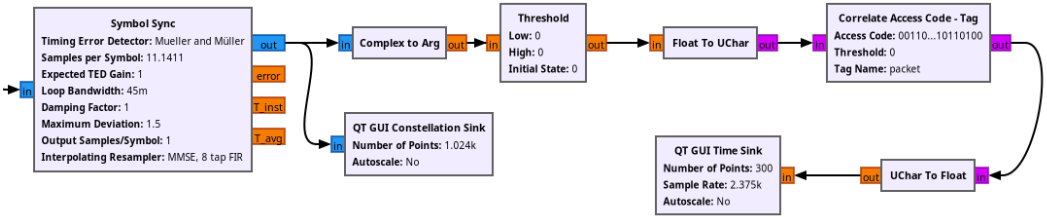
- Manual synchronization has only an educational purpose
- We use the Costas Loop block

Conclusion:

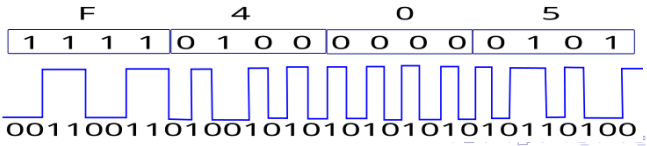
- Manual synchronization has only an educational purpose
- We use the Costas Loop block
- Illustrate and understand notions of synchronization without going into great mathematical details

3. Viewing and slicing Frames

To retrieve and visualize the frames, first we add a threshold and try to synchronize the data stream on the station code, in this example it is "France bleu Alsace" whose code is F405



in differential manchester:



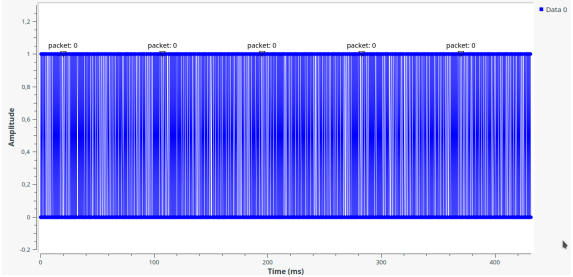
GNU Radio is open source!

Everything is on github:

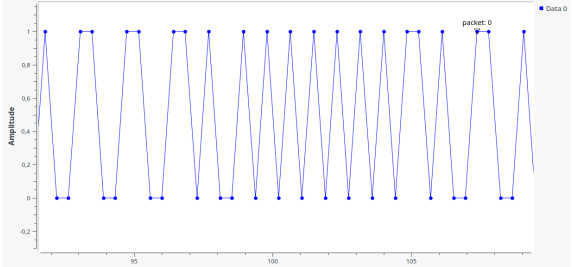
https://github.com/gnuradio/gnuradio/blob/master/gr-digital/lib/correlate_access_code_bb_ts_impl.cc

```
137             while (count < noutput_items) {
138                 // shift in new data
139                 d_data_reg = (d_data_reg << 1) | ((in[count++]) & 0x1);
140                 if (d_data_reg_bits + 1 < d_len) {
141                     d_data_reg_bits++;
142                     continue;
143                 }
144                 // compute hamming distance between desired access code and current
145                 // data
146                 uint64_t wrong_bits = 0;
147                 uint64_t nwrong = d_threshold + 1;
148
149                 wrong_bits = (d_data_reg ^ d_access_code) & d_mask;
150                 volk_64u_popcnt(&nwrong, wrong_bits);
151
152                 if (nwrong <= d_threshold) {
153                     enter_have_sync();
154                     break;
155                 }
156             }
157             break;
```

Results:

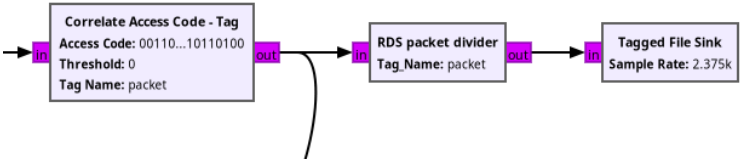


Various frames are detected.

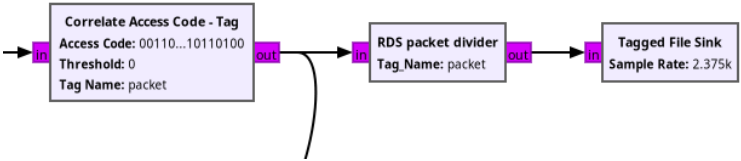


The 'packet' tag added by the correlate - access code block permit to synchronize the visualization on the Time Sink block.

To process the frames, we choose to cut them and send them to separate files on the hard disk for a post-processing in python.

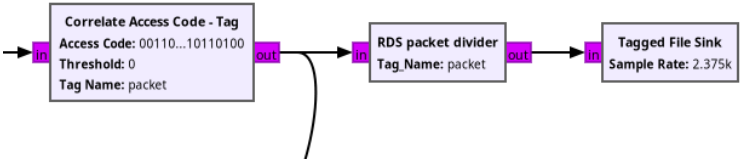


To process the frames, we choose to cut them and send them to separate files on the hard disk for a post-processing in python.



Tagged File sink block works with a '**burst**' tag:

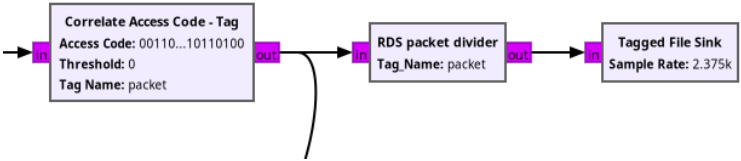
To process the frames, we choose to cut them and send them to separate files on the hard disk for a post-processing in python.



Tagged File sink block works with a '**burst**' tag:

if it detects a '**burst**' tag with **PMT 'True'**, it opens a new file and saves all the data until it detects a new '**burst**' tag with **PMT 'False'**.

To process the frames, we choose to cut them and send them to separate files on the hard disk for a post-processing in python.



Tagged File sink block works with a '**burst**' tag:

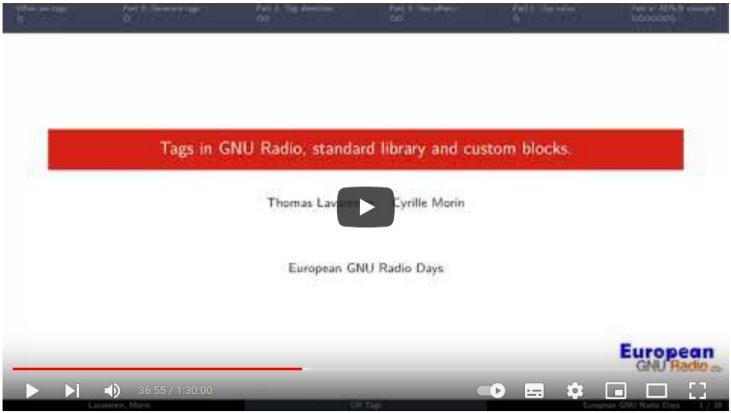
if it detects a '**burst**' tag with **PMT 'True'**, it opens a new file and saves all the data until it detects a new '**burst**' tag with **PMT 'False'**.

Adding the '**burst**' tags is the role of the **RDS Packet divider**.

RDS Packet divider is a **Python block**:

```
class blk(gr.sync_block):  # other base classes are basic_block, decim_block, interp_block
    """Embedded Python Block that put a tag with key burst and pmt:True when a tag is recieved and
key burst and pmt:False 178 samples after that"""

    def __init__(self, tag_name="packet"):  # only default arguments here
        """arguments to this function show up as parameters in GRC"""
        gr.sync_block.__init__(
            self,
            name='RDS packet divider',   # will show up in GRC
            in_sig=[np.int8],
            out_sig=[np.int8]
        )
        # if an attribute with the same name as a parameter is found,
        # a callback is registered (properties work, too).
        self.tag_name = tag_name
        self.max_block_length = 178

    def work(self, input_items, output_items):
        """example: multiply with constant"""
        output_items[0][:] = input_items[0]
        tags = self.get_tags_in_window(0, 0, len(input_items[0]))

        for i in range(len(tags)):

            print("We got a tag: ", tags[i].key, tags[i].value, tags[i].offset)
            #if tags[i].key == self.tag_name:
            self.add_item_tag(0, tags[i].offset, pmt.intern("burst"), pmt.PMT_T,
pmt.intern("Blc"))
            self.add_item_tag(0, tags[i].offset + self.max_block_length, pmt.intern("burst"),
pmt.PMT_F, pmt.intern("Blc"))
        return len(output_items[0])
```

More details on tags and how to build such a python block?
https://www.youtube.com/watch?v=j4Cn8U2K190



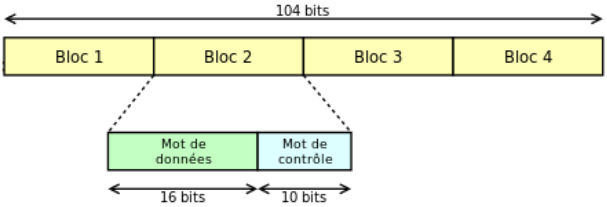
European GNU Radio Days Advanced Tutorial 4: Tags, std lib & custom blocks (T. Lavarenne & C. Morin)

European GNU Radio Days
1,01 k abonnés
Abonné
18
Partager
Extrait

4. Decoding and displaying Information

Decoding in python:

```
i=0
while i<=len(byte)-1:
    if (byte[i] != byte[i+1] ):
        trame_dec += '0'
    else:
        trame_dec += '1'
```



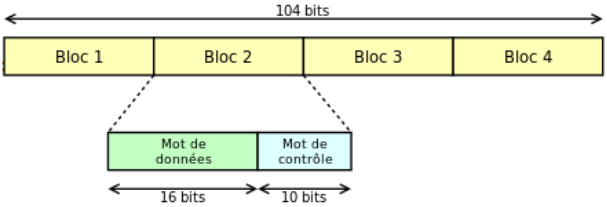
No major difficulties!

- Open the files

4. Decoding and displaying Information

Decoding in python:

```
i=0
while i<=len(byte)-1:
    if (byte[i] != byte[i+1] ):
        trame_dec += '0'
    else:
        trame_dec += '1'
```



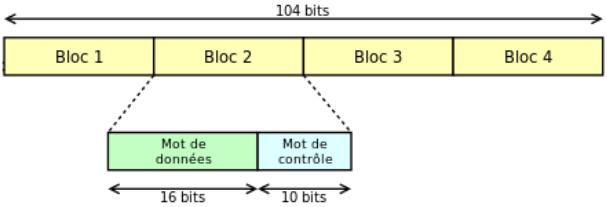
No major difficulties!

- Open the files
- Read the bytes

4. Decoding and displaying Information

Decoding in python:

```
i=0
while i<=len(byte)-1:
    if (byte[i] != byte[i+1] ):
        trame_dec += '0'
    else:
        trame_dec += '1'
```



No major difficulties!

- Open the files
- Read the bytes
- Read the doc...

What is hidden in our files?

```
Chemin_home='/home/lavarenne/GNuradiodaysParis2023'
Chemin_dossier='/home/lavarenne/trames_rds'

nom=[' ',' ',' ',' ',' ',' ',' ',' ']
texte=[]
for i in range(0,65):
    texte.append('  ')

text=''
while(True):
    time.sleep(0.2)

    fichierhome=[]
    for files in os.listdir(Chemin_home):
        fichierhome.append(files)
    listehome=''
    for j in range(0,len(fichierhome)):
        listehome+=str(fichierhome[j])

    #déplacer les fichiers trames dans le dossier de travail
    if ('.dat' in listehome)==True:
        os.system('mv {0}/file*
{1}/'.format(Chemin_home,Chemin_dossier))


        #lecture du nom des fichiers et insertion dans la liste
fichier
        fichier=[]
        for files in os.listdir(Chemin_dossier):
            fichier.append(files)

        '''préambule: access code FRance Musique F203
00110011010100101010101010101100 ''' #ou
11001100101011010101010101010011
        '''préambule: access code FRance Inter F201
00110011010100101010101010101011 '''
```

Python

```
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
BLEUALSA
FRANCE BLEU ALSACE - DIS MOI QUE L'AMOUR - M. LAVOINE    BAMBOU
BLEUALSA
```

Conclusion

- RDS and FM Radio is a little old but still in use and interesting

Conclusion

- RDS and FM Radio is a little old but still in use and interesting
- Many notions of physics, signal processing and computer science in a concrete way

Conclusion

- RDS and FM Radio is a little old but still in use and interesting
- Many notions of physics, signal processing and computer science in a concrete way
- Working with real signals is an important source of motivation

Thank you

- Thank you to Jean-Michel Friedt with whom I have been talking a lot about this ...

Thank you

- Thank you to Jean-Michel Friedt with whom I have been talking a lot about this ...
- Thank you to Cyrille Morin, for the help on the tutorial on tags and the creation of the python block.

Thank you

- Thank you to Jean-Michel Friedt with whom I have been talking a lot about this ...
- Thank you to Cyrille Morin, for the help on the tutorial on tags and the creation of the python block.
- Thank you to all the developers and contributors of GNU Radio !!

Thank you

- Thank you to Jean-Michel Friedt with whom I have been talking a lot about this ...
- Thank you to Cyrille Morin, for the help on the tutorial on tags and the creation of the python block.
- Thank you to all the developers and contributors of GNU Radio !!
- Thank you all for your attention!

Thank you

- Thank you to Jean-Michel Friedt with whom I have been talking a lot about this ...
- Thank you to Cyrille Morin, for the help on the tutorial on tags and the creation of the python block.
- Thank you to all the developers and contributors of GNU Radio !!

- See you at the coffee break!