

دانشگاه صنعتی خواجه نصیرالدین طوسی

پروژه درس اصول سیستم های مخابراتی

مدرس: دکتر محمد علی جمشیدی

تهیه کننده: مصطفی لطیفیان - ۴۰۱۲۲۱۹۳

فهرست راهنمای پروژه

۱. مدولاسیون و دمودلاسیون VSB برای سیگنال پیام سینوسی	۳
۲.۱. طراحی مدولاسیون VSB به همراه نویز در GNU Radio	۵
۳.۱. طراحی دمودلاسیون VSB در GNU Radio	۱۰
۴.۱. مشاهده سیگنال پیام قبل و بعد از دمودلاسیون VSB	۱۵
۵.۱. مشاهده SNR در VSB	۱۶
۶.۱. مشاهده چگالی طیف توان (PSD) در VSB	۱۹
۲. مدولاسیون و دمودلاسیون FM برای سیگنال پیام سینوسی	۲۲
۲.۲. طراحی مدولاسیون FM به همراه نویز در GNU Radio	۲۳
۳.۲. طراحی دمودلاسیون FM به همراه نویز در GNU Radio	۲۵
۴.۲. مشاهده سیگنال پیام قبل و بعد از دمودلاسیون FM	۲۶
۵.۲. مشاهده SNR در FM	۲۷
۶.۲. مشاهده چگالی طیف توان (PSD) در FM	۲۸
۳. طراحی امتیازی	۲۹
۱.۳. مدولاسیون امتیازی در GNU Radio	۲۹
۲.۳. مدولاسیون امتیازی در GNU Radio	۲۹
.....	۳۰
۳.۳. مشاهده خروجی امتیازی	۳۰
۴. منابع	۳۰

۱. مدولاسیون و دمودلاسیون VSB برای سیگنال پیام سینوسی

برای شروع طراحی ابتدا به سراغ نحوه تحلیل و قوانین مدولاسیون VSB می‌پردازیم.

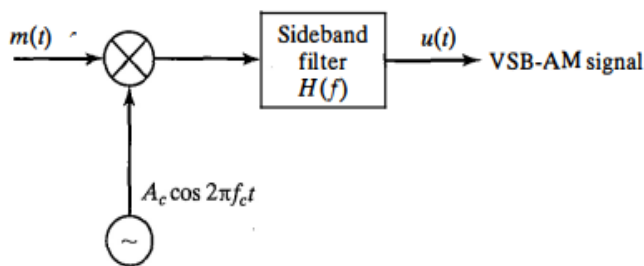
با اجازه عبور قسمتی از باند جانبی^۱ و ظاهر شدن آن در خروجی دمولاتور، شرایط لازم برای داشتن پاسخ فرکانسی دقیق در فیلتر باند جانبی یک سیستم SSB AM ساده تر می‌شود؛ لذا در ازای افزایش پهنای باند کانال لازم برای انتقال سیگنال، می‌توان طراحی فیلتر باند جانبی را ساده تر کرد. سیگنال حاصل را AM باند جانبی اثری^۲ یا VSB گویند.

برای تولید سیگنال VSB-AM، ابتدا سیگنال DSB-AM تولید شده و سپس مطابق تصویر ۱.۱ از یک فیلتر باند جانبی با پاسخ فرکانسی با پاسخ فرکانسی $H(f)$ عبور می‌کند. سیگنال VSB در حوزه زمان را می‌توان به شکل زیر بیان کرد:

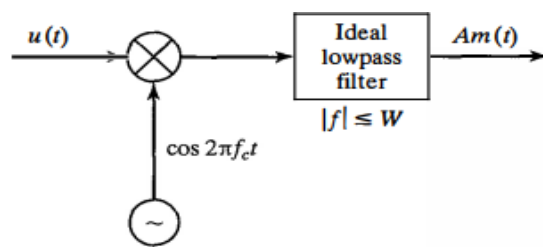
$$u(t) = [A_c m(t) \cos 2\pi f_c t] * h(t) \quad (1.1)$$

که در آن $h(t)$ پاسخ ضربه فیلتر VSB است. بیان متناظر آن در حوزه فرکانس به صورت زیر است:

$$U(f) = \frac{A_c}{2} [M(f - f_c) + M(f + f_c)] H(f) \quad (2.1)$$



تصویر ۱.۱ ساخت سیگنال VSB-AM



تصویر ۲.۱ دمودلاسیون سیگنال VSB

سیگنال $u(t)$ در مولفه حامل $\cos 2\pi f_c t$ ضرب شده و سپس مطابق تصویر ۲.۱ از فیلتر پایین گذر ایده آلی عبور می‌کند. بنابراین سیگنال حاصل ضرب برابر خواهد بود با:

$$v(t) = u(t) \cos 2\pi f_c t \quad (3.1)$$

معادل این عبارت در حوزه فرکانس برابر است با:

$$V(f) = \frac{1}{2} [U(f - f_c) + U(f + f_c)] \quad (4.1)$$

¹ Side Band

² Vestigial Sideband

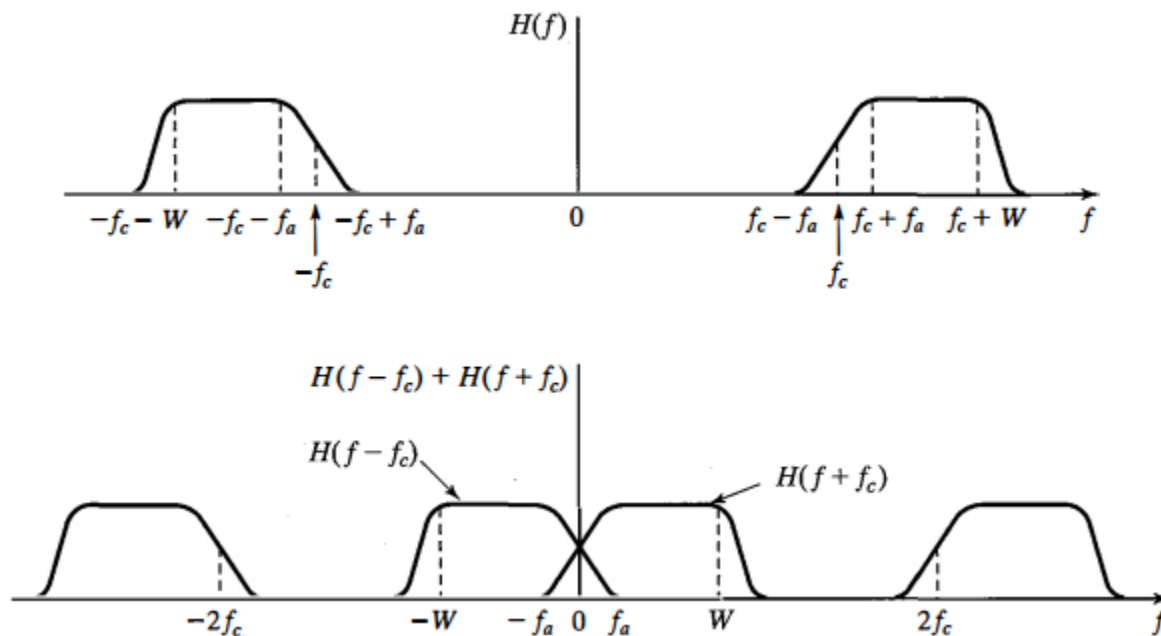
فیلتر پایین گذر جملات با فرکانس دو برابر را حذف و فقط مولفه های واقع در محدوده ی فرکانسی $|f| \leq W$ را از خود عبور می دهد. در نتیجه، طیف سیگنال خروجی فیلتر پایین گذر ایده آل به صورت زیر خواهد بود:

$$V_l(f) = \frac{A_c}{4} M(f) [H(f - f_c) + H(f + f_c)] \quad (5.1)$$

سیگنال پیام در خروجی فیلتر پایین گذر ایده آل باید بدون اعوجاج باشد؛ لذا مشخصه ی فیلتر VSB باید در شرط زیر صدق کند:

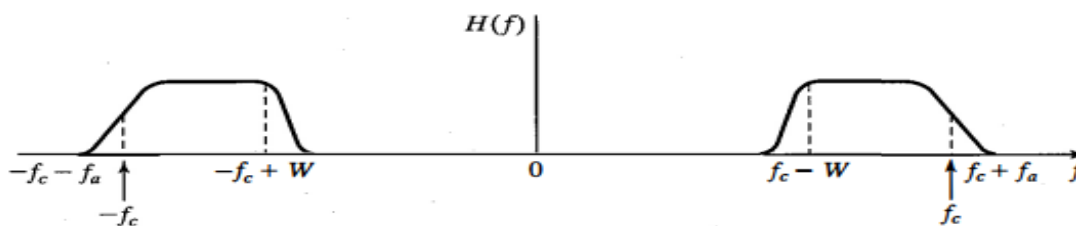
$$H(f - f_c) + H(f + f_c) = \text{constant}, |f| \leq W \quad (6.1)$$

فیلتری با مشخصه پاسخ فرکانسی تصویر ۳.۱ این شرط را تامین می کند.



تصویر ۳.۱ مشخصه فیلتر VSB

$H(f)$ باند جانبی بالایی و اثری از باند جانبی پایینی را از خود عبور می دهد که به صورت $f_c - f_a < f < f_c + f_a$ نسبت به فرکانس حامل تقارن فرد دارد و f_a فرکانس کوچک دلخواهی است و لذا می توان نسخه بدون اعوجاجی از سیگنال ارسالی را بدست آورد. تصویر ۴.۱ پاسخ فرکانسی فیلتر VSB را نشان می دهد که باند جانبی اثری از باند جانبی بالایی را از خود عبور خواهد داد.

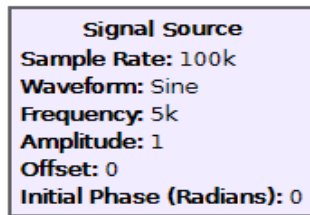


تصویر ۴.۱ پاسخ فرکانسی فیلتر VSB برای انتخاب باند جانبی پایینی از سیگنال های پیام

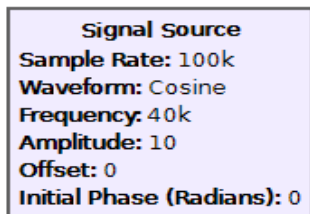
۲.۱. طراحی مدولاسیون VSB به همراه نویز در GNU Radio

در طول این طراحی برای مدولاسیون و دمودولاسیون از سیگنال پیام سینوسی استفاده شده است. نرخ samp rate نیز در طول این طراحی برابر 100K Hz فرض شده است.

در شروع طراحی از تولید کننده سیگنال^۳ برای ایجاد سیگنال پیام و سیگنال حامل^۴ استفاده می‌کنیم. در این طراحی فرکانس سیگنال پیام برابر 5K Hz و فرکانس حامل برابر 40K Hz در نظر گرفته شده‌اند، همچنین دامنه سیگنال پیام برابر ۱ و سیگنال حامل برابر ۱۰۰ در نظر گرفته شده‌اند. شکل ۱.۲.۱ بیانگر این طرح‌ها می‌باشد.



طبق فرمول ۱.۱ این دو سیگنال باید در هم ضرب شده و سپس از فیلتر میان‌گذر عبور کنند.

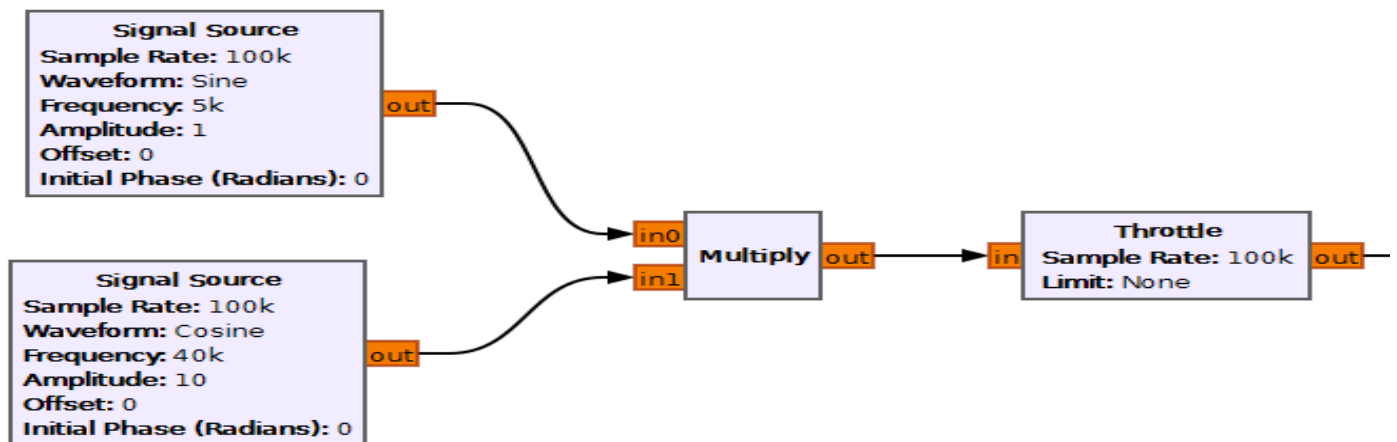


عمل ضرب این دو سیگنال توسط بلوک Multiply انجام می‌شود و نکته‌ای که باید در نظر گرفت این است که باید در طول انجام تمام این اعمال باید نوع فیلترها و بلوک‌ها به صورت عددی مشخص باشد به عنوان مثال اگر سیگنال به صورت complex تنظیم شده است، بلوک‌های دیگر نیز باید به صورت complex تنظیم شوند، در این طراحی تمام بلوک‌ها به صورت float در نظر گرفته شده‌اند.

شکل ۱.۲.۱ منابع تولید سیگنال

شکل ۲.۲.۱ نشان دهنده ضرب دو سیگنال پیام و حامل است.

استفاده از بلوک throttle تأثیری در سیگنال‌ها نمی‌گذارد و صرفاً برای کنترل کردن میانگین نرخ ارز است تا از مقدار خاصی تجاوز نکند که در اینجا این مقدار برابر samp rate فرض شده، این مسئله در سایت wiki.gnuradio.org بیان شده است و متن این گفتار در تصویر ۱.۲.۱ نیز ذکر شده.



شکل ۲.۲.۱ ضرب سیگنال پیام در سیگنال حامل

³ Signal source

⁴ Carrier Signal

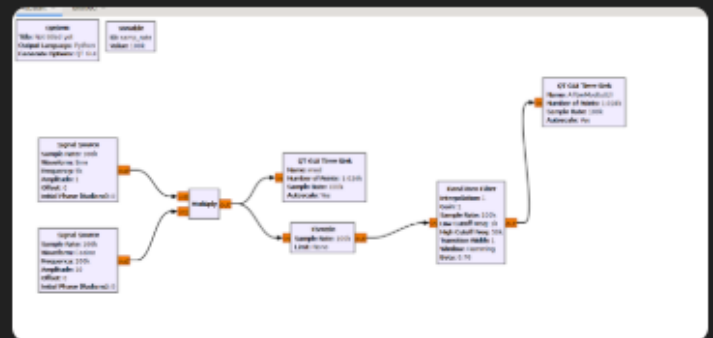
Throttle

Throttle flow of samples such that the average rate does not exceed the specific rate (in samples per second).

The throttle block copies the items from its input to its output, or just consumes the input samples at a specified average rate. It does not in any way change the digital signal.

تصویر ۱.۲.۱ کارکرد throttle، [لینک مربوطه](#)

در ادامه طبق فرمول ۱.۱ باید سیگنال ضرب شده را از یک فیلتر میان گذر عبور دهیم، طبق تصویر ۳.۱ فرکانس این فیلتر باید بین فرکانس حامل و پیام باشد، باتوجه به اینکه سیگنال حاصل ضرب ۴ فرکانس دارد که عبارتند از (45,-35,35,45)- که برای مدوله و دمودله کردن تنها نیاز به ۲ فرکانس یا سیگنال داریم که ما در این طراحی فرکانس های پایین تر یعنی فرکانس های 35, 35- را انتخاب می کنیم و طبق شکل ۳.۲.۱ این طراحی را انجام می دهیم، برای انتخاب فرکانس های high , low cut off , cut off از چت جی پی تی نیز کمک گرفته شده است که در تصویر ۲.۲.۱ قابل مشاهده است.



i want have band pass filter to make VSB signal, how should i custom my low and high cut of frequency in gnu radio (my sin frequency is 5k and my cos frequency is 100k and my sample rate is 100k)



To create a **Vestigial Sideband (VSB)** signal using a **Band-Pass Filter (BPF)** in **GNU Radio**, you need to carefully adjust your **low cutoff** and **high cutoff frequencies** to filter out the undesired portion of the signal.

Understanding Your Signal

- **Modulating Signal:** A 5 kHz sine wave
- **Carrier Signal:** A 100 kHz cosine wave
- **Sampling Rate:** 100 kHz
- **Modulated Signal Spectrum:** The multiplication of a 5 kHz sine wave with a 100 kHz cosine will generate **Upper Sideband (USB)** and **Lower Sideband (LSB)** around the carrier at:
 - **USB:** $100kHz + 5kHz = 105kHz$
 - **LSB:** $100kHz - 5kHz = 95kHz$

How to Set Band-Pass Filter for VSB

For Vestigial Sideband (VSB), you want to partially suppress one sideband while keeping most of the other.

Choice of Cutoff Frequencies

- Low Cutoff Frequency: 95 kHz (to allow LSB but suppress lower unwanted frequencies)
- High Cutoff Frequency: 105 kHz (to allow USB but suppress higher unwanted frequencies)

This filter will help retain most of one sideband while reducing the strength of the other, which is the principle of VSB.

Steps in GNU Radio

1. Modify your Band Pass Filter settings:

- Low Cutoff Frequency: 95k
- High Cutoff Frequency: 105k
- Transition Width: Keep it small (e.g., 1k)
- Window: Hamming (or other smooth windowing options)

2. Check the Filtered Signal in Time & Frequency Domain

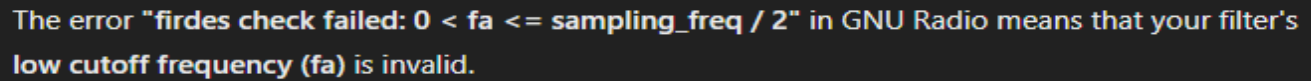
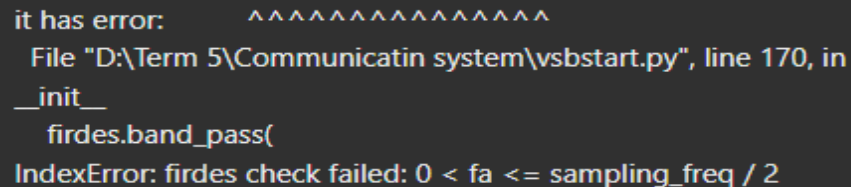
- Use a QT GUI Frequency Sink to verify that you see a vestigial sideband structure.
- A QT GUI Time Sink will help visualize how the signal behaves after filtering.

تصویر ۲.۲.۱ پیشنهاد CHAT GPT برای طرح فیلتر میان گذر

طبق تصویر بالا پیشنهاد CHAT GPT برای طراحی به شرح بالا است، که ابتدا برای سیگنال پیام 5K Hz و حامل 100K Hz مطرح شده است اما با استفاده از این طراحی با یک خطای منطقی در برنامه GNU Radio مواجه می‌شویم که این خطا به شرح زیر است:

```
^^^^^^^^^^^^^^^^^^^^ File "D:\Term 5\Communicatin system\vsbstart.py", line 170, in __init__
firdes.band_pass( IndexError: firdes check failed: 0 < fa <= sampling_freq / 2
```

این خطا بیانگر آن است که فرکانس f_a که در تصویر ۴.۱ به آن اشاره شده است و باید از نصف فرکانس samp rate کوچک‌تر باشد، که این مقدار برابر 50K Hz است، پس از ارائه این مشکل به CHAT GPT پیشنهاد آن به شرح تصویر ۳.۲.۱ می‌باشد.



The condition $0 < fa \leq \frac{\text{sampling_freq}}{2}$ must be satisfied for filter design in GNU Radio. Given your parameters:

- The issue is that the high cutoff frequency (105 kHz) exceeds the Nyquist frequency, which is $\frac{100k}{2} = 50kHz$.

You must design the filter within the valid Nyquist range. Since your modulated signal components (95 kHz and 105 kHz) are outside the Nyquist frequency (50 kHz), you need to shift everything down into the valid range before filtering.

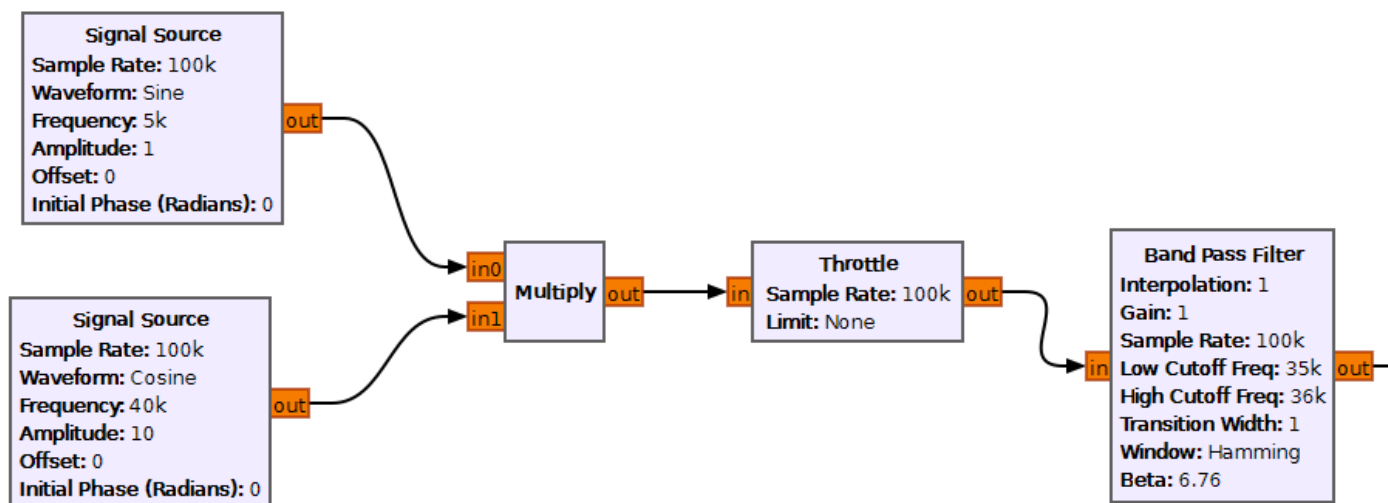
1. Use a Lower Carrier Frequency

- Then, your USB and LSB will be $(40\text{k} + 5\text{k} = 45\text{kHz})$ and $(40\text{k} - 5\text{k} = 35\text{kHz})$.
- You can apply a BPF with cutoff frequencies around $35\text{kHz} - 45\text{kHz}$, which is within the Nyquist range.

- This ensures that 95 kHz and 105 kHz do not alias.
- Your BPF can then correctly filter 95 kHz - 105 kHz.

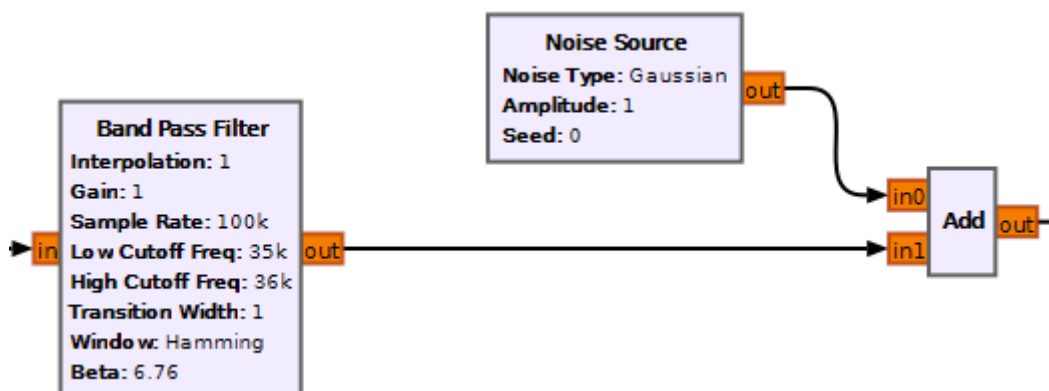
- Change the cosine carrier frequency from 100 kHz to 40 kHz.
- Set the BPF cutoff frequencies to 35 kHz - 45 kHz.
- This will give you a valid VSB signal without aliasing errors.

با توجه به پاسخ CHAT GPT فیلتر مورد نیاز را طرح می‌کنیم، با توجه به اینکه در طراحی مطرح شده توسط CHAT GPT حاشیه امنیت فرکانس‌های بالایی در نظر گرفته نشده است، فرکانس بالا را به جای 45K Hz برابر 38K Hz در نظر می‌گیریم تا از خطای احتمالی جلوگیری کنیم.



شکل ۳.۲.۱ طراحی فیلتر میان گذر مناسب در GNU Radio

برای اضافه کردن نویز به سیگنال مدوله شده از یک بلوک جمع کننده استفاده می‌کنیم که در شکل ۵.۲.۱ قابل مشاهده است.



شکل ۴.۲.۱ اضافه کردن نویز به سیگنال مدوله شده

پس از اتمام طراحی فیلتر حال سیگنال $u(t)$ ساخته شده است، پس از آن باید به سراغ اضافه کردن نویز به سیستم برویم این کار را می‌توان با استفاده از یک تولید کننده نویز^۵ انجام داد، در این طراحی با توجه به خواسته پروژه از یک نویز گوسی استفاده می‌کنیم که تولید آن با استفاده از تولید کننده نویز امکان پذیر است، برای تغییر مقدار نویز از بلوکی به اسم QT GUI Range استفاده می‌کنیم، با استفاده از این بلوک می‌توان مقدار نویز موثر در دمودلاسیون را به میزان دلخواه تغییر داد تا تاثیر آن را در خروجی مشاهده کرد؛ طراحی این بلوک در شکل ۵.۲.۱ قابل مشاهده است.

^۵ Noise source

QT GUI Range
 ID: noise_amp
 Default Value: 1
 Start: 0
 Stop: 10
 Step: 1

Properties: QT GUI Range

GeneralAdvancedDocumentation

IDnoise_amp

Label

[string]

Typefloat

▼

Default Value1

[float]

Start0

[float]

Stop10

[float]

Step1

[float]

WidgetCounter + Slider

▼

Minimum Length200

[int]

OKCancelApply

Properties: Noise Source

GeneralAdvancedDocumentation

Output Typefloat

▼

Noise TypeGaussian

▼

Amplitudenoise_amp

[real]

Seed0

[int]

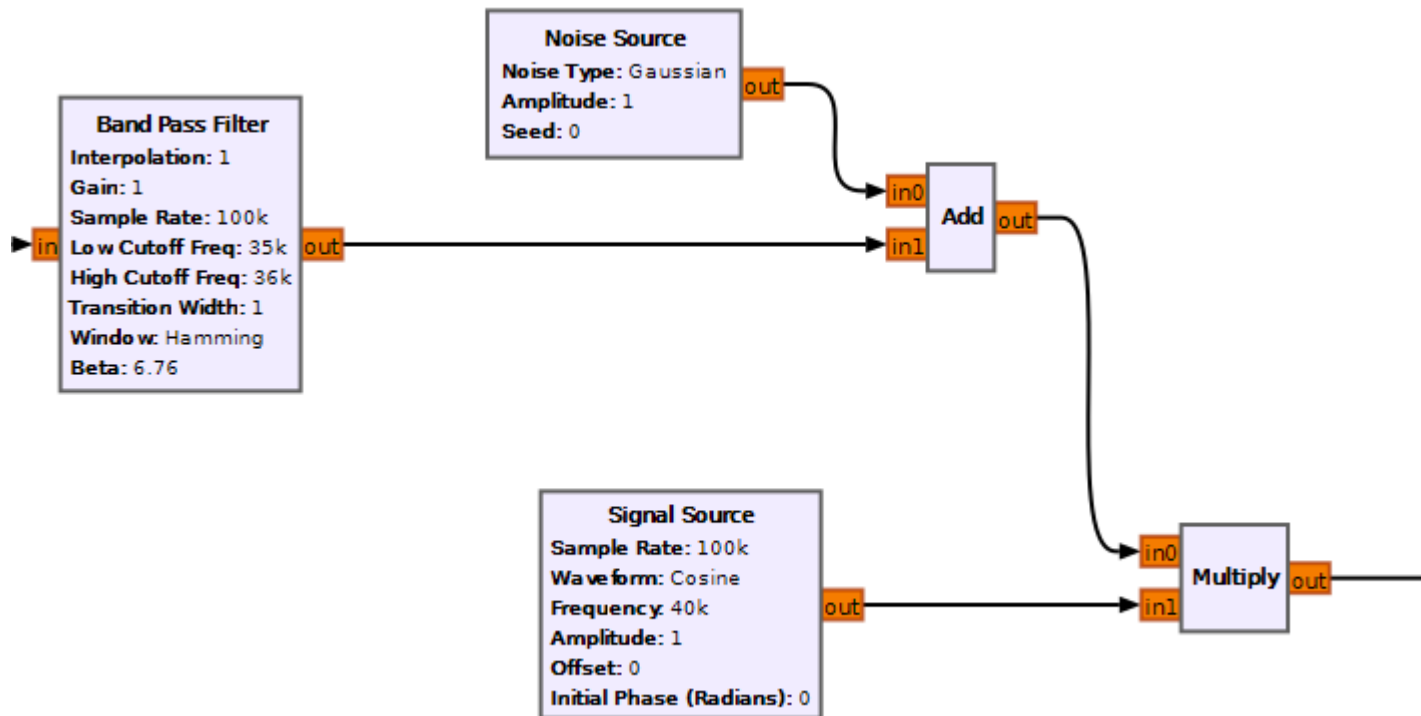
OKCancelApply

شکل ۵.۲.۱ نویز طراحی شده برای سیگنال مدوله شده

۳.۱. طراحی دمودلاسیون VSB در GNU Radio

پس از مدوله کردن سیگنال و اضافه کردن نویز به آن حال به سراغ دمودلاسیون سیگنال می‌رویم.

برای طراحی طبق فرمول ۳.۱ می‌دانیم باید سیگنال مدوله شده را در سیگنال حامل با دامنه واحد ضرب کنیم که این عمل در شکل ۱.۳.۱ قابل مشاهده است، همچنین طبق تصویر ۲.۱ پس از انجام عمل ضرب به یک فیلتر پایین گذر نیز نیازمندیم که در ادامه به طراحی آن می‌پردازیم.



شکل ۱.۳.۱ ضرب سیگنال مدوله شده به همراه نویز در سیگنال حامل

Correct Low-Pass Filter Settings

Your LPF should:

- Pass frequencies below ~5.5 kHz (to recover the original message)
- Cut off frequencies above 5.5 kHz (to remove high-frequency components)
- Use a transition width of ~1 kHz

LPF Configuration:

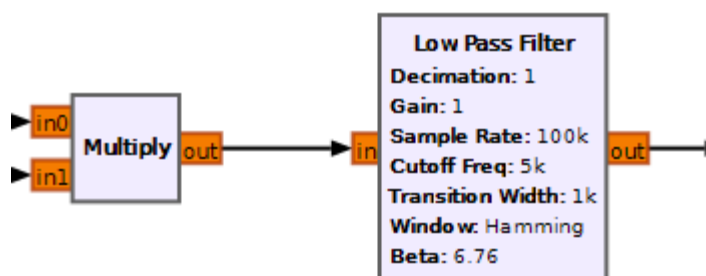
- Cutoff Frequency: 5.5 kHz ✓ (Covers the original 5 kHz signal)
- Transition Width: 1 kHz ✓ (Ensures smooth filtering)
- Window Function: Hamming (Good balance of frequency response & attenuation)
- Gain: 1

Final Steps

- Verify Output with a Time Sink: Ensure the demodulated signal looks like the original 5 kHz sine wave.
- Check the Frequency Sink: Confirm that unwanted high-frequency components are removed.

تصویر ۱.۳.۱ طرح فیلتر LPF با استفاده از CHAT GPT


همانطور که قابل مشاهده است فرکانس مطرح شده توسط CHAT GPT با استاندارد به مقدار اولیه داده شده برابر با 5.5K Hz است اما با استفاده از شرط $|f| \leq W$ این مقدار را برابر 5K Hz در نظر می گیریم و طراحی را انجام می دهیم که در شکل ۲.۳.۱ این طراحی قابل مشاهده است.



شکل ۲.۳.۱ طراحی فیلتر پایین گذر

پس از طراحی فیلتر پایین گذر مشاهده می‌شود که سیگنال دموذله شده به همراه نویز کارکرد خوبی ندارد و مطلوب نیست، برای رفع این مشکل به سراغ کمک گرفتن از CHAT GPT می‌رویم و این فرایند در تصویر ۲.۳.۱ قابل مشاهده است.

now i added a noise to my graph, how can i reduce noise effect on my demodulation?

 Adding noise is a great way to test your system's robustness! To reduce its effect on demodulation, you can apply the following techniques:

1. Adjust Low-Pass Filter Parameters

Your current LPF has a cutoff at 5 kHz, which is good, but you can:

- Decrease the transition width to 500 Hz (instead of 1 kHz) for sharper filtering.
- Use a higher beta value (e.g., 8-10) for the Hamming window to improve noise suppression.

2. Use a Moving Average or Integrate-Comb Filter

A moving average filter (or integrate-comb filter) can smooth out high-frequency noise:

- Add a "Moving Average" block before the final QT GUI Time Sink.
- Set the length of the moving average to 5-10 samples.

3. Implement a Wiener Filter (Advanced)

A Wiener Filter can adaptively remove noise based on signal properties.

- This requires Python scripting inside GNU Radio, using `scipy.signal.wiener`.

4. Increase Signal Amplitude Before Filtering

- Multiply your modulated signal by a factor (e.g., 2-5) before the Band-Pass Filter.
- This increases the signal-to-noise ratio (SNR).

5. Apply AGC (Automatic Gain Control)

- Add an AGC (Automatic Gain Control) block after the band-pass filter to maintain consistent signal strength.

در اینجا ۵ مورد برای بهبود نویز مطرح شده است که با استفاده از سعی و خطا می‌توان دریافت که مورد ۱ سیگنال دمودله شده را از نظر سینوسی بودن دچار اختلال می‌کند؛ مورد ۲ سیگنال را از نظر دامنه به شدت از خواسته ما دور می‌سازد؛ مورد ۳ با استفاده از روش های پیشرفته پیاده سازی می‌شود که فرایند سخت و مطلوبی برای این درس نمی‌باشد؛ پس به سراغ مورد ۴ و ۵ می‌رویم. در مورد ۵ با استفاده از CHAT GPT به تصویر ۳.۳.۱ می‌رسیم.



To implement Automatic Gain Control (AGC) in your GNU Radio flowgraph, follow these steps:

1. Add the "AGC2" Block

- Open GNU Radio Companion (GRC).
- Search for "AGC2" in the block list.
- Place it **before the Low Pass Filter** in your demodulation chain.

2. Configure AGC2 Block Parameters

Parameter	Recommended Value	Explanation
Rate	1e-3 (0.001)	Adjusts how fast the AGC adapts to changes. Lower values mean slower adjustments, reducing noise amplification.
Reference Level	1.0	The target amplitude level of the signal.
Gain (Max)	10.0	Maximum allowed gain to prevent excessive amplification.
Gain (Min)	0.1	Prevents gain from going too low and losing weak signals.
Attack Rate	1e-2 (0.01)	How quickly AGC increases gain for weak signals.
Decay Rate	1e-3 (0.001)	How quickly AGC reduces gain for strong signals.

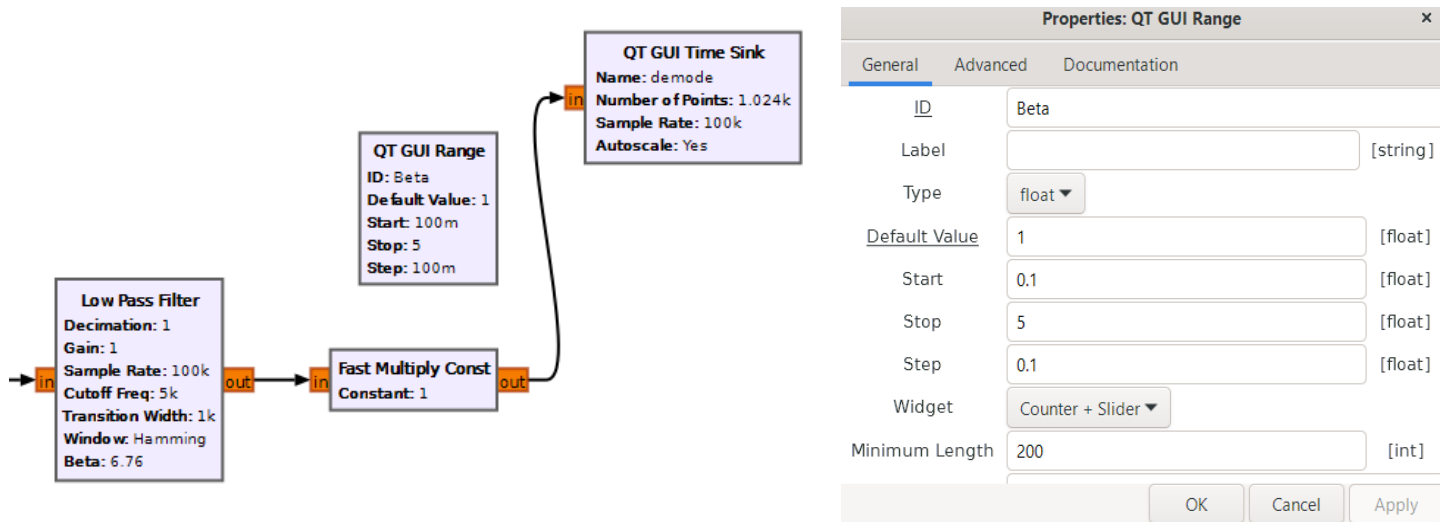
3. Connect AGC2 in Your Flowgraph

- Place it **after the Band-Pass Filter and before the Low-Pass Filter**.
- This ensures that AGC operates on a cleaned-up version of the signal.

تصویر ۳.۳.۱ استفاده از AGC برای کاهش نویز

با پیاده سازی این طرح به این نتیجه می‌رسیم که بازهم مشکلات دامنه به قوت خود باقی می‌مانند.

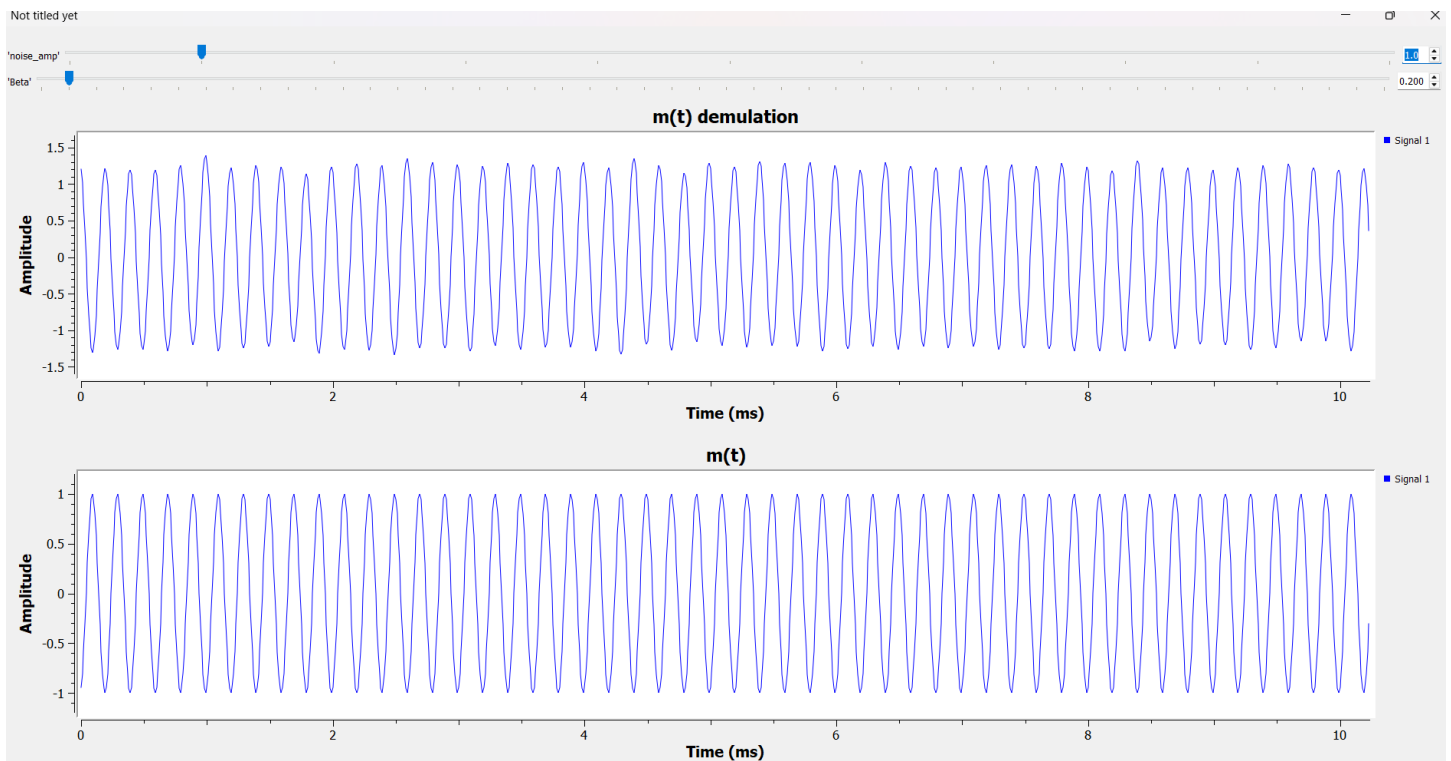
به سراغ مورد ۴ می‌رویم، در این مورد مطرح شده که دامنه قبل از فیلتر دچار افزایش شود اما با استفاده از روش بهتری مانند فرمول ۵.۱ می‌توان با استفاده از یک ضرب کننده در عدد ثابت دامنه را پس از خروجی فیلتر پایین گذر افزایش داد که برای این منظور از بلوک Multiply const استفاده می‌کنیم همچنین با استفاده از یک QT GUI Range این مقدار ضرب شونده را نیز به صورت اختیاری تغییر می‌دهیم؛ این طراحی در شکل ۳.۳.۱ قابل مشاهده می‌باشد.



شکل ۳.۳.۱ طراحی کاهش دهنده اثر نویز

۴.۱. مشاهده سیگنال پیام قبل و بعد از دمودلاسیون VSB

شکل ۱.۴.۱ خروجی نهایی طراحی می باشد.



شکل ۱.۴.۱ خروجی نهایی

همانطور که قابل مشاهده است با استفاده از بتا سیگنال مدوله شده و سیگنال پیام بسیار شبیه به یکدیگر می باشند و می توان دریافت که طراحی انجام شده مناسب است.

۵.۱. مشاهده SNR در VSB

همانطور که می‌دانیم فرمول SNR به صورت نسبت توان سیگنال به توان نویز است که به صورت زیر مطرح می‌شود:

$$SNR = \frac{P_s}{P_n} \quad (7.1)$$

همچنین برای بیان SNR به صورت دسی بل می‌توان از فرمول زیر استفاده کرد:

$$\left(\frac{S}{N}\right)_{dB} = 10 \log \left(\frac{P_s}{P_n}\right) \quad (8.1)$$

برای گرفتن خروجی توان و محاسبه SNR بلوک مشخص و واحدی وجود ندارد بنابراین از CHAT GPT کمک می‌گیریم و مسئله خواسته شده را مطرح می‌کنیم که نتیجه آن در تصویر ۱.۵.۱ آورده شده است.

it is my final graph, tell me how can i calculate SNR?



How to Calculate Signal-to-Noise Ratio (SNR) in GNU Radio

Step 1: Understanding SNR

SNR (Signal-to-Noise Ratio) is defined as:

$$SNR = \frac{P_{signal}}{P_{noise}}$$

or in decibels (dB):

$$SNR_{dB} = 10 \log_{10} \left(\frac{P_{signal}}{P_{noise}} \right)$$

where:

- P_{signal} is the average power of the signal,
- P_{noise} is the average power of the noise.

Step 2: Add Blocks to Measure Signal and Noise Power

To measure SNR, you need to calculate the power of both the **signal** and the **noise**.

1. Use "Probe RMS" Block

- This block calculates the **Root Mean Square (RMS)** of a signal.
- Place one "Probe RMS" **before noise addition** (to measure signal power).
- Place another "Probe RMS" **after noise addition** but before demodulation (to measure total power).

2. Calculate Noise Power

- Compute **Noise Power** as:

$$P_{noise} = P_{total} - P_{signal}$$

- If the noise is **Gaussian with variance σ^2** , then:

$$P_{noise} \approx \sigma^2$$

- If noise amplitude is **1**, then:

$$P_{noise} \approx 1$$

Alternative: Theoretical SNR Calculation

If you know the signal and noise power values beforehand:

$$SNR_{dB} = 10 \log_{10} \left(\frac{A_{signal}^2}{A_{noise}^2} \right)$$

where A is the amplitude of the signal or noise.

تصویر ۱.۵.۱ نحوه محاسبه SNR توسط CHAT GPT

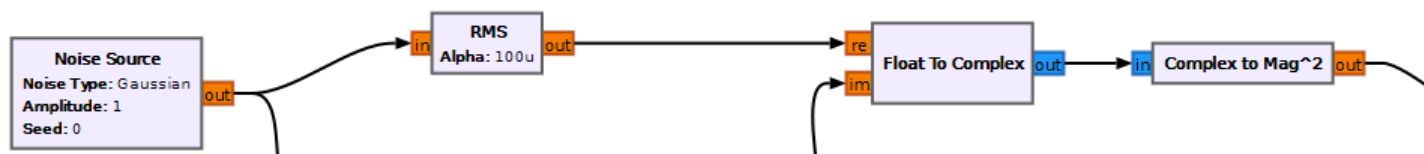
با توجه به مطالب مطرح شده باید از بلوک RMS استفاده کنیم، پس از جستجو متوجه می‌شویم که طبق مطلب بیان شده در سایت wiki.gnuradio.org طبق تصویر ۱.۵.۱ این بلوک ریشه میانگین مربع را به عنوان خروجی به ما می‌دهد.

RMS

Calculates RMS average power.

تصویر ۱.۵.۱ نحوه عملکرد [RMS](#)

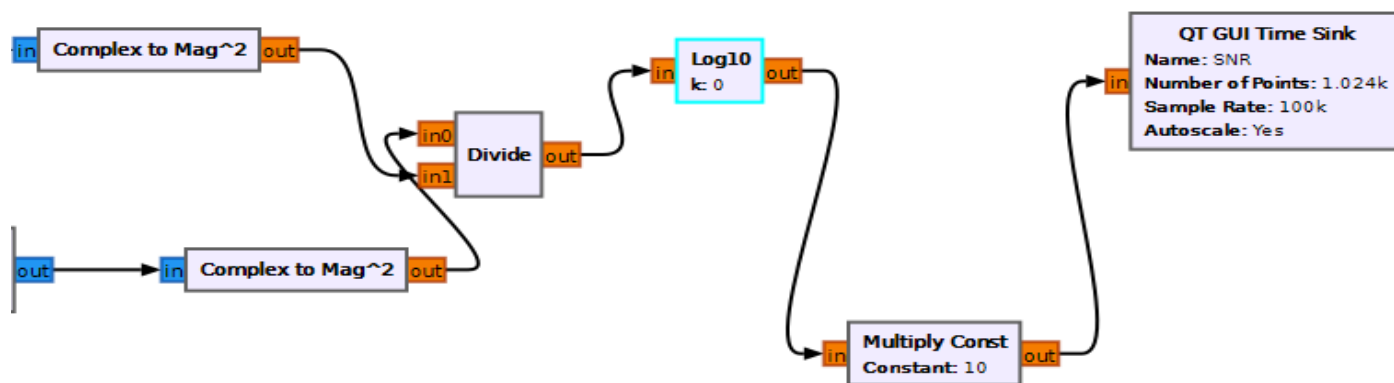
برای محاسبه توان با توجه به اینکه یک رادیکال از حاصل پس از RMS گرفته می‌شود از یک توان ۲ ساز اندازه استفاده می‌کنیم؛ با توجه به اینکه یک سیگنال float داریم و بلوک توان ۲ برای این سیگنال وجود ندارد ابتدا سیگنال را به یک سیگنال complex تبدیل می‌کنیم و سپس اندازه آن را به توان ۲ می‌رسانیم که این مورد در شکل ۱.۵.۱ آورده شده است.



شکل ۱.۵.۱ محاسبه توان سیگنال نویز

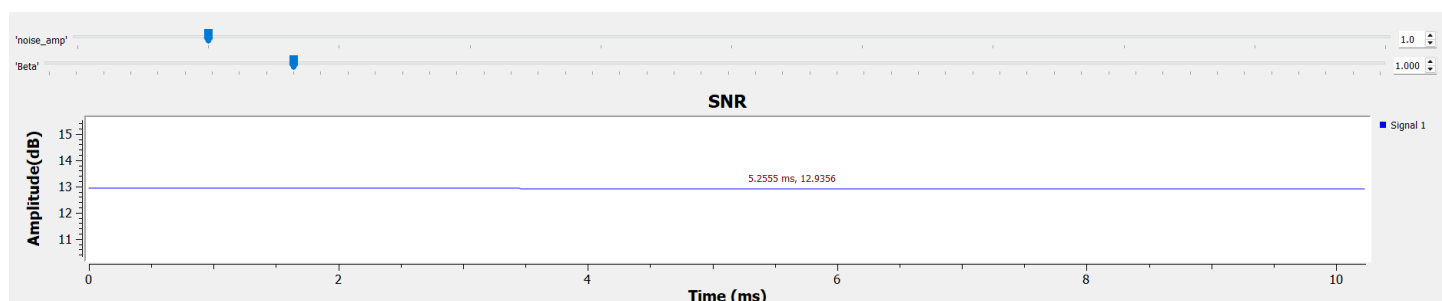
برای سیگنال دمودله شده نیز از همین روش استفاده می‌شود.

طبق فرمول ۸.۱ پس از بدست آوردن حاصل توان سیگنال ها باید از حاصل تقسیم آن ها لگاریتم در مبنای ۱۰ گرفته شود و سپس حاصل را در ۱۰ ضرب کنیم تا خروجی به صورت dB مطرح شود، این فرایند در شکل ۲.۵.۱ نشان داده شده است.



شکل ۲.۵.۱ بدست آوردن SNR در GNU Radio

حاصل خروجی و جواب نهایی سیگنال SNR طبق شکل ۳.۵.۱ خواهد بود.



شکل ۳.۵.۱ خروجی SNR

طبق شکل ۳.۵.۱ حاصل خروجی به ازای اندازه نویز ۱ برابر 13dB خواهد بود که عدد مناسبی است و این عدد با افزایش نویز کاهش خواهد یافت.

۶.۱. مشاهده چگالی طیف توان (PSD) در VSB

برای یافتن چگالی طیف توان از فرمول زیر بهره می‌گیریم:

$$S_x(f) = |X(f)|^2 \quad (9.1)$$

برای داشتن تبدیل فوریه سیگنال خروجی طبق تصویر ۱.۶.۱ از CHAT GPT کمک می‌گیریم.

this is my graph after calculate SNR, now how can i calculate PSD?



How to Calculate Power Spectral Density (PSD) in GNU Radio

The **Power Spectral Density (PSD)** shows how the signal's power is distributed across different frequencies. It is commonly measured in dB/Hz.

Step 1: Use the "FFT" Block

To compute the PSD, you need to transform the time-domain signal into the frequency domain.

- Add an "FFT" Block:
 - Set **FFT Size** (e.g., 1024 or 2048 for higher resolution).
 - Choose **Window Function** (Hamming, Blackman, etc.).
 - Enable **Forward FFT** (since we are analyzing the spectrum).

Step 2: Convert FFT Output to Power

After the FFT, you need to compute the magnitude squared of the frequency components.

1. Add "Complex to Mag^2" Block

- Converts FFT output into power values.

2. Apply "Log10" Block

- To express the power in **decibels (dB)**, use:

$$P_{dB} = 10 \log_{10}(P)$$

3. Scale by FFT Size

- Normalize by dividing by FFT size to get power per Hz.

Step 3: Display PSD Using QT GUI Sink

- Add "QT GUI Frequency Sink" to visualize PSD.
 - Set:
 - FFT Size (same as FFT block).
 - Ref Level to adjust the dB scale.
 - Y-Axis Label as Power (dB/Hz).
-

Alternative: Use the "Periodogram" Method

Instead of FFT, you can also use the **Welch Method** for PSD estimation:

1. Use the "Stream to Vector" Block:
 - Converts continuous samples into FFT-sized chunks.
 2. Apply "FFT" and "Complex to Mag^2" blocks.
 3. Use a "Moving Average" to smooth the spectrum.
-

Final Pipeline for PSD Calculation

Your final graph should look like this:

1. Input Signal → FFT
2. FFT Output → "Complex to Mag^2"
3. Magnitude Squared Output → "Log10"
4. Log Output → "QT GUI Frequency Sink"

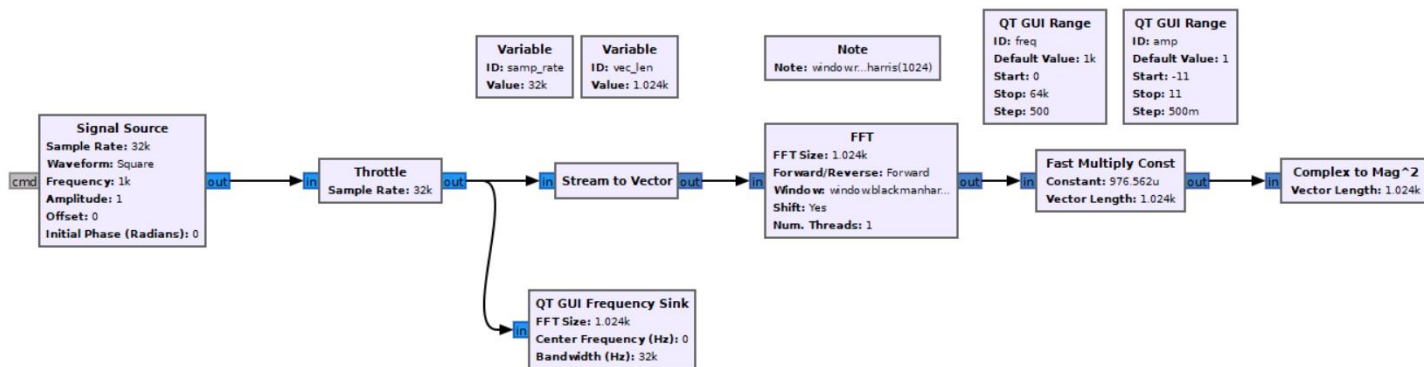
تصویر ۱.۶.۱ نحوه گرفتن تبدیل فوریه از سیگنال در GNU Radio

طبق گفته CHAT GPT برای گرفتن تبدیل فوریه از سیگنال خروجی عمل خواهیم کرد اما مشاهده می شود که با استفاده از داده های CHAT GPT مشکلاتی در حالت های float و complex پیش می آید و برنامه اجازه خروجی گرفتن را نمی دهد.

برای رفع این مشکل به سایت برنامه مراجعه می‌کنیم و طبق تصویر ۲.۶.۱ می‌توانیم مشکلات پیش آمده را برطرف سازیم.

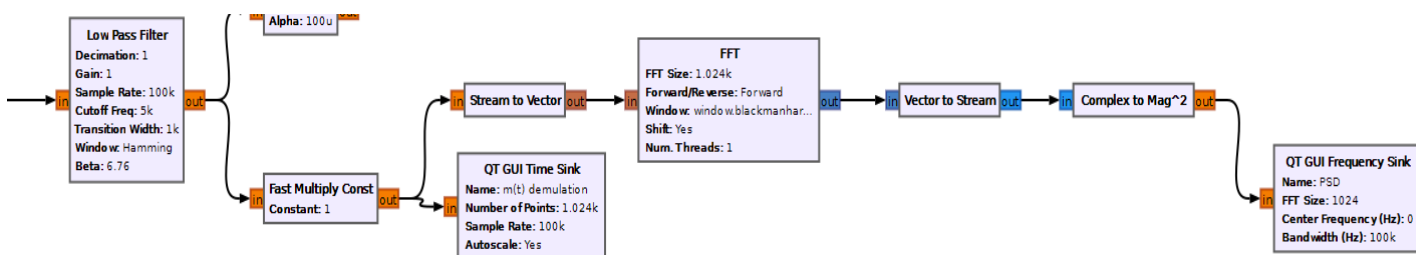
Example Flowgraph

This flowgraph shows how the FFT block can be used to reproduce the behavior of the **QT GUI Frequency Sink** block. Both outputs match, but using the FFT block directly requires converting from a stream to vector, and performing the magnitude and log manually.

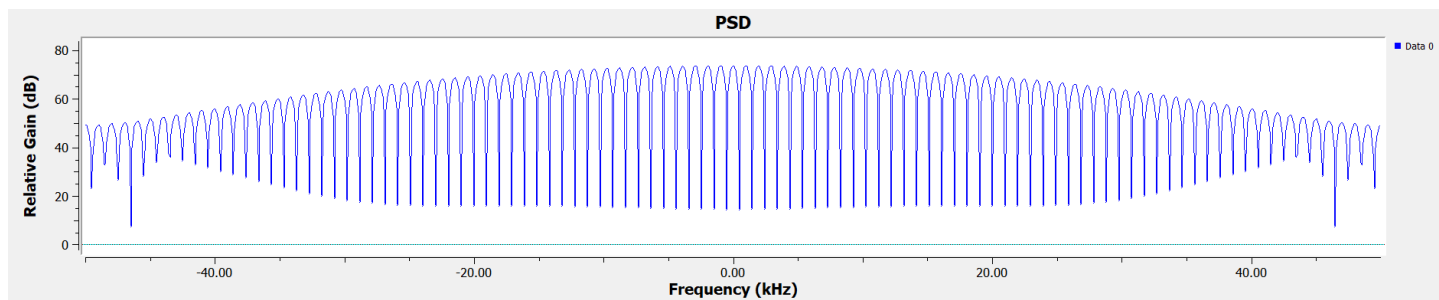


تصویر ۲.۶.۱ بهینه سازی و رفع مشکل FFT به کمک [WikiGNU](http://wiki.gnuradio.org)

طبق تصویر قبل از اتصال خروجی به FFT باید از بلوک Stream to Vector استفاده نمود؛ به دلیل اینکه استفاده از FFT در این طراحی محاسبه چگالی طیف توان است بعد از بلوک FFT از یک بلوک Vector to Stream استفاده می‌کنیم و سیگنال را در فضای فرکانسی نمایش می‌دهیم. در شکل ۱.۶.۱ این طراحی قابل مشاهده است و تصویر ۳.۶.۱ خروجی نهایی برای چگالی طیف توان می‌باشد.



شکل ۱.۶.۱ طراحی برای محاسبه چگالی طیف توان



تصویر ۳.۶.۱ خروجی نهایی چگالی طیف توان

۲. مدولاسیون و دمودلاسیون FM برای سیگنال پیام سینوسی

سیگنال مدوله شده زاویه را در حالت کلی می‌توان به صورت زیر نمایش داد:

$$u(t) = A_c \cos(2\pi f_c t + \varphi) \quad (1.2)$$

. در سیستم FM انحراف فرکانس لحظه‌ای از فرکانس حامل، متناسب با سیگنال پیام است؛ یعنی:

$$f_i(t) - f_c = k_f m(t) = \frac{1}{2\pi} \frac{d}{dt} \varphi(t) \quad (2.2)$$

در اینجا k_f ثابت انحراف است و داریم:

$$\varphi(t) = 2\pi k_f \int_{-\infty}^t m(\tau) d\tau \quad (3.2)$$

اگر سیگنال حامل با انتگرال پیام، مدولاسیون فاز شود معادل آن است که سیگنال حامل توسط سیگنال پیام، مدولاسیون فرکانس شده باشد. دمودلاسیون FM مستلزم به دست آوردن فرکانس لحظه‌ای سیگنال مدوله شده و سپس تفریق فرکانس حامل از آن است. حداکثر انحراف فرکانس در سیستم FM به صورت زیر است:

$$\Delta f_{max} = k_f \max[|m(t)|] \quad (4.2)$$

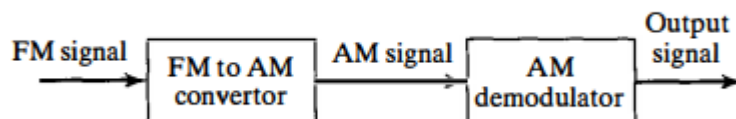
همچنین با استفاده از شاخص مدولاسیون FM خواهیم داشت:

$$\beta_f = \frac{k_f a}{f_m} = \frac{\Delta f_{max}}{W} \quad (5.2)$$

$$u(t) = A_c \cos(2\pi f_c t + \beta_f \sin(2\pi f_m t)) \quad (6.2)$$

همانطور که مشخص است فرمول ۶.۲ بیانگر مدولاسیون FM است.

در تصویر ۲.۱ نیز می‌توان نحوه دمودلاسیون FM را مشاهده کرد.



تصویر ۲.۱ دمودلاسیون سیگنال FM

هرچند پیاده سازی دمودلاسیون FM انواع متفاوتی دارد که تصویر ۲.۱ بیانگر یکی از اشکال پیاده سازی کلی آن می‌باشد.

۲.۲. طراحی مدولاسیون FM به همراه نویز در GNU Radio

در این طراحی از تجارب و مطالب بدست آمده در طراحی VSB استفاده های بسیاری می شود؛ بنابراین در این گزارش مجدداً به مطالب تکراری پرداخته نخواهد شد و فقط گزارش مطالب جدید ارائه می شود.

در مدولاسیون فرکانس داریم:

$$u(t) = A_c \cos(2\pi f_c t + 2\pi k_f \int_{-\infty}^t m(t) dt) \quad (7.2)$$

برای راحت تر شدن تحلیل و حل در این پروژه سیگنال پیام را به صورت کسینوسی در نظر می گیریم تا انتگرال آن حاصل سینوسی را به ما بدهد، با انجام عمل انتگرال در نهایت خواهیم داشت:

$$u(t) = A_c \cos(2\pi f_c t + \beta \sin(2\pi f_m t)) \quad (8.2)$$

این عبارت را مبنای شبیه سازی قرار می دهیم.

برای پیاده سازی مراحل زیر را انجام می دهیم:

$$1. \quad u(t) = A_c \cos(2\pi f_c t + \beta \sin(2\pi f_m t))$$

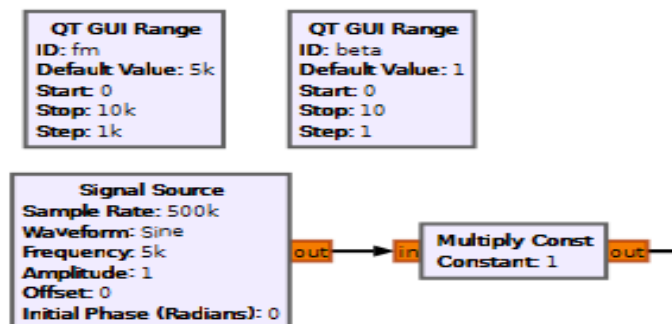
$$2. \quad \text{Re}\{A_c e^{j\beta \sin(2\pi f_m t)} e^{j2\pi f_c t}\}$$

$$3. \quad \text{Re}\{u^*(t) e^{j2\pi f_c t}\}$$

$$4. \quad \text{Re}\{u^+(t)\}$$

نحوه انجام مرحله اول در بالا بیان شده است؛ حال به سراغ اجرای مرحله ۲ می رویم.

ابتدا عبارت $e^{j\beta \sin(2\pi f_m t)}$ و سپس عبارت $e^{j2\pi f_c t}$ را تولید می کنیم و حاصل آن ها را در یکدیگر ضرب می کنیم و عبارت حقیقی را از آن ها جدا می کنیم. با استفاده از یک منبع سیگنال، سیگنال سینوسی با فرکانس 5K Hz ایجاد می کنیم و در جلوی آن یک بلوک ضرب کننده ثابت قرار می دهیم تا بتوانیم β را ایجاد کنیم، در اینجا ما می توانیم بتا و فرکانس سیگنال پیام را به صورت دلخواه تغییر دهیم که این طراحی در شکل ۱.۲.۲ مشخص شده است.



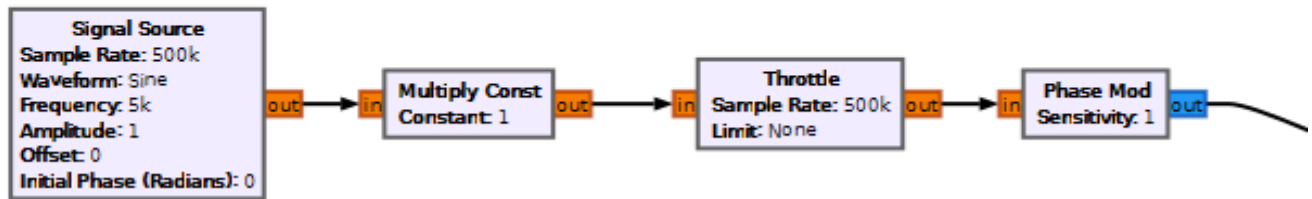
شکل ۱.۲.۲ تولید سیگنال پیام

با این طراحی توانستیم سیگنال $\beta \sin(2\pi f_m t)$ را ایجاد کنیم برای تبدیل این سیگنال به فرم نمایی طبق راهنمایی تصویر ۱.۲.۲ که از [سایت](#) برنامه برگرفته شده از بلوک Phase Mod استفاده می‌کنیم و به شکل ۲.۲.۲ می‌رسیم.

Phase Mod

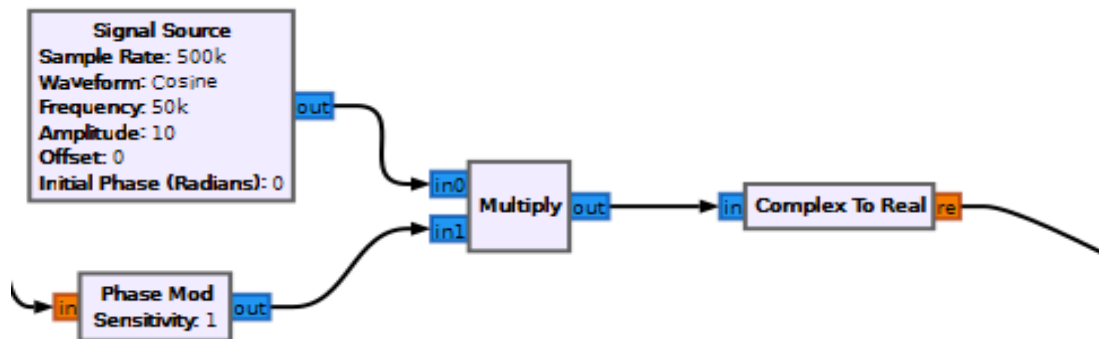
Phase modulator. output = complex(cos(in*sensitivity), sin(in*sensitivity))

تصویر ۱.۲.۲ کارکرد بلوک Phase Mod



شکل ۲.۲.۲ طراحی فرم نمایی

برای تولید سیگنال حامل نیز از تولید کننده سیگنال همانند گذشته بهره می‌گیریم و فرکانس حامل را برابر 50K Hz قرار می‌دهیم و سپس این سیگنال را در سیگنال پیام ضرب می‌کنیم و قسمت Re حاصل ضرب را جدا می‌کنیم که این بخش در شکل ۳.۲.۲ آورده شده است.



شکل ۳.۲.۲ اجرای مرحله دوم طراحی

پس از انجام این کار سیگنال مدوله شد $u(t)$ را در اختیار داریم و مثل طراحی قبل با استفاده از یک جمع کننده نویز را به این سیگنال اضافه می‌کنیم.

۳.۲. طراحی دمدولاسیون FM به همراه نویز در GNU Radio

برای طراحی دمدولاسیون از GPT کمک می‌گیریم که در تصویر ۱.۳.۲ قابل مشاهده است.

how can i demodulation FM signal in GNU?



To demodulate an FM (Frequency Modulated) signal in GNU Radio, follow these steps:

1. Understanding FM Demodulation

FM modulation encodes information in the **instantaneous frequency** of a carrier wave. To recover the original message signal, we need to extract the frequency variations.

The most common FM demodulation techniques are:

- Quadrature Demodulation (Differentiation + Envelope Detection)
- PLL (Phase-Locked Loop) Demodulation
- FIR Differentiation + Phase Detector

2. FM Demodulation in GNU Radio

Method 1: Quadrature Demodulation (Recommended)

♦ Blocks Needed:

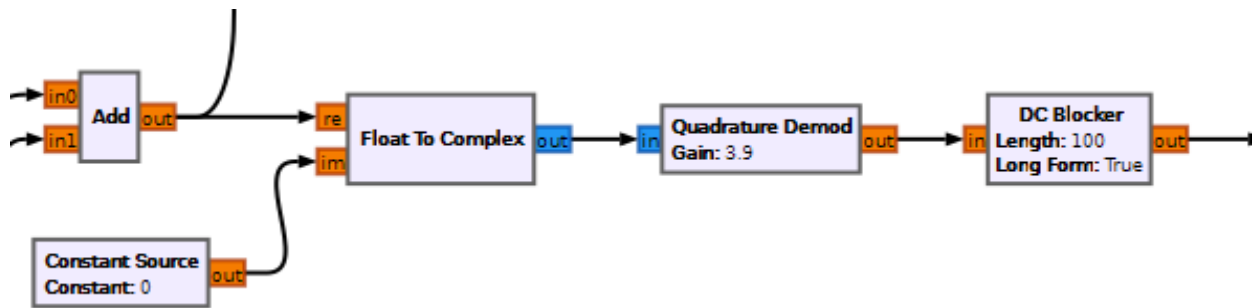
1. Complex Source / Signal Source (FM signal input)
2. Frequency Xlating FIR Filter (optional, for frequency shifting)
3. Quadrature Demod (Main demodulation block)
4. Low Pass Filter (to remove noise)
5. QT GUI Time Sink (to visualize output)

♦ Steps:

1. Add a "Quadrature Demod" block.
2. Connect the FM signal source (e.g., received signal) to its input.
3. Set Gain:
 - The gain should be $1 / (\text{Sensitivity} * \text{Sample Rate})$, where Sensitivity is the frequency deviation (e.g., 75 kHz for FM radio).
 - Example: If Sensitivity = 5kHz, Sample Rate = 100k, then Gain = $1 / (5000 / 100000) = 20$.
4. Add a Low Pass Filter to smooth the signal.
5. Connect to a QT GUI Sink (time/frequency) to view the demodulated output.

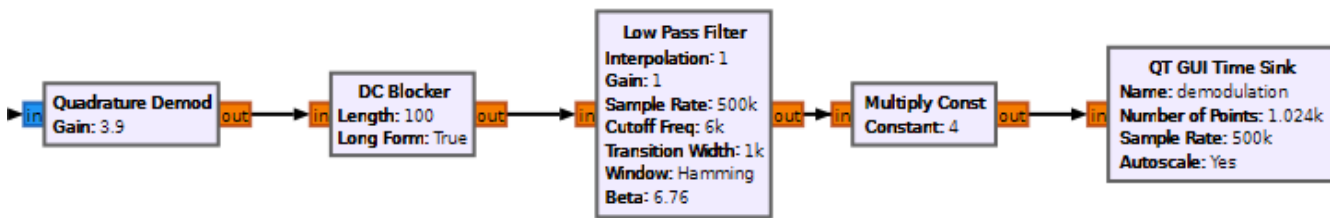
از بلوک Quadrature Demodulation استفاده می‌کنیم.

سیگنال‌های ورودی ممکن است شامل یک بخش ثابت باشند که به صورت پیش‌بینی نشده به سیگنال اضافه شده است، بنابراین برای حفظ ویژگی‌های اصلی سیگنال از بلوک DC Blocker استفاده می‌کنیم که این طراحی در شکل ۱.۳.۲ آورده شده است.



شکل ۱.۳.۲ شروع طراحی دمودلاسیون

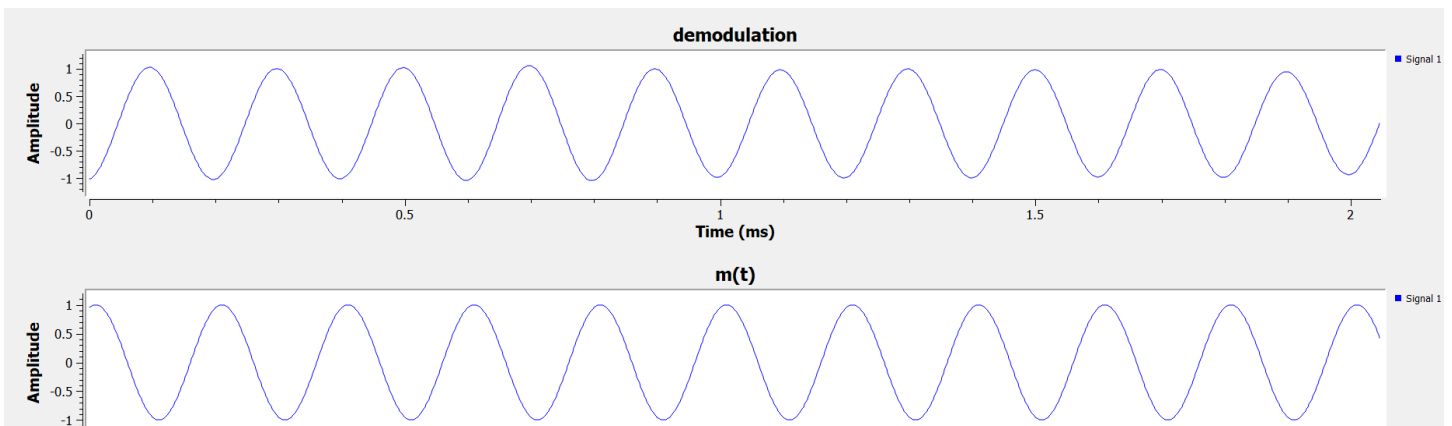
حال باید از یک فیلتر پایین‌گذر استفاده کرد تا تنها سیگنال پیام اصلی که در فرکانس‌های پایین وجود دارد جدا شوند و فرکانس‌های بالای اضافی (نویز و اغتشاش) را حذف کرد و سپس سیگنال را در ۴ ضرب می‌کنیم تا دامنه سیگنال دمودله شده با سیگنال پیام برابر شود، این طراحی در شکل ۲.۳.۲ قابل مشاهده است.



شکل ۲.۳.۲ طرح نهایی دمودلاسیون

۴.۲. مشاهده سیگنال پیام قبل و بعد از دمودلاسیون FM

تصویر ۱.۴.۲ نشان‌دهنده سیگنال قبل و بعد از دمودلاسیون به همراه نویز است که مشخصاً این دو سیگنال به یکدیگر شباهت بسیار زیادی دارند و می‌توان نتیجه گرفت دمودلاسیون طرح شده مناسب است.

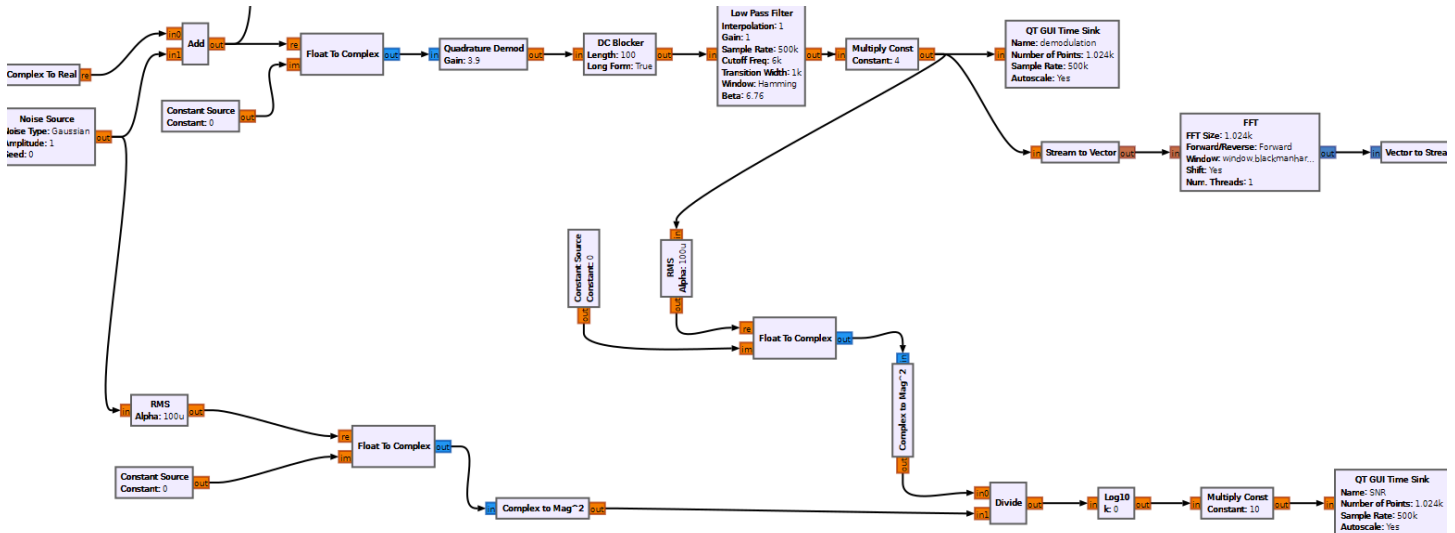


تصویر ۱.۴.۲ سیگنال پیام قبل و بعد از دمودلاسیون

۵.۲. مشاهده SNR در FM

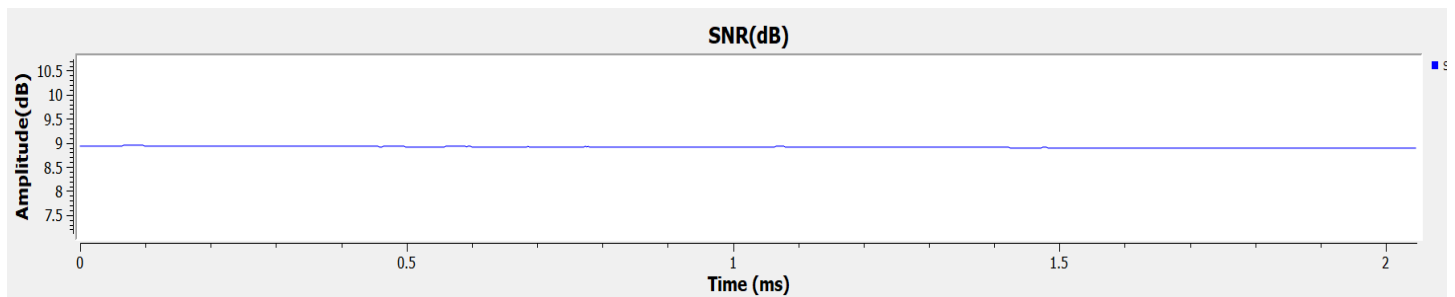
همانطور که می‌دانیم فرمول SNR در طراحی‌ها ثابت است و در این طراحی نیز می‌توان از فرمول‌های ۷.۱ و ۸.۱ استفاده نمود. برای محاسبه SNR در طراحی FM درست همانند طراحی VSB عمل می‌کنیم.

در شکل ۱.۵.۲ نحوه محاسبه SNR به صورت کلی آورده شده است.



شکل ۱.۵.۲ طراحی SNR در مدولاسیون FM

در خروجی به ازای بتا برابر ۴، SNR مانند تصویر ۱.۵.۲ خواهد بود.



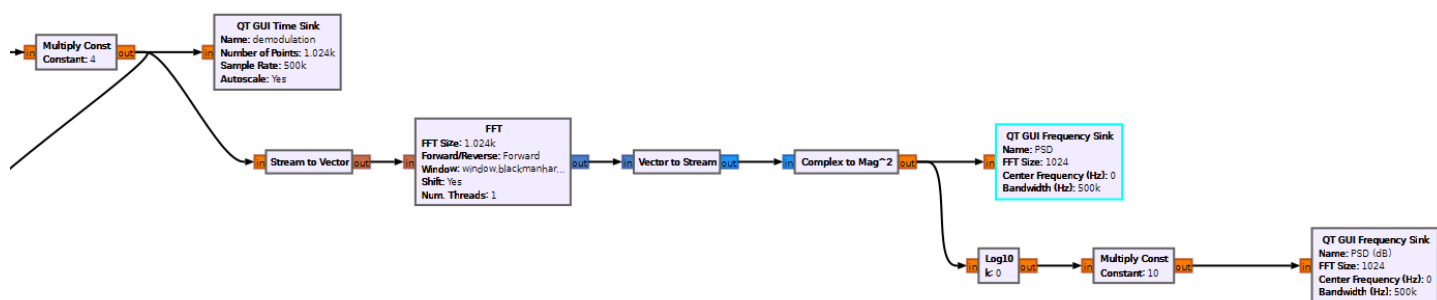
تصویر ۱.۵.۲ خروجی SNR طراحی FM

همانطور که قابل مشاهده است خروجی برابر 9dB است که میزان خوبی محسوب می‌شود.

۶.۲. مشاهده چگالی طیف توان (PSD) در FM

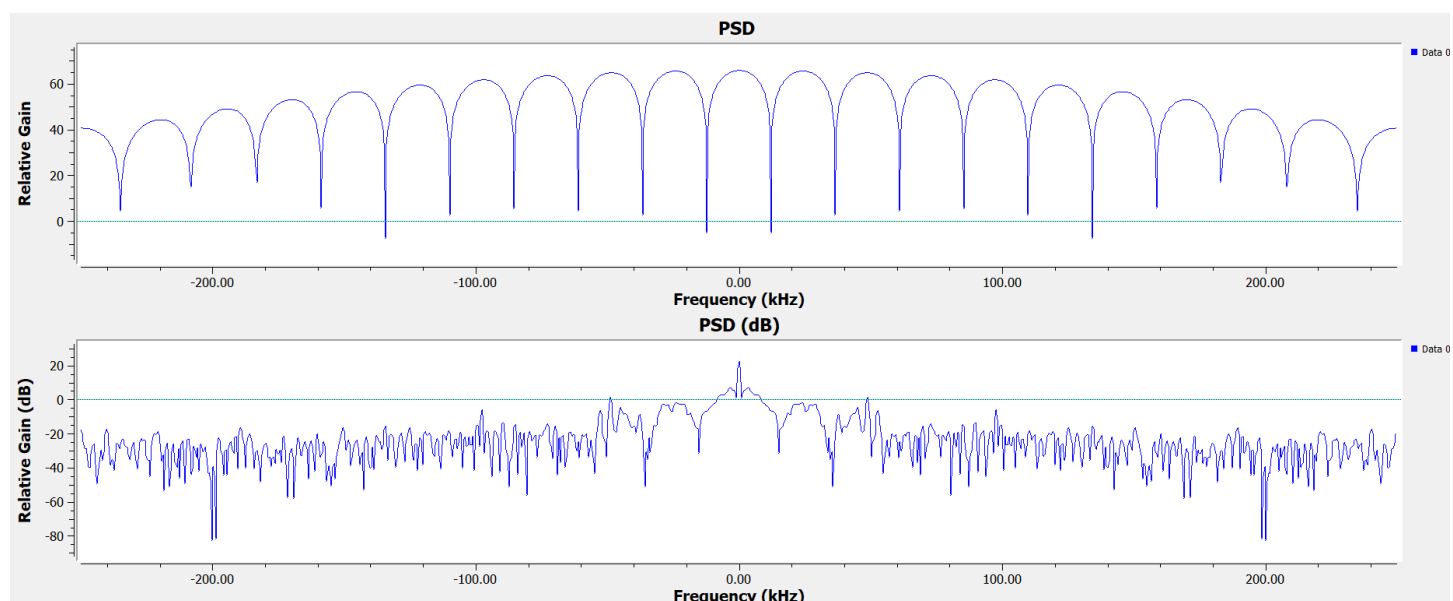
مجددا همانند بخش ۶.۱ عمل می‌کنیم.

طراحی بلوک برای مشاهده چگالی طیف توان در شکل ۱.۶.۲ آورده شده است و در نمایش چگالی طیف توان در این طراحی از اندازه بر حسب دسی بل نیز بهره گرفته شده است.



شکل ۱.۶.۲ طراحی برای مشاهده PSD

خروجی این طراحی در تصویر ۱.۶.۲ آورده شده است.



تصویر ۱.۶.۲ خروجی PSD

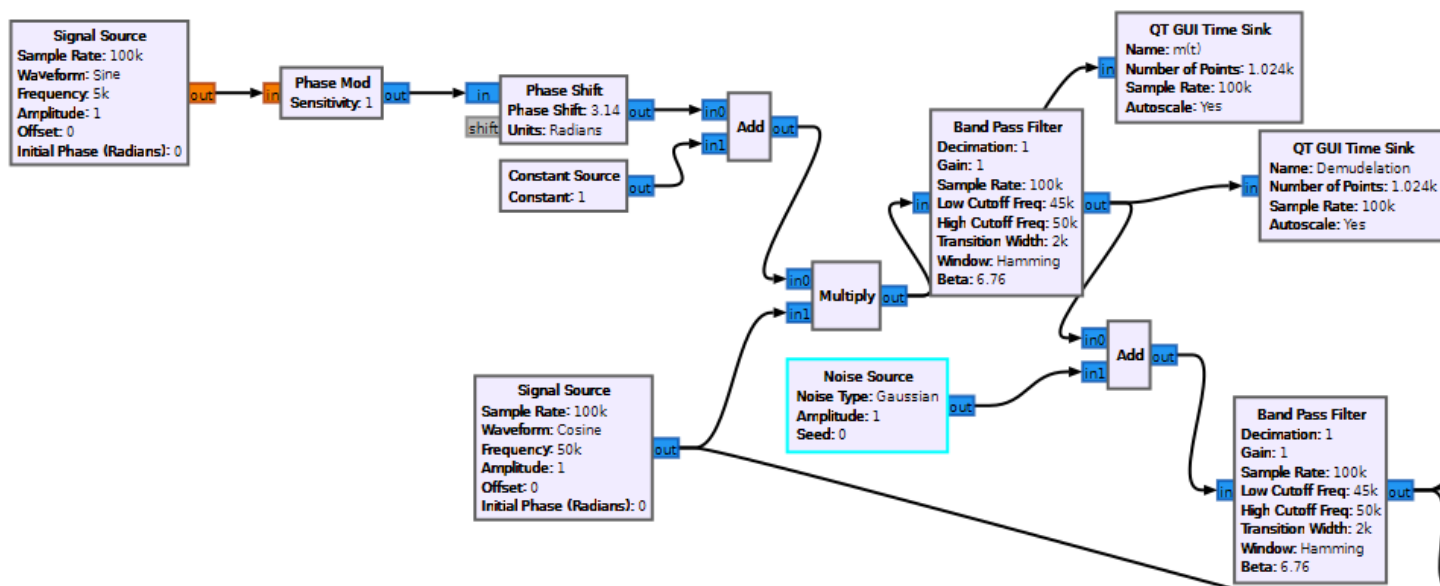
همانطور که از شکل پیداست چگالی طیف توان در نزدیکی فرکانس حامل توزیع شده است.

۳. طراحی امتیازی

در این بخش یک روش مدولاسیون خاص برای انتقال سیگنال طراحی شده است که ترکیبی از مدولاسیون فاز و مدولاسیون دامنه VSB را شامل میشود. این روش میتواند برای بهینه سازی پهنای باند و بهبود مقاومت در برابر نویز مورد استفاده قرار گیرد.

۱.۳. مدولاسیون امتیازی در GNU Radio

سیگنال پیام اولیه یک موج سینوسی است، این سیگنال ورودی اصلی سیستم مدولاسیون است. مدولاسیون ترکیبی شامل مدولاسیون فاز و مدولاسیون VSB است. شکل ۱.۱.۳ نشان دهنده سیگنال ارسال شده پس از اعمال این مدولاسیون است همچنین سیگنال نویز نیز به آن اضافه شده است.

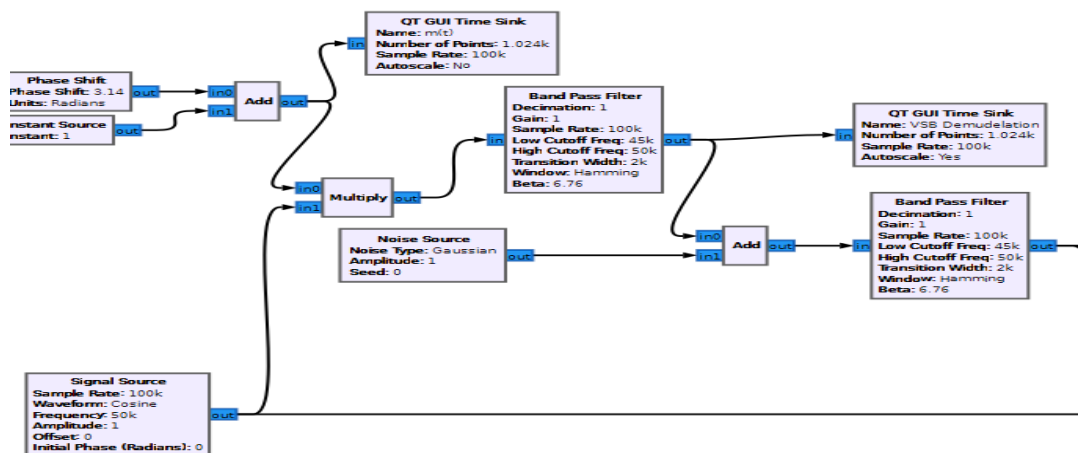


شکل ۱.۱.۳ سیگنال امتیازی به همراه نویز

۲.۳. مدولاسیون امتیازی در GNU Radio

در گیرنده، از فیلترهای باند-پایین و دمدولاتور فاز استفاده شده است تا سیگنال اصلی بازیابی شود. تصویر خروجی نشان میدهد که سیگنال بازسازی شده شباهت زیادی به سیگنال اصلی دارد، اما مقداری اعوجاج و کاهش دامنه مشاهده میشود که این طرح در شکل ۱.۲.۳ قابل مشاهده است.

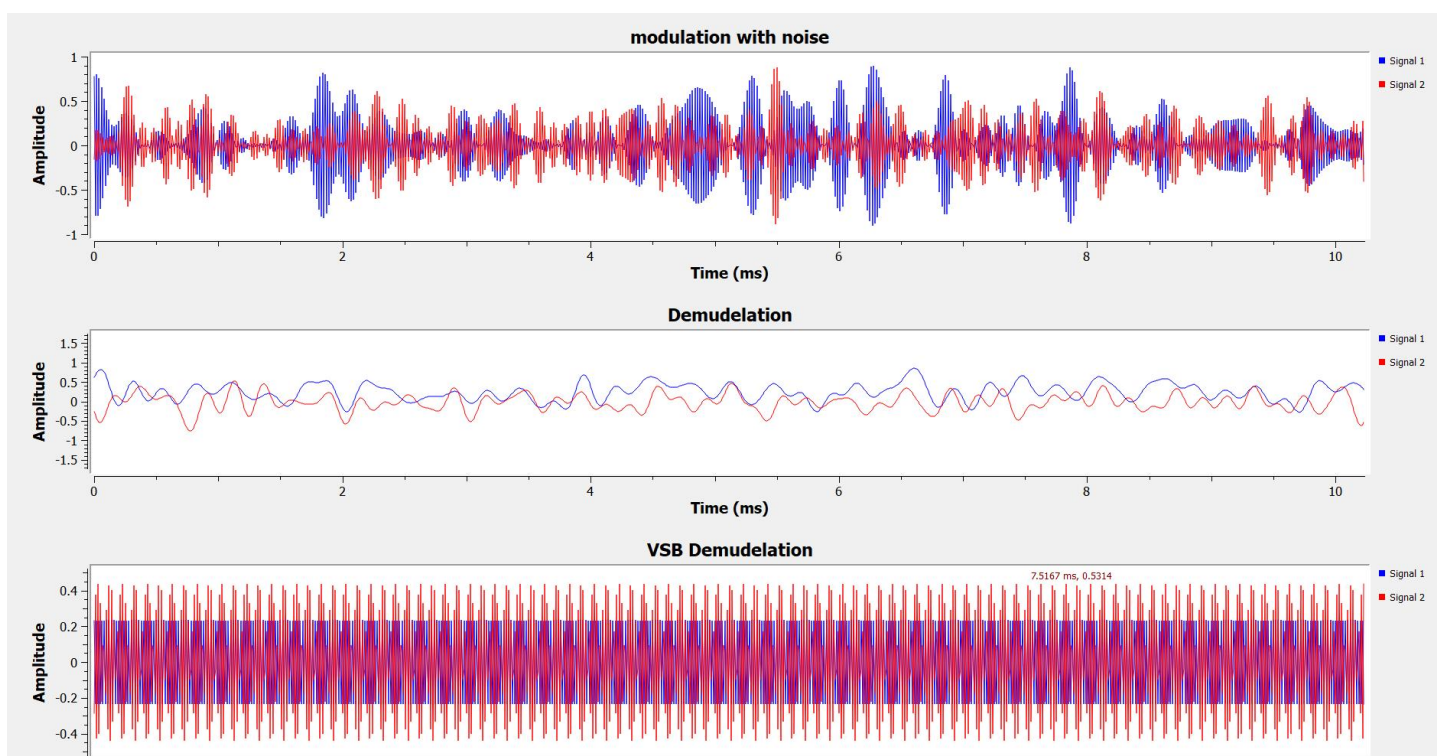
خروجی



۳.۳. مشاهده

امتیازی

مدولاسیون ویژه ترکیبی فاز + VSB بهین هسازی مناسبی برای سیگنال های باند محدود ارائه میدهد. مقدار SNR نسبت به VSB بهبود یافته اما همچنان مقداری اعوجاج باقی مانده است. این روش میتواند در کاربردهایی که نیاز به مقاومت بیشتر در برابر نویز دارند، مورد استفاده قرار گیرد



تصویر ۳.۳

۴. منابع

2. [Wiki GNU Radio](#)
3. Muhammed Mustaqim [YouTube Chanell](#)
4. GNU Radio [Github](#)