



بسمه تعالی

آزمایشگاه مخابرات دیجیتال

استاد: دکتر علی الفت

ویرایش ۱/۴ (آزمایشی)

فهرست مطالب آزمایشگاه مخابرات دیجیتال

| | |
|----|---|
| ۳ | مقدمه: معرفی آزمایشگاه مخابرات دیجیتال |
| ۵ | آزمایش اول: آشنایی با نرم افزار MATLAB |
| ۸ | آزمایش دوم: پردازش سیگنال دیجیتال با نرم افزار MATLAB |
| ۱۲ | آزمایش سوم: رادیونرم افزار و پایش طیف سیگنال های رادیویی |
| ۲۵ | آزمایش چهارم: معرفی مدولاسیون دیجیتال خطی |
| ۳۵ | آزمایش پنجم: کاوش در مدولاسیون دیجیتال خطی |
| ۴۱ | آزمایش ششم: مدولاسیون FSK همدوس |
| ۴۷ | آزمایش هفتم: آشکارسازی ناهمدوس |
| ۵۵ | آزمایش هشتم: انتقال دیجیتال از درون کانال باند محدود AWGN |



مقدمه

معرفی آزمایشگاه مخابرات دیجیتال

رویکرد فعالیت‌های آزمایشگاه مخابرات دیجیتال

به عنوان مقدمه، مطالبی در مورد رویکرد فعالیت‌های آزمایشگاه مخابرات دیجیتال ارائه می‌شود. آزمایشگاه مخابرات دیجیتال، در این ترم شامل ۱۲ جلسه می‌باشد. دانشجوی در طول این ۱۲ جلسه با مفاهیم مخابرات دیجیتال، ابزار شبیه‌سازی و یک سخت‌افزار آموزشی این حوزه آشنا می‌شود.

عمده‌ی آزمایشگاه‌هایی که در حوزه‌ی مخابرات سیستم برگزار می‌گردد، بر مبنای شبیه‌سازی است. این امر باعث می‌شود دانشجوی با چالش‌های پیاده‌سازی عملی الگوریتم‌ها آشنا نشود. علاوه بر این معمولاً سخت‌افزارهایی که پیاده‌سازی‌های عملی بر روی آن صورت می‌پذیرد، دارای پیچیدگی بالایی است. از این رو می‌بایست مدت زمان قابل توجهی صرف راه‌اندازی و آموزش این سخت‌افزارها شود. این امر باعث می‌شود فرصتی برای آموزش مفاهیم حوزه‌ی مخابرات دیجیتال باقی نماند.

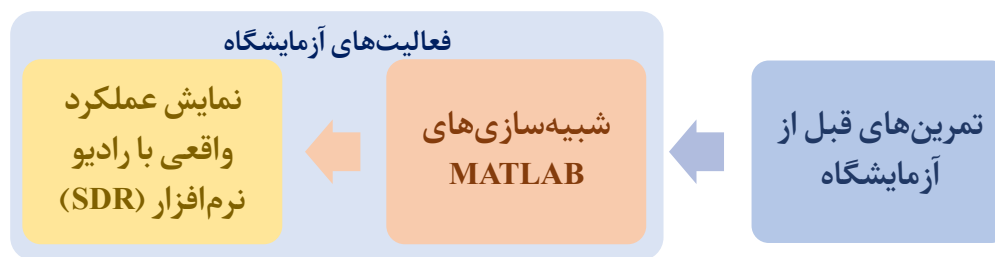
مطلب حائز اهمیت دیگر آن است که دانشجویانی که وارد این آزمایشگاه می‌شوند، تسلط کاملی نسبت به ابزارهای شبیه‌سازی ندارند و مدت زمان قابل توجهی در طول آزمایش‌ها، صرف آموزش این ابزارها می‌شود. بنابراین می‌بایست یک بسته‌ی آموزشی برای تسلط بر روی ابزارهای شبیه‌سازی تهیه شود. به منظور کاهش زمان این آموزش، می‌بایست تمرکز بر روی یک ابزار شبیه‌سازی باشد و راه‌اندازی سخت‌افزار و شبیه‌سازی‌های متکی بر آن با استفاده از همین یک ابزار انجام شود.

با استدلال‌های ذکر شده طراحی این آزمایشگاه مبتنی بر نرم‌افزار MATLAB و تنها اختصاص به بخش کدنویسی آن دارد. به علت محدودیت زمان، امکان آموزش بخش Simulink این نرم‌افزار نیز وجود ندارد. سخت‌افزارهایی که برای بخش عملی آزمایشگاه مورد استفاده قرار می‌گیرد، رادیو نرم‌افزار ADALM-PLUTO شرکت Analog Devices می‌باشد که در شکل ۱ نشان داده شده است. این دو رادیو نرم‌افزار را می‌توان به راحتی با برنامه‌ی MATLAB کنترل نمود و از آن‌ها برای دریافت و ارسال سیگنال استفاده نمود و یک سامانه‌ی کامل مخابرات دیجیتال را پیاده‌سازی کرد.



شکل ۱ رادیو نرم‌افزار ADALM-PLUTO

با توجه به شرایط آموزشی فعلی و برگزاری آزمایشگاه به صورت مجازی، بخش کار با سخت افزار کم رنگ شده است و تنها ویدیوهایی برای آشنایی با سخت افزار و ملاحظات عملی پیاده سازی ارایه می گردد. فعالیت های این آزمایشگاه شامل سه بخش تمرین های قبل از آزمایشگاه، شبیه سازی درون آزمایشگاه و نمایش عملکرد واقعی با رادیو نرم افزار می باشد. این روند در شکل ۲ نشان داده شده است.



شکل ۲ روند فعالیت های آزمایشگاه مخابرات دیجیتال

در این روند به صورت تدریجی پایه های هر مرحله در مرحله ی قبل چیده می شود تا آزمایش با موفقیت و با بیشترین بازده انجام شود. با این رویکرد، روند پیشرفت آزمایش ها با سرعت بالاتری انجام شده و دانشجو بر روی مطالب و مفاهیم مطرح شده در هر آزمایش تسلط بیشتری کسب خواهد نمود.

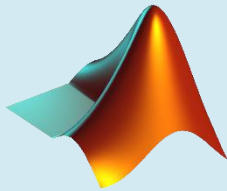
به منظور تسلط بر مباحث تئوری ارایه شده در درس های گذشته نظیر مخابرات دیجیتال و آمادگی برای شبیه سازی آزمایش ها، تمرین هایی به عنوان فعالیت قبل از کلاس در نظر گرفته شده است. دانشجویان با انجام تمرین ها قبل از آزمایشگاه، با مفاهیم کلیدی هر آزمایش آشنا می شوند. زمانی که پایه های تئوری هر آزمایش بنا شد، دانشجویان موظفند شبیه سازی های مربوط به طراحی و پیاده سازی اجزای یک سامانه ی مخابرات دیجیتال را انجام دهند و نتیجه ی کار را در محیط شبیه سازی نرم افزار مهندسی MATLAB ارزیابی و مشاهده نمایند. منطق این شبیه سازی ها آن است که دانشجو رفتار عملکردی اجزای یک سامانه ی مخابرات دیجیتال را در یک محیط کنترل شده مشاهده نماید. در نتیجه دانشجو آمادگی کافی جهت ورود به دنیای واقعی با عوامل غیر ایده آل فراوان را کسب می نماید. زمانی که مجموعه ی مهارت های دانشجویان با آمادگی تئوریک و شبیه سازی تقویت گردید، آمادگی لازم برای ورود به پیاده سازی اجزای سامانه ی مخابرات دیجیتال در یک محیط بی سیم واقعی به دست می آید. دانشجو در این مرحله با استفاده از ابزارهایی که نرم افزار MATLAB در اختیار می گذارد، عملکرد رادیو نرم افزار را مشاهده می نماید.

شیوه ی ارزیابی آزمایشگاه

در ادامه سهم هر یک از فعالیت های آزمایشگاه در نمره ی پایانی آمده است. دقت نمایید که همه ی مراحل به صورت انفرادی می باشد.

- تمرین های قبل از آزمایش: ۶ نمره
- انجام آزمایش ها: ۸ نمره
- گزارش کار آزمایشگاه: ۲ نمره
- پروژه ی پایانی یا آزمون: ۴ نمره
- حضور و غیاب: هر جلسه غیبت کسر ۱ نمره

در این ترم دستورکار آزمایشگاه در اختیار دانشجو قرار می گیرد. در روز آزمایشگاه توضیحاتی در مورد آزمایش به صورت زنده ارایه می شود. هر کدام از گروه ها در طول سه ساعت آزمایشگاه با همیاران آموزشی آزمایشگاه در ارتباط هستند. ارسال فایل ها در سایت elearn.ut.ac.ir انجام می شود و همیاران در تالار گفتگوی معرفی شده رفع اشکال می نمایند. فایل های نهایی نیز می بایست تا زمان مقرر در همان روز در سایت بازگزاری شود. در پایان روز ارایه ی درس، تمرین های قبل از آزمایش برای هفته ی آینده بازگزاری شده و دانشجویان می بایست تا یک روز قبل از شروع کلاس بعد آن را در سایت بازگزاری نمایند.



آزمایش اول

آشنایی با نرم افزار MATLAB


اهداف آزمایش


یکی از ابزارهای شبیه سازی و پیاده سازی سامانه ها و الگوریتم های مخابراتی نرم افزار MATLAB می باشد. در این آزمایشگاه به منظور انجام شبیه سازی ها، این نرم افزار مورد استفاده قرار می گیرد. در این آزمایش بناست تا آشنایی کافی با این نرم افزار به دست آید تا فرآیند آموزشی نرم افزار در طول آزمایش های آتی به حداقل رسیده و انجام آزمایش ها تسریع شود. عمده ی مطالبی که در اکثر کتاب های آموزشی MATLAB وجود دارد، چندان مورد استفاده قرار نمی گیرد و تنها فرآیند آموزش را ملال آور و کند می نماید. از این رو در این آزمایش تأکید بر آموزش مطالب و موارد پرکاربرد و مفید می باشد.

در انتهای این آزمایش دانشجو می بایست:

- 🔧 با محیط نرم افزار MATLAB آشنایی پیدا نماید.
- 🔧 انواع محاسبات و عملیات های ریاضی و محاسبات سمبلیک را به راحتی انجام دهد.
- 🔧 از بردارها، ماتریس ها و عملگرهای مرتبط با آنها استفاده نماید.
- 🔧 متغیرها و توابع داخلی نرم افزار MATLAB را بشناسد و از آنها استفاده نماید.
- 🔧 تسلط کافی در نوشتن M-File و Function داشته باشد.
- 🔧 اصول برنامه نویسی حاکم بر نرم افزار را بیاموزد.
- 🔧 از ابزارهای گرافیکی موجود بهره برده و انواع نمودارها را رسم نماید.

ابزارهای مورد نیاز

رایانه 

نرم افزار MATLAB R2020b 

شرح آزمایش



آزمایش ۱-۱: محاسبات جبری و گرافیک

۱. مقادیر زیر را محاسبه نمایید.

- ا. کسرهای $2709/1024$ ، $10583/4000$ و $2024/765$ کدام یک از این کسرها تقریب بهتری برای $\sqrt{7}$ است. (راهنمایی: کسرها را درون یک بردار قرار دهید. کسر دقیق تر را یکبار با مقایسه ارقام کسرها با ارقام $\sqrt{7}$ و یکبار با استفاده از دستور **min** جهت یافتن اندیس درایه ای از بردار، با کمینه ای اختلاف با $\sqrt{7}$ به دست آوردید.)
- ب. عدد 3^{301} را یک بار به صورت یک عدد اعشاری تقریبی با ۱۵ رقم اعشار (نمایش به صورت نماد علمی) و بار دیگر به صورت یک عدد صحیح دقیق محاسبه نمایید. (راهنمایی: از دستور **format** و دستور **vpa** با ورودی سمبلیک استفاده نمایید.)
- ت. $20/3 - 20 \times (1/3)$. جواب جبری برابر با ۰ است. اگر جواب ۰ نشد، علت را بیان کنید.
- ث. $10^{16} + 1 - 10^{16}$. به صورت جبری حاصل برابر با ۱ است. اگر جواب ۱ نشد، علت را بیان کنید.

۲. مقادیر زیر را تا ۱۵ رقم اعشار محاسبه نمایید. (راهنمایی: از دستور **vpa** استفاده ننمایید.)

ا. $\cosh(0.1)$

ب. $\ln(2)$

ت. $\arctan(1/2)$

۳. با استفاده از دستور **plot**، **fplot** و یا **ezplot** متناسب با نیاز، تابع های زیر را رسم نمایید.

- ا. $y = \sin(1/x^2)$ برای $-2 \leq x \leq 2$ و $-2 \leq y \leq 2$. سعی کنید از هر سه دستور **plot**، **fplot** و **ezplot** استفاده نمایید. آیا هر سه جواب درست است؟ اگر از دستور **plot** استفاده می نمایید، یک بار گام ها را برابر با ۰.۱ و یکبار گام ها را برابر ۰.۰۰۱ در نظر بگیرید. (راهنمایی: ابتدا نمودار را رسم کرده و سپس از دستور **axis** استفاده نمایید.)

ب. منحنی پروانه را با استفاده از معادله های پارامتری زیر رسم نمایید.

$$x = \sin(t) \left[e^{\cos(t)} - 2 \cos(4t) + \sin^5\left(\frac{t}{12}\right) \right]$$

$$y = \cos(t) \left[e^{\cos(t)} - 2 \cos(4t) + \sin^5\left(\frac{t}{12}\right) \right]$$

این نمودار را برای $0 \leq t \leq 2\pi$ و $0 \leq t \leq 10\pi$ و با گام های $\pi/100$ رسم نمایید.

آزمایش ۱-۲: ریاضیات و برنامه نویسی

۱. حاصل عبارت های زیر را به دست آورید و منطق جواب ها را توضیح دهید.

ا. $2 \times (3 < 8/4 + 2)^2 < (-2)^3$

ب. $(5 + \sim 0)/3 == 3 - \sim(10/5 - 2)$

ت. $\sim 4 < 5 | 0 > = 12/6$

ث. $-7 < -5 < -2 \& 2 + 3 < = 15/3$

۲. برای عدد صحیح مثبت n ، ماتریس $K(n)$ یک ماتریس $n \times n$ پایین مثلثی است که n سطر مثلث خیام را نمایش می دهد. برنامه ای بنویسید که ۷ سطر مثلث خیام را ایجاد نمایید.

$$K(5) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 \\ 1 & 3 & 3 & 1 & 0 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$



۳. برای یک عدد صحیح مثبت n ، ماتریس $A(n)$ یک ماتریس $n \times n$ است که درایه‌های آن در مکان (i, j) به صورت $a_{ij} = 1/(i + j - 1)$ است. برای مثال

$$A(3) = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

مقادیر ویژه‌ی $A(n)$ همه اعدادی حقیقی است. یک M-file بنویسد که بزرگترین مقدار ویژه‌ی $A(500)$ را بدون هیچ خروجی اضافی چاپ نماید. (راهنمایی: اگر از دو حلقه‌ی تو در تو استفاده نمایید، M-file مدت زمانی طول می‌کشد که اجرا شود. سعی کنید از این کار برحذر باشید! این کار را بدون استفاده از حلقه می‌توان انجام داد. برای به دست آوردن مقادیر ویژه از دستور **eig** استفاده نمایید.)

۴. برنامه‌ای بنویسید که

- ا. یک بردار با ۲۰ عدد صحیح تصادفی بین ۱۰ و ۳۰ تولید نماید.
- ب. همه‌ی درایه‌های فرد را با اعداد تصادفی دیگری بین ۱۰ و ۳۰ جایگزین نماید. (بدون استفاده از حلقه)
- ت. بخش (ب) را تا جایی که همه‌ی درایه‌ها اعدادی زوج شود، ادامه دهید.
- ث. برنامه می‌بایست تعداد دفعات تکرار بخش (ب) برای زوج شدن همه‌ی درایه‌های بردار را شمارش کند. وقتی این اتفاق رخ داد، برنامه می‌بایست بردار و متنی را نمایش دهد که بیان می‌کند که چندبار تکرار برای تولید این بردار نیاز بوده است. محتوای متن می‌بایست شبیه عبارت زیر باشد.

The Vector Generated After 10 Iteration(s) .

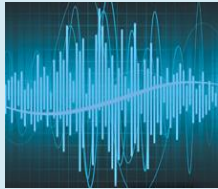
(کلید واژه‌های راهنما: **mod**, **randi**, **while**, **sprintf**)

۵. MATLAB تابعی به نام **lcm** دارد که کوچک‌ترین مضرب مشترک دو عدد را محاسبه می‌نماید. M-file تابع **mylcm.m** را بنویسید که کوچک‌ترین مضرب مشترک هر تعداد عدد مثبت صحیح را پیدا نماید. این اعداد می‌بایست بتوانند به صورت آرگومان‌های جداگانه و یا در قالب یک بردار وارد شوند. به عنوان مثال **mylcm(4, 5, 6)** و **mylcm([4 5 6])** می‌بایست عدد 60 را به عنوان جواب برگردانند. برنامه می‌بایست اگر یکی از ورودی‌ها مثبت نبود، یک پیغام خطای مفید تولید نماید. (راهنمایی: برای سه عدد می‌توان دستور **lcm** را برای به دست آوردن کوچک‌ترین مضرب مشترک دو عدد اول استفاده کرد و سپس از دستور **lcm** برای به دست آوردن کوچک‌ترین مضرب مشترک حاصل به دست آمده و عدد سوم استفاده نمود. M-file شما می‌بایست این رویکرد را تعمیم دهد.)



- [1] B. R. Hunt, R. L. Lipsman, and J. M. Rosenberg, *A Guide to MATLAB For Beginners and Experienced Users*. Cambridge: Cambridge University Press, 2014.
- [2] A. Gilat, *MATLAB: An Introduction with Applications*. Hoboken, NJ: John Wiley & Sons, Inc., 2017.





آزمایش دوم

پردازش سیگنال دیجیتال با نرم افزار MATLAB

اهداف آزمایش

پس از آشنایی با امکاناتی که نرم افزار MATLAB در اختیار قرار می دهد، در این آزمایش بناست تا این امکانات را در حوزه ی پردازش سیگنال های دیجیتال و سامانه های مخابراتی به کار ببندیم. در طول آزمایش های بعدی مکرر با سیگنال های گسسته زمان سر و کار داریم. از این رو دانشجو می بایست کار با این سیگنال های گسسته زمان، فرایندها و محاسباتی که بر روی این سیگنال ها صورت می پذیرد به منزله ی الفبا بیاموزد و بر روی آن ها مسلط باشد. به عبارت دیگر در آزمایش های بعد مکرر با مفاهیم، ابزارها و روش هایی از حوزه ی پردازش سیگنال و سامانه های مخابراتی روبه رو هستیم. از این رو می بایست این موارد در ابتدای این آزمایشگاه آموخته شود تا بتوان اصول سامانه های مخابرات دیجیتال را با سرعت و کیفیت بهتری فراگرفت.

در انتهای این آزمایش دانشجو می بایست:

- بتواند سیگنال های گسسته زمان مختلف را به همراه اندیس گذاری زمانی تولید نماید.
- بتواند عملیات مختلف نظیر جمع، ضرب، مقیاس کردن، جابه جایی زمانی و ... یک یا چند سیگنال را انجام دهد.
- خروجی عبور سیگنال از یک سامانه ی خطی تغییرناپذیر با زمان را با عمل کانولوشن به دست آورد.
- کار با معادلات تفاضلی و پیاده سازی یک سیستم گسسته زمان با استفاده از این معادلات در نرم افزار MATLAB آشنا باشد.
- بتواند طیف فرکانسی سیگنال های گسسته زمان را به دست آورد.
- تحلیل سامانه های خطی تغییرناپذیر با زمان را در حوزه فرکانس انجام دهد.
- بتواند به راحتی توان و انرژی سیگنال های گسسته زمان را به دست آورد.
- متغیرهای تصادفی را تولید نماید و آمارگان های آن ها را محاسبه نماید.
- بتواند چگالی طیف توان یک فرآیند تصادفی را به دست آورد.
- به مفهوم همبستگی و کاربردهای مسلط باشد.

ابزارهای مورد نیاز

رایانه

نرم افزار MATLAB R2020b

شرح آزمایش

آزمایش ۱-۲: تولید سیگنال دیجیتال و عملیات بر روی آن

۱. دنباله‌های تصادفی: دنباله‌های تصادفی زیر را تولید نمایید. همچنین با استفاده از دستور **histogram**، هیستوگرام این دنباله را که بر اساس تابع چگالی احتمال، به‌هنگار شده، رسم نمایید. تعداد میله‌های نمودار برابر ۱۰۰ باشد. برای محاسبه‌ی دنباله‌های زیر می‌توان از دستورهای **rand** و **randn** استفاده نمود.
 - ا. $x_1[n]$ یک دنباله‌ی تصادفی با نمونه‌های مستقل و با توزیع گاوسی با میانگین ۱۰ و واریانس ۱۰ می‌باشد. ۱۰۰۰۰ نمونه‌ی این دنباله را تولید نمایید. (راهنمایی: $(X \sim \mathcal{N}(0,1) \rightarrow \sigma X + m \sim \mathcal{N}(m, \sigma^2))$)
 - ب. $x_3[n] = x_2[n] + x_2[n-1]$ که در آن $x_2[n]$ دنباله‌ای با نمونه‌های مستقل و با توزیع یک‌نواخت در بازه‌ی $[0,2]$ می‌باشد. ۱۰۰۰۰۰ نمونه‌ی این دنباله را تولید نمایید. چرا شکل نمودار هیستوگرام بدین شکل است؟ (راهنمایی: $(X \sim U(0,1) \rightarrow aX + b \sim U(b, a+b))$)
 - ت. $x_4[n] = \sum_{k=1}^4 y_k[n]$ که هر یک از دنباله‌های $y_k[n]$ مستقل از بقیه و نمونه‌های آن به صورت یک‌نواخت بین $[-0.5, 0.5]$ توزیع شده است. ۱۰۰۰۰۰ نمونه‌ی این دنباله را تولید نمایید. نمودار هیستوگرام به چه شکلی در آمده و علت آن چیست؟

۲. کاهش نرخ نمونه‌برداری: عمل کاهش نرخ نمونه‌برداری یک سیگنال به صورت $y[n] = x[nM]$ تعریف می‌شود. در این جا نرخ نمونه‌برداری سیگنال با ضریب صحیح M کاهش می‌یابد. به عنوان مثال اگر نرخ نمونه‌برداری دنباله‌ی $x[n] = \{-2, 4, 3, -6, 5, -1, 8\}$ را با ضریب ۲ کاهش دهیم به دنباله‌ی $y[n] = \{-2, 3, 5, 8\}$ دست پیدا می‌نماییم.
 - ا. در نرم‌افزار MATLAB تابع **dwnsmp1** را به صورت زیر بنویسید تا عملیات فوق را انجام دهد.
`function [y] = dwnsmp1(x,M)`
 - ب. تابع $x[n] = \sin[0.125\pi n]$ را در بازه‌ی $-50 \leq n \leq 50$ تولید نمایید. نرخ نمونه‌برداری $x[n]$ را با ضریب ۴ کاهش دهید و $y[n]$ را تولید نمایید. هر دو دنباله‌ی $x[n]$ و $y[n]$ را با انتخاب مناسب نقاط محور افقی با استفاده از دستور **stem** بر روی هم رسم نمایید.
 - ت. عملیات فوق را برای تابع $x[n] = \sin[0.5\pi n]$ و در بازه‌ی $-50 \leq n \leq 50$ تکرار نمایید.

۳. افزایش نرخ نمونه‌برداری: عمل افزایش نرخ نمونه‌برداری یک سیگنال به صورت $y[n] = x[n/N]$ تعریف می‌شود. در این جا نرخ نمونه‌برداری سیگنال با ضریب صحیح N افزایش می‌یابد. به عنوان مثال اگر نرخ نمونه‌برداری دنباله‌ی $x[n] = \{-2, 3, 5, 8\}$ را با ضریب ۲ افزایش دهیم به دنباله‌ی $y[n] = \{-2, 0, 3, 0, 5, 0, 8\}$ دست پیدا می‌نماییم.
 - ا. در نرم‌افزار MATLAB تابع **upsmp1** را به صورت زیر بنویسید تا عملیات فوق را انجام دهد.
`function [y] = upsmp1(x,N)`
 - ب. با استفاده از مثال موجود در صورت سوال عملکرد تابع را ارزیابی نمایید.

آزمایش ۲-۲: عبور سیگنال گسسته از یک سیستم

۱. حل معادله‌ی تفاضلی: یک سیستم خطی و تغییر ناپذیر با زمان به صورت معادله‌ی تفاضلی زیر تعریف می‌شود.

$$y[n] - 0.5y[n-1] + 0.25y[n-2] = x[n] + 2x[n-1] + x[n-3]$$
 - ا. بدون استفاده از دستور **filter** پاسخ ضربه‌ی سیستم فوق را در بازه‌ی $0 \leq n \leq 100$ را رسم نمایید. با استفاده از دستور **filter** محاسبات خود را صحت‌سنجی نمایید.
 - ب. اگر ورودی سیستم به صورت $x[n] = (5 + 3 \cos[0.2\pi n] + 4 \sin[0.6\pi n])u[n]$ باشد، پاسخ خروجی $y[n]$ را در بازه‌ی $0 \leq n \leq 200$ را بدون استفاده از دستور **filter** تعیین نمایید.

¹ Normalized

۲. تابع همبستگی و کاربرد آن:

ا. تابعی بنویسید که با استفاده از فرم مختلط فرمول مستقیم همبستگی، همبستگی دو دنباله‌ی x و y را محاسبه نماید. عنوان تابع به صورت `corr_m(x, y)` باشد. در این جا طول خروجی برابر با طول سیگنال بلندتر منهای طول سیگنال کوچک‌تر به علاوه‌ی ۱ می‌باشد. به عبارتی تنها نمونه‌هایی که اشتراک دو سیگنال برابر با طول سیگنال کوچک‌تر است، در خروجی ظاهر شود.

ب. تعدادی بسته‌ی داده با یک مدولاسیون دیجیتال از یک فرستنده به سمت یک گیرنده ارسال می‌شود. ابتدای هر بسته یک هدر وجود دارد تا بتوان به وسیله‌ی آن هدر ابتدای بسته را پیدا کرد. سیگنال ارسالی در هوا با سرعت نور $c = 3 \times 10^8$ m/s (سرعت انتشار نور) منتشر می‌شود و به گیرنده می‌رسد. فرض کنید گیرنده لحظه‌ی شروع ارسال را می‌داند و از آن جا شروع به دریافت سیگنال می‌نماید (تحقق عملی دشواری دارد). لحظه‌ای که سیگنال دریافت می‌شود نسبت به لحظه‌ی ارسال بسته با τ مشخص می‌شود و فاصله‌ی گیرنده و فرستنده نیز با $R = c\tau$ تعیین می‌شود. فایل `PacketsAndHeader.mat` شامل سیگنال دریافتی در گیرنده و هدر آن می‌باشد. با استفاده از تابع همبستگی و با فرض ثابت بودن فاصله‌ی گیرنده و فرستنده، این فاصله و طول زمانی هر بسته را به دست آورید. اگر بدانید تعداد نمونه‌های زمانی هر سمبل در این فایل ارسالی برابر با ۸ است، تعداد سمبل‌ها رو نیز حساب کنید. نمودار بخش حقیقی و موهومی مربوط به داده‌های هر دو متغیر موجود در فایل ذکر شده را نیز رسم نمایید. فرکانس نمونه‌برداری مبدل آنالوگ به دیجیتال گیرنده و فرستنده برابر با 10MHz بوده است. (راهنمایی: می‌توانید از تابع `load` برای بازکردن فایل‌های داده شده استفاده کنید).

آزمایش ۳-۲: تحلیل حوزه‌ی فرکانس

۱. محاسبه‌ی طیف سیگنال: دستور `fft` در نرم افزار MATLAB عمل تبدیل فوریه‌ی گسسته‌ی سریع را انجام می‌دهد. این دستور تنها تعدادی نمونه‌ی زمانی دریافت کرده و به همان اندازه و یا برابر تعداد نقاطی که به دستور گفته می‌شود، نمونه‌ی فرکانسی تولید می‌نماید. فرض کنید فرکانس نمونه‌برداری برابر 250MHz می‌باشد. با این فرکانس نمونه‌برداری، سیگنال $x(nT_s) = x[n] = Ae^{j2\pi f_0 nT_s}$ را تولید نمایید و طیف فرکانسی آن را در بازه‌ی $[-125, 125]$ MHz را رسم نمایید. (راهنمایی: از کد موجود در بسته‌ی آموزشی ۲ می‌توان استفاده کرد).

ا. $f_0 = \frac{250 \times 51}{256}$ MHz, $A = 2$ و تعداد نقاط FFT برابر ۲۵۶ نقطه است.

ب. $f_0 = \frac{250 \times 51.5}{256}$ MHz, $A = 2$ و تعداد نقاط FFT برابر ۲۵۶ نقطه است.

ت. $f_0 = \frac{250 \times 51}{256}$ MHz, $A = 2$ و تعداد نقاط FFT برابر ۵۱۲ نقطه است.

ث. $f_0 = -\frac{250 \times 51}{256}$ MHz, $A = 2$ و تعداد نقاط FFT برابر ۵۱۲ نقطه است.

ج. $f_0 = \frac{250 \times 153}{256}$ MHz, $A = 4$ و تعداد نقاط FFT برابر ۵۱۲ نقطه است.

ح. علت انتخاب فرکانس‌های فوق را کاملاً درک نمایید. به منظور درک بیشتر در هر مرحله نسبت طول زمانی یک فریم $FFT(N_{FFT}T_s)$ را به دوره تناوب سیگنال $(1/f_0)$ را به دست آورید. مفهوم نشتی طیف را با استفاده از مشاهدات صورت گرفته توضیح دهید.

خ. اثر تعداد نقاط FFT را بر روی دامنه‌ی طیف مشاهده نمایید و سعی کنید با افزودن ضریب مناسب در کد موجود در بخش تئوری این اثر را خنثی کنید. به عبارتی دامنه‌ی طیف می‌بایست برابر A^2 باشد.

د. اثر تعداد نقاط FFT را بر قابلیت تفکیک فرکانسی چیست. برای این منظور فرکانس f_0 را به مقدار کمی تغییر دهید تا مقدار فرکانس آن از یک خانه‌ی FFT به خانه‌ی بعد منتقل شود.

ذ. توان سیگنال را در حوزه‌ی زمان محاسبه نمایید و با توان به دست آمده در حوزه‌ی فرکانس مقایسه کنید.

ر. در عمل به جای گزارش دامنه از توان سیگنال بر حسب dBm استفاده می‌شود. معمولاً توان را نیز با فرض

مقاومت ۵۰ اهم از روی ولتاژ موثر به دست می‌آورند. به عنوان مثال برای سیگنال $x[n] = A \cos[2\pi f_0 nT_s]$

تبدیل بین دامنه و توان بر حسب dBm به صورت زیر انجام می‌شود.

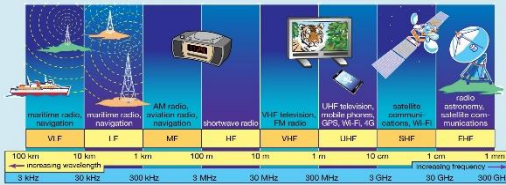
² Spectral Leakage

$$A \rightarrow \text{دامنه موثر} \frac{A}{\sqrt{2}} \rightarrow \text{توان موثر} \frac{A^2}{2R} \text{ watt} \rightarrow 10 \log \frac{A^2}{2R} \text{ dBW} \rightarrow \left(10 \log \frac{A^2}{2R} + 30 \right) \text{ dBm}$$

حال با این اطلاعات طیف سیگنال بخش A و B را بر حسب dBm است رسم نمایید. در نرم افزار MATLAB با استفاده از دستور **db** بسیاری از این تبدیل ها انجام می گیرد که اعمال مقاومت در آن نیازمند دقت است.



- [1] V. K. Ingle and J. G. Proakis, *Digital Signal Processing using MATLAB*. 2010.
- [2] J. G. Proakis, M. Salehi, and G. Bauch, *Contemporary Communication Systems Using MATLAB®*. 2013.
- [3] S. K. Mitra, *Digital signal processing: a computer-based approach*. New Delhi: Mc Graw Hill Education, 2011.
- [4] Ingle, V. K., & Proakis, J. G. (2012). *Digital signal processing using MATLAB*. Stanford: Cengage Learning.
- [5] T. F. Collins, R. Getz, D. Pu, and A. M. Wyglinski, *Software-defined radio for engineers*. Norwood, MA: Artech House, 2018.



آزمایش سوم

معادل باندپایه و میان گذر به همراه آشنایی با رادیونرم افزار

اهداف آزمایش

طیف رادیویی به بخشی از طیف الکترومغناطیس در محدوده‌ی فرکانسی 3kHz تا 300GHz گفته می‌شود. از طیف رادیویی به صورت گسترده در سرویس‌های مخابراتی نظیر ارسال سیگنال‌های تلویزیون، رادیو، موبایل، WiFi و سامانه‌های ناوبری و آشکارسازی نظیر رادار، GPS، دیده‌بانی رادیویی، فرستنده‌ها و ... استفاده می‌شود. در این آزمایش استفاده از سخت‌افزار ADALM-PLUTO شروع می‌شود و بنابر طیف فرکانسی محدوده‌ی 25MHz تا 6GHz مورد پایش قرار بگیرد. آزمایش‌هایی که صورت می‌پذیرد کمک می‌کند تا این بخش از طیف مورد کنکاش قرار بگیرد و دانشجو به جست‌جو و شناسایی سیگنال‌های رادیویی در این محدوده‌ی فرکانسی بپردازد. از جمله مفاهیمی که در رادیو نرم‌افزارها و سامانه‌های مخابراتی مورد استفاده قرار می‌گیرد، مفهوم معادل پایین‌گذر و میان‌گذر سیگنال است. دانشجو در این آزمایش با این مفاهیم آشنایی کامل پیدا می‌کند.

در انتهای این آزمایش دانشجو می‌بایست:

- تسلط کامل بر روی مفاهیم معادل پایین‌گذر و بالاگذر داشته باشد و بتواند این دو معادل را به یکدیگر تبدیل نماید.
- با طیف فرکانسی محدوده‌ی 25MHz تا 6GHz آشنا باشد و درک مناسبی از سیگنال‌های موجود در این گستره‌ی فرکانسی داشته باشد.
- آشنایی کامل با بسته‌ی رادیو نرم‌افزار ADALM-PLUTO داشته باشد و اتصالات کابل‌ها و آنتن‌ها را به رادیو نرم‌افزار انجام دهد.
- آشنایی ضمنی با نحوه‌ی افزودن این رادیو نرم‌افزارها به نرم‌افزار MATLAB را داشته باشد.
- با ساختار رادیو نرم‌افزارهای ADALM-PLUTO آشنایی کافی داشته باشد.
- بتواند پارامترهای رادیو نرم‌افزارها را با استفاده از یک برنامه‌ی MATLAB تنظیم نماید و رادیو نرم‌افزارها را در فرکانس دلخواه تیون نماید.
- بتواند داده‌های خروجی رادیو نرم‌افزار را دریافت نموده و طیف آن را به صورت به‌لحظه رسم نماید.
- سیگنال‌های رادیو FM، سیگنال‌های تلفن همراه نسل دوم، سوم و چهارم، سیگنال ریموت درب، سیگنال صوت و تصویر دیجیتال را پایش نموده و به ویژگی‌های طیف فرکانسی مسلط باشد.

ابزارهای مورد نیاز

رایانه

نرم‌افزار MATLAB R2020b

- MathWorks DSP System Toolbox
- MathWorks Communications System Toolbox
- MathWorks Signal Processing Toolbox

رادیو نرم‌افزار ADALM-PLUTO

کابل کوکسیال و آنتن



تئوری

معادل باندپایه‌ی مختلط سیگنال های میان گذر

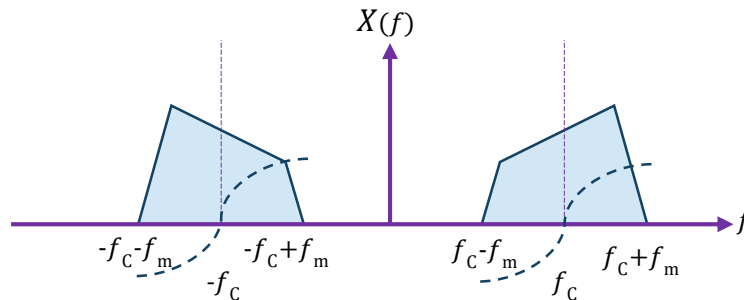
تقریباً همه‌ی سامانه‌های مخابراتی با مدوله کردن یک شکل موج حاوی اطلاعات بر روی یک سیگنال حامل سینوسی پیاده‌سازی می‌شوند. فرکانس حامل سیگنال ارسالی حاوی هیچ اطلاعاتی نیست و در واقع سیگنال مدوله‌کننده‌ی حامل در برگیرنده‌ی اطلاعات است. از این رو به توصیف سیگنال‌های مخابراتی، مستقل از فرکانس حامل نیاز است. در نتیجه‌ی این نیاز، مهندسین سامانه‌های مخابراتی برای سادگی کارها از معادل باندپایه‌ی مختلط یا معادل پایین‌گذر سیگنال‌های مخابراتی استفاده می‌نمایند. تمام سامانه‌های مخابراتی می‌تواند با بیان باندپایه‌ی مختلط تحلیل شود و معمولاً نیز چنین کاری صورت می‌پذیرد. یکی از مزایای معادل باندپایه‌ی مختلط سادگی است. همه‌ی سیگنال‌ها پایین‌گذر هستند و ایده‌های پایه‌ای مدولاسیون و پردازش سیگنال مخابراتی به سادگی توسعه می‌یابد. علاوه بر این عمده‌ی گیرنده‌هایی که شکل موج‌های دریافتی را به صورت دیجیتال پردازش می‌نمایند، از معادل باندپایه‌ی مختلط برای توسعه‌ی الگوریتم‌های پردازش استفاده می‌نمایند.

گام اول در بسط معادل باندپایه‌ی مختلط تعریف سیگنال باندپایه است.

تعریف کیفی سیگنال میان‌گذر: سیگنال میان‌گذر $x(t)$ سیگنالی است که مرکز طیف انرژی یک طرفه‌ی آن در فرکانس غیرصفر f_c باشد و دنباله‌ی این طیف تا فرکانس صفر ادامه ندارد.

تعریف ریاضی سیگنال میان‌گذر: به عبارت دیگر سیگنالی میان‌گذر است که حقیقی باشد و تبدیل فوریه‌ی آن دارای این ویژگی است که بتوان f_c و f_m را به نحوی پیدا کرد که $f_c > f_m$ باشد و $\forall f: |f| - f_c > f_m \rightarrow X(f) = 0$

اگر پهنای باند دوطرفه‌ی انتقال سیگنال میان‌گذر برابر با $2f_m$ هرتز باشد، طیف یک‌طرفه‌ی سیگنال میان‌گذر تنها در بازه‌ی فرکانسی $[f_c - f_m, f_c + f_m]$ غیرصفر می‌باشد. این مسأله بدان معناست که سیگنال‌های میان‌گذر دارای قید $f_m < f_c$ می‌باشند. شکل ۱ نمونه‌ای از یک طیف میان‌گذر را نشان می‌دهد. از آن‌جا که یک سیگنال میان‌گذر $x_c(t)$ سیگنالی است که به صورت فیزیکی قابل تحقق است، سیگنال حقیقی است و انرژی سیگنال نسبت به فرکانس $f = 0$ متقارن است. اندازه‌ی نسبی B_T و f_c اهمیت نداشته و تنها می‌بایست طیف سیگنال مقدار ناچیزی در حول فرکانس DC داشته باشد. در مخابرات تلفنی این ناحیه‌ی طیف، ناچیز و در حدود 300Hz است در حالی که در مخابرات ماهواره‌ای این مقدار می‌تواند چندین گیگاهرتز باشد.



شکل ۱ پاسخ فرکانسی یک سیگنال حقیقی میان‌گذر

یک سیگنال میان‌گذر دارای بیانی به صورت زیر است.

$$x(t) = x_i(t) \cos 2\pi f_c t - x_q(t) \sin 2\pi f_c t \quad (1)$$

$$= x_a(t) \cos(2\pi f_c t + x_p(t)) \quad (2)$$

در رابطه‌ی فوق f_c فرکانس حامل بوده که در محدوده‌ی $f_c - f_m \leq f_c \leq f_c + f_m$ قرار دارد. معمولاً به سیگنال $x_i(t)$ در معادله‌ی (۱) مولفه‌ی هم‌فاز سیگنال و به سیگنال $x_q(t)$ مولفه‌ی غیرهم‌فاز سیگنال میان‌گذر گفته می‌شود. $x_i(t)$ و $x_q(t)$ سیگنال‌های پایین‌گذر با مقادیر حقیقی هستند که طیف یک‌طرفه‌ی انرژی آن در فرکانس‌های بالاتر از $2f_m$ دارای مقدار ناچیزی است. می‌بایست توجه کرد که فرکانس مرکزی سیگنال میان‌گذر f_c و فرکانس حامل f_c همواره با یکدیگر یکسان نیستند. در حالی که f_c به صورت تئوری می‌تواند زنجیره‌ای از مقادیر را اختیار کند، مقدار بدیهی f_c می‌تواند منجر به ساده‌ترین بیان شود. برای سیگنال حامل معمولاً یک عبارت کسینوسی

³ Complex Baseband Representation



در نظر گرفته می شود. از این رو مولفه ی $x_i(t)$ هم فاز با سیگنال حامل است. به صورت مشابه عبارت سینوسی به اندازه ی ۹۰ درجه با کسینوس یا عبارت سیگنال حامل اختلاف فاز دارد و از این رو مولفه ی $x_q(t)$ با حامل غیرهم فاز می باشد. **معادله ی (۱) با عنوان فرم کانونی سیگنال میان گذر شناخته می شود.** **معادله ی (۲) صورت دامنه و فاز سیگنال میان گذر است** که در آن $x_a(t)$ دامنه ی سیگنال و $x_p(t)$ فاز سیگنال است. سیگنال میان گذر دارای دو درجه ی آزادی است و بیان های کانونی I/Q با دامنه و فاز با یکدیگر معادل هستند. تبدیل بین این دو بیان به صورت زیر می باشد.

$$x_a(t) = \sqrt{x_i^2(t) + x_q^2(t)}, \quad x_p(t) = \tan^{-1}(x_q(t), x_i(t)) \quad (3)$$

$$x_i(t) = x_a(t) \cos x_p(t), \quad x_q(t) = x_a(t) \sin x_p(t) \quad (4)$$

می بایست دقت نمود که تابع تناؤت در معادله ی (۳) دارای برد $[-\pi, \pi]$ می باشد و به عبارتی علامت هر دو $x_i(t)$ و $x_q(t)$ و نسبت $x_i(t)$ و $x_q(t)$ برای محاسبه ی تابع مورد نیاز است. جزییات تحلیل طراحی مخابراتی تعیین می کند که کدام صورت سیگنال باندپایه راحت تر به کار گرفته می شود.

یک سیگنال با مقادیر مختلط با نام پوش مختلط به صورت زیر تعریف می شود.

$$x_z(t) = \frac{1}{2}x_i(t) + \frac{j}{2}x_q(t) = \frac{x_a(t)}{2}e^{jx_p(t)} \quad (5)$$

سیگنال میان گذر اصلی را می توان به صورت زیر از روی پوش مختلط به دست آورد.

$$x(t) = \Re[2x_z(t)e^{j2\pi f_c t}] \quad (6)$$

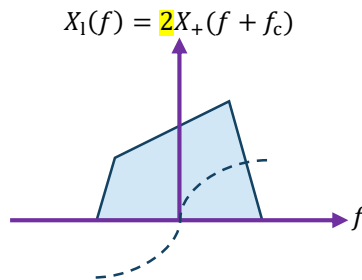
از آن جا که تابع نمایی مختلط تنها فرکانس مرکزی را تعیین می نماید، سیگنال مختلط $x_i(t)$ همه ی اطلاعات موجود در $x(t)$ را

در بر دارد. با استفاده از بیان باندپایه ی سیگنال های میان گذر، تحلیل های سامانه ی مخابراتی صورت ساده تری به خود می گیرد.

تبدیل بین سیگنال میان گذر و معادل پایین گذر

در ادامه روشی برای تبدیل بین سیگنال میان گذر و معادل پایین گذر یا سیگنال پوش مختلط آورده شده است. **سیگنال معادل پایین گذر** که با $x_1(t)$ نمایش داده می شود، دارای پاسخ فرکانسی به صورت زیر است که در شکل ۲ نشان داده شده است.

$$X_1(f) = 2X_+(f + f_c) \quad (7)$$

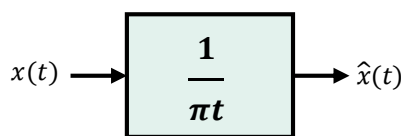


شکل ۲ پاسخ فرکانسی مربوط به معادل پایین گذر سیگنال شکل

می توان نشان داد که معادل پایین گذر یک سیگنال میان گذر را می توان با استفاده از رابطه ی زیر به دست آورد.

$$x_1(t) = (x(t) + j\hat{x}(t))e^{-j2\pi f_c t} \quad (8)$$

در رابطه ی فوق $\hat{x}(t)$ ، تبدیل هیلبرت سیگنال $x(t)$ است که در شکل ۳ نشان داده شده است.



شکل ۳ تبدیل هیلبرت سیگنال $x(t)$

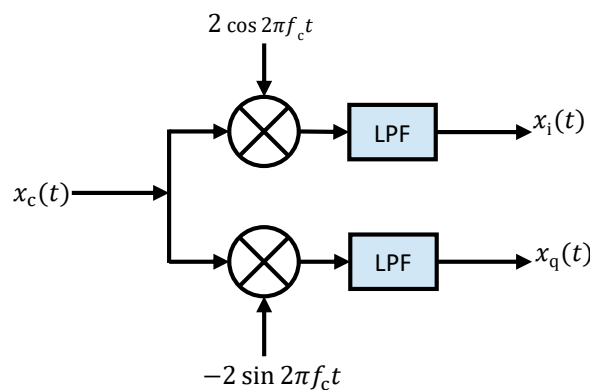
با توجه به رابطه ی (۸)، سیگنال $x_1(t)$ یک سیگنال مختلط خواهد بود. همان طور که اشاره شد، قسمت حقیقی و مختلط این سیگنال به ترتیب مولفه های هم فاز و غیرهم فاز^۴ نامیده شده و با $x_i(t)$ و $x_q(t)$ نشان داده می شود. بنابراین می توان نوشت.

$$x_1(t) = (x(t) + j\hat{x}(t))e^{-j2\pi f_c t} = x_i(t) + jx_q(t) \quad (9)$$

$$\begin{cases} x_i(t) = x(t) \cos(2\pi f_c t) + \hat{x}(t) \sin(2\pi f_c t) \\ x_q(t) = \hat{x}(t) \cos(2\pi f_c t) - x(t) \sin(2\pi f_c t) \end{cases} \quad (10)$$

از آنجا که فیلتر هیلبرت یک فیلتر غیرعلی است، در عمل برای به دست آوردن $x_i(t)$ و $x_q(t)$ نیازی به محاسبه ی تبدیل هیلبرت نیست و می توان از روشی مشابه شکل ۴ دسترسی چندگانه با تقسیم فرکانس

شکل ۵ استفاده می شود. فیلتر پایین گذر می بایست سیگنال پوش مختلط را عبور دهد و مولفه های دو برابر فرکانس مرکزی را حذف کند.



شکل ۴ دسترسی چندگانه با تقسیم فرکانس

شکل ۵ به دست آوردن معادل پایین گذر بدون استفاده از تبدیل هیلبرت

اگر معادل پایین گذر را داشته باشیم، با در نظر گرفتن رابطه ی (۹) می توان سیگنال میان گذر را به دست آورد. معادل میان گذر به صورت زیر به دست می آید.

$$\begin{cases} x(t) = \Re\{x_1(t)e^{j2\pi f_c t}\} = x_i(t) \cos 2\pi f_c t - x_q(t) \sin 2\pi f_c t \\ \hat{x}(t) = \Im\{x_1(t)e^{j2\pi f_c t}\} = x_q(t) \cos 2\pi f_c t + x_i(t) \sin 2\pi f_c t \end{cases} \quad (11)$$

چگالی طیف توان فرآیند تصادفی ایستان

فرآیند $x(t)$ ، ایستان به مفهوم وسیع گفته می شود اگر میانگین فرآیند $\mathbb{E}[x(t)] = m_x$ مستقل از زمان و تابع خودهمبستگی

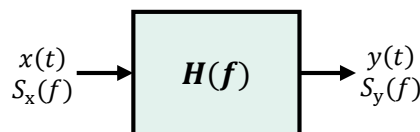
$$R(t + \tau, t) = \mathbb{E}[x(t)x(t + \tau)] = R_x(\tau) \quad \text{تنها تابعی از تأخیر } \tau \text{ باشد.}$$

برای فرآیند ایستان با تابع خودهمبستگی $R_x(\tau)$ ، چگالی طیف توان به صورت زیر محاسبه می شود.

$$S_x(f) = \mathcal{F}[R_x(\tau)] \quad (12)$$

یکی از موارد که به هنگام کار با سیستم های نامتغیر با زمان خطی مورد استفاده قرار می گیرد، طیف سیگنال خروجی از روی طیف سیگنال ورودی است. این رابطه در زیر آمده است.

$$S_y(f) = S_x(f)|H(f)|^2 \quad (13)$$



شکل ۶ عبور یک فرآیند از سیستم خطی نامتغیر با زمان

⁴ In-Phase

⁵ Quadrature

رادیونرم افزار ADALM-PLUTO

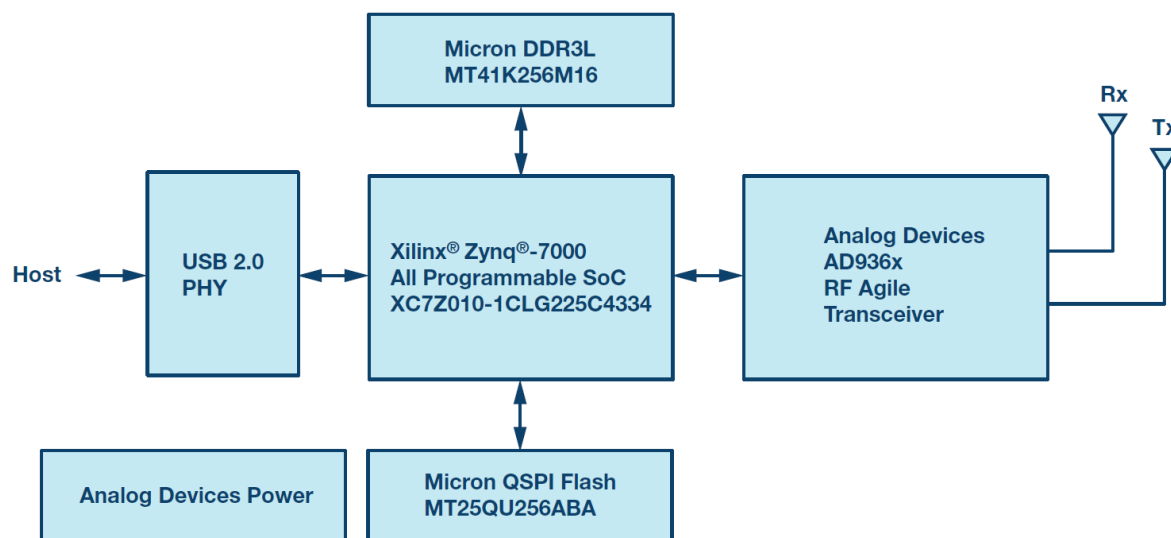
ماژول یادگیری فعال ADALM-PLUTO کمک شایانی در آشنایی دانشجویان با اصول رادیونرم افزار، فرکانس رادیویی و مخابرات بی سیم می نماید. این ماژول برای دانشجویان با هر سطح و زمینه ای طراحی شده است و می توان از آن هم زمان با رشد دانش علمی و مهندسی دانشجویان جهت افزایش درک آنان از دنیای واقعی فرکانس های رادیویی و مخابرات استفاده شود.

رادیونرم افزار PlutoSDR شبیه یک آزمایشگاه سیار عمل می کند و یادگیری کلاسی را تقویت می کند. MATLAB و Simulink دو بسته ی نرم افزاری از میان بسته های متعدد است که از PlutoSDR پشتیبانی می نماید و یک واسط کاربری گرافیکی را در اختیار دانشجویان قرار می دهد تا سرعت یادگیری افزایش یابد و کارها و پژوهش های با هوشمندی بیشتری صورت پذیرد.

رادیونرم افزار PlutoSDR دارای کانال های مستقل فرستندگی و گیرندگی است که می تواند به صورت دوطرفه عمل نمایند. این ماژول می تواند به سیگنال های آنالوگ با فرکانس های بازه ی 325MHz تا 3800MHz و با حداکثر نرخ 20 مگامونه در ثانیه (MSPS) را تولید و دریافت نماید. PlutoSDR به هیچ سخت افزار جانبی نیاز ندارد و با میان افزار پیش فرض خود به صورت کامل با USB عمل می نماید و با توجه قابلیت درایورهای libiio این سخت افزار، می تواند از سیستم عامل های مختلف پشتیبانی نماید. قابلیت های این رادیونرم افزار در زیر آمده است.

- Portable self-contained RF learning module
- Cost-effective experimentation platform
- Based on Analog Devices AD9363--Highly Integrated RF Agile Transceiver and Xilinx® Zynq Z-7010 FPGA
- RF coverage from 325 MHz to 3.8 GHz
- Up to 20 MHz of instantaneous bandwidth (Complex I/Q)
- Flexible rate, 12-bit ADC and DAC
- One transmitter and one receiver, half or full duplex
- MATLAB®, Simulink® support
- GNU Radio sink and source blocks
- libiio, a C, C++, C#, and Python API
- USB 2.0 Powered Interface with Micro-USB 2.0 connector
- High quality plastic enclosure

در شکل ۷ بلوک دیاگرام ساده ی این رادیونرم افزار رسم شده است.



شکل ۷ بلوک دیاگرام ساده شده ی ADALM-PLUTO

همچنین در ادامه مشخصات فنی این رادیونرم افزار نیز آمده است.

جدول 1 مشخصه های فنی رادیونرم افزار ADLAM-PLUTO

| مقدار | مشخصات |
|--------------------------------------|--------------------------------|
| مصرف توان | |
| ولتاژ DC ورودی (USB) | ۴/۵ تا ۵/۵ ولت |
| عملکرد واگردان و سیگنال ساعت | |
| نرخ نمونه برداری DAC و ADC | 61.44MSPS تا 65.2kSPS |
| رزولوشن DAC و ADC | ۱۲ بیت |
| صحت فرکانسی | ±25ppm |
| عملکرد RF | |
| گستره ی تنظیم فرکانس | 3800MHz تا 325MHz |
| توان خروجی فرستنده | 7dBm |
| عدد نویز گیرنده | < 3.5dB |
| صحت مدولاسیون (EVM) گیرنده و فرستنده | -34dB (2%) |
| شیلد RF | وجود ندارد |
| دیجیتال | |
| USB | 2.0 On-the-Go |
| هسته ی پردازشی | Single Arm Cortex®-A9 @ 667MHz |
| تعداد سلول های منطقی FPGA | 28k |
| قسمت های DSP | 80 |
| DDR3L | 4Gb (512MB) |
| QSPI Flash | 256Mb (32MB) |
| مشخصات فیزیکی | |
| ابعاد | 117mm×79mm×24mm |
| وزن | 114g |
| دما | 40°C تا 10°C |

پیگر بندی رادیو نرم افزار

قبل از ارسال و دریافت سیگنال های رادیویی با استفاده از رادیو نرم افزار ADALM-PLUTO می بایست پارامترهای سخت افزار رادیو و مشخصه های تنظیم فرکانس رادیو را اعمال نمود. توابعی که در ارتباط با این رادیو نرم افزار می باشد به شرح زیر است.

جدول 2 توابع رادیونرم افزار ADLAM-PLUTO

| تابع | عملکرد |
|----------------------------------|--|
| <code>configurePlutoRadio</code> | تنظیم میان افزار رادیو نرم افزار ADALM-PLUTO |
| <code>findPlutoRadio</code> | گزارش اطلاعاتی در مورد رادیو نرم افزارهای متصل شده |
| <code>sdrdev</code> | تولید شیء رادیو برای یک سخت افزار رادیویی مشخص |
| <code>sdrxx</code> | ایجاد شیء سیستم گیرنده برای سخت افزار رادیو |
| <code>sdrtx</code> | ایجاد شیء سیستم فرستنده برای سخت افزار رادیو |
| <code>designCustomFilter</code> | طراحی فیلتر دلخواه برای چیپ های AD936x |
| <code>info</code> | کسب اطلاعات در مورد رادیو |

کلاس نرم افزاری این رادیو نرم افزار `comm.SDRDevPluto` می باشد که شیء هایی را برای رادیو نرم افزار ADALM-PLUTO شرکت Analog Device تولید می نماید.

آشنایی با طیف های فرکانسی

انواع روش های دسترسی چندگانه

دسترسی چندگانه با تقسیم فرکانس (FDMA)

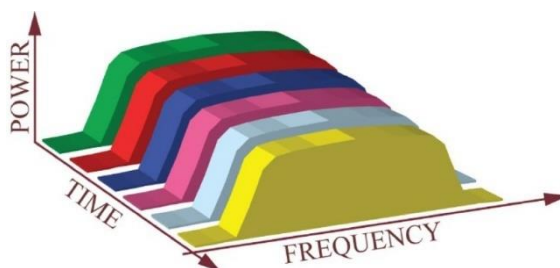
در دسترسی چندگانه با تقسیم فرکانس به هر کاربر یک کانال فرکانسی مجزا اختصاص داده می شود. به منظور عدم تداخل بین باندهای مجاور یک گارد فرکانسی بین کانال ها قرار می گیرد. معمولاً یک باند فرکانسی برای دریافت از ایستگاه پایه و یک باند فرکانسی برای ارسال به ایستگاه پایه مورد استفاده قرار می گیرد. شمای کلی این دسترسی چندگانه در شکل ۸ نشان داده شده است.



شکل ۸ دسترسی چندگانه با تقسیم فرکانس

دسترسی چندگانه با تقسیم زمان (TDMA)

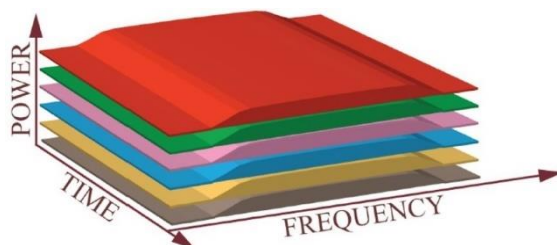
در این دسترسی چندگانه، زمان به تکه هایی تقسیم می شود و در هر تکه تنها یک موبایل داده های خود را ارسال می نماید. به عبارتی به هر کاربر یک تکه زمان داده شده است. شمای کلی این دسترسی چندگانه در شکل ۹ نشان داده شده است.



شکل ۹ دسترسی چندگانه با تقسیم زمان

دسترسی چندگانه با تقسیم کد (CDMA)

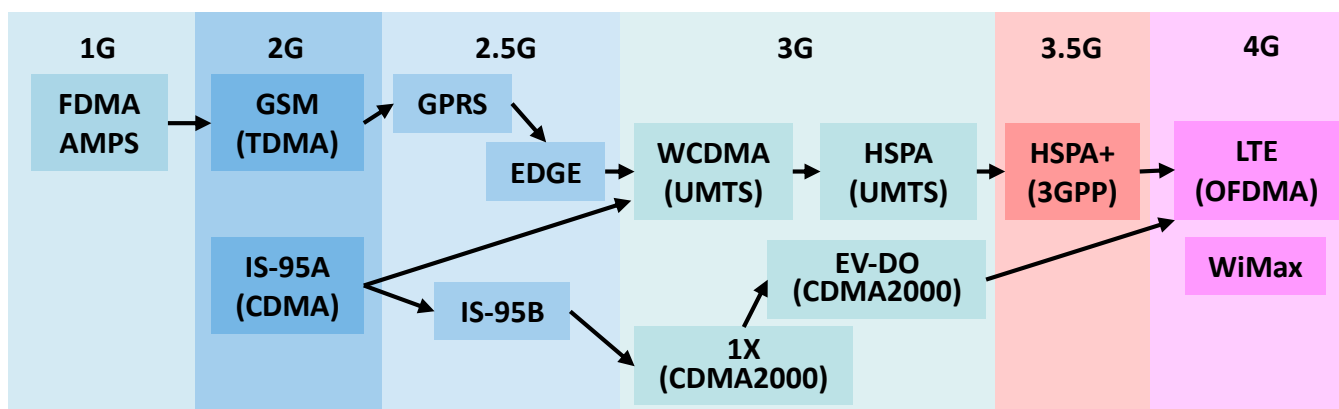
در این دسترسی چندگانه، با استفاده از کدهای متعامد برای جدا کردن ارسال و دریافت های مختلف استفاده می شود. هر بیت داده با تعداد بیشتری بیت ارسال می شود و هر کاربر دارای کد اختصاصی خود است. پهنای باندی که توسط کاربر اشغال می شود از پهنای باند مورد نیاز اطلاعات بیشتر است. همه ی کاربرها باند فرکانسی یکسانی را انتخاب می نمایند. شمای کلی این دسترسی چندگانه در شکل ۱۰ نشان داده شده است.



شکل ۱۰ دسترسی چندگانه با تقسیم کد

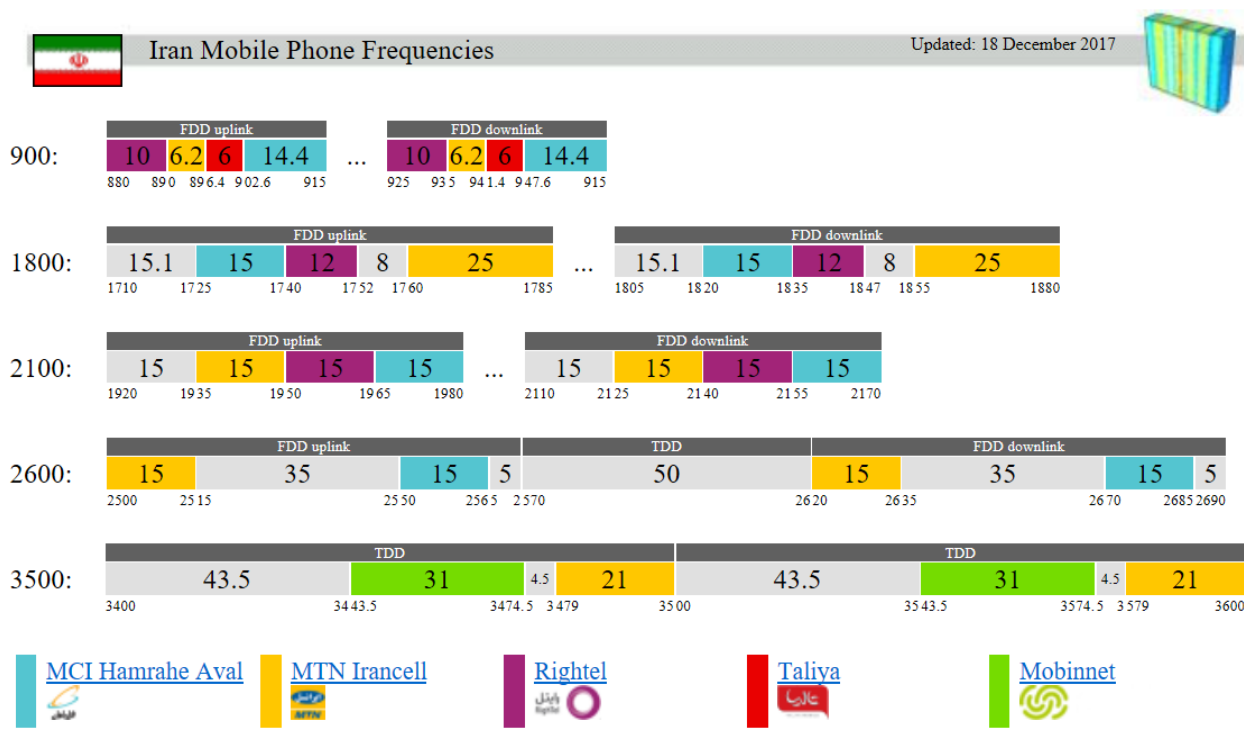
طیف سیگنال نسل های مختلف شبکه ی سلولی

در این جا به صورت مختصر روش های دسترسی چندگانه که در نسل های مختلف شبکه ی سلولی استفاده می شود، ارایه می گردد. برای این منظور به شمای کلی شکل ۱۱ توجه نمایید.



شکل ۱۱ روش های دسترسی چندگانه در نسل های مختلف تلفن همراه

در شکل زیر گستره ی فرکانسی مربوط به نسل های مختلف در شبکه ی سلولی ایران مربوط به اپراتورهای مختلف نشان داده شده است.



شکل ۱۲ فرکانس های اختصاصی یافته در شبکه ی سلولی ایران

شرح آزمایش

آزمایش ۱-۳: معادل پایین گذر و میان گذر سیگنال

۱. تولید فرآیند میان گذر و پایین گذر: در صورتی که یک فرآیند تمام گذر از یک فیلتر پایین گذر عبور کند، یک فرآیند پایین گذر حاصل می شود. در این جا می خواهیم یک فرآیند پایین گذر را به معادل میان گذر تبدیل نماییم و مجدداً آن را به معادل پایین گذر تبدیل کنیم. کلیه ی طیف ها می بایست بر حسب dBm باشد و برچسب گذاری محور فرکانس به درستی انجام و بر حسب MHz بیان شود.

ا. تولید فرآیند پایین گذر: دنباله ی تصادفی مختلط با ۴۰۹۶ نمونه ی مستقل و با توزیع گاوسی مختلط با میانگین صفر و واریانس ۱ تولید نماییم. با استفاده از ابزار طراحی فیلتر نرم افزار MATLAB (دستور **filterDesigner**) در پنجره ی دستور وارد نماییم و ضرایب فیلتر را به workspace ارسال نماییم (یک فیلتر پایین گذر FIR از نوع Equiripple و با پارامترهای فیلتری $f_s = 28.8\text{MHz}$, $f_{\text{pass}} = 1.4\text{MHz}$, $f_{\text{stop}} = 2\text{MHz}$, $A_{\text{pass}} = 0.5\text{dB}$ و $A_{\text{stop}} = 70\text{dB}$ طراحی نماییم. پس از به دست آوردن ضرایب فیلتر با استفاده از عمل کانولوشن خروجی این فیلتر را برای ورودی تولید شده، به دست آورید. طیف ۴۰۹۶ نقطه ای فرآیند تمام گذر ورودی و فرآیند پایین گذر خروجی را به دست آورید و در یک شکل با دو زیر نمودار رسم نماییم. توان فرآیند را با استفاده از نمونه های زمانی قبل و بعد فیلتر شدن به دست آورده و ارتباط آن را با پهنای باند فیلتر به دست آورید. (راهنمایی: خروجی عمل کانولوشن تعداد نقاط بیشتری دارد و می بایست این تعداد نقاط را برابر با ۴۰۹۶ نقطه نمود. برای به دست آوردن پهنای باند فیلتر از دستور **noisebw** نرم افزار MATLAB استفاده نماید).

ب. تولید فرآیند میان گذر معادل یک فرآیند پایین گذر: حال با استفاده از مباحثی که در بخش تئوری مطرح شد، فرآیند پایین گذر تولید شده را به یک فرآیند میان گذر بر روی فرکانس 3.57MHz منتقل نماییم. توان هر سه فرآیند را مشابه قبل به دست آورید. آیا توان فرآیند معادل پایین گذر با میان گذر یکی است؟ علت را شرح دهید. حال طیف ۴۰۹۶ نقطه ای هر سه فرآیند را در یک شکل با سه زیر نمودار رسم نماییم.

ت. تولید فرآیند پایین گذر از یک سیگنال میان گذر: فرآیند تولید شده در بخش ب را دوباره به معادل پایین گذر خود تبدیل نموده و خروجی را از سه فیلتر پایین گذر مختلف با پهنای باند 0.7MHz، 1.4MHz و 2.8MHz عبور دهید. طیف فرآیند خروجی این سه فیلتر را با طیف بخش ا مقایسه نماییم و در هر مورد توان خروجی زمانی هر فیلتر را به دست آورید و علت تفاوت ها را شرح دهید. فیلترها را با استفاده از دستور **fir1** تولید نموده و مرتبه ی فیلتر را برابر با ۱۲۰ در نظر بگیرید. (فرکانس نمونه برداری همان $f_s = 28.8\text{MHz}$ می باشد).

آزمایش ۲-۳: کار با رادیونرم افزار ADALM-PLUTO

۱. اطمینان از نرم افزار بسته ی پشتیبانی سخت افزاری ADALM-PLUTO: درون نرم افزار MATLAB دستور `sdrdev('Pluto')` را وارد نمایید و اطمینان حاصل نمایید که این دستور برای نرم افزار MATLAB شناخته شده باشد و هیچ پیغام خطایی ظاهر نمی شود.

۲. اطمینان از سخت افزار بسته ی پشتیبانی سخت افزاری ADALM-PLUTO:
 ا. اتصال ADALM-PLUTO: رادیو نرم افزار ADALM-PLUTO خود را درون یکی از درگاه های USB2.0 یا USB3.0 وارد نمایید. در این مرحله اتصال آنتن لازم نیست ولی اتصال آن نیز بلامانع است.
 ب. بررسی شناخته شدن سخت افزار ADALM-PLUTO: عبارت `my_pluto = findPlutoRadio` را درون پنجره ی دستور MATLAB وارد نماید. اگر سخت افزار شناخته شود عبارت زیر ظاهر می شود.

```
>> my_pluto = sdrinfo
my_pluto =
    struct with fields:
        RadioID: 'usb:0'
        SerialNum: '100000235523730700230031090216eae'
```

اگر سخت افزار قابل شناسایی نباشد و یا به رایانه متصل نباشد یک خروجی ساختار تهی دریافت خواهید نمود. می توانید سخت افزار خود را قطع نمایید تا خروجی زیر را دریافت کنید.

```
>> my_pluto = sdrinfo
my_pluto
0x1 empty struct array with fields:
```

```
RadioID
SerialNum
```

۳. پارامترها و راه اندازی اولیه ی رادیونرم افزار ADALM-PLUTO: به منظور راه اندازی رادیونرم افزار ADALM-PLUTO نیاز به آشنایی با پارامترها و شیوه ی تعریف شیء های رادیونرم افزار است. در ادامه کلیت کدی برای راه اندازی اولیه ی این رادیونرم افزار آمده است. این کد را وارد نمایید و از اجرای آن و آشنایی با کلیه ی پارامترها مطمئن شوید.

```
%% Parameters
fs = 10e6; % Baseband Sampling Rate (65105 to 61.44e6 Hz)
frame_size = 4096; % Samples per Each Frame (< 2^20)
% Transmitter Parameters
tx_fc = 325e6; % Set Transmitter Center Frequency
% (AD9363: 325-3800MHz) (AD9364: 70-6000MHz)
tx_gain = -30; % Set Transmitter Attenuation as a Negative Gain
% (-89.75 to 0 dB)
tx_address = 'usb:0'; % Set Transmitter Identification Number
% Receiver Parameters
rx_fc = 325e6; % Set Receiver Center Frequency
% (AD9363: 325-3800MHz) (AD9364: 70-6000MHz)
rx_gain = 20; % Set Receiver Gain (-4dB to 71dB)
rx_address = 'usb:0'; % Set Receiver Identification Number
% Initialize ADALM-PLUTO
dev = sdrdev('Pluto'); % Create Radio Object for ADALM-PLUTO
setupSession(dev)
configurePlutoRadio('AD9363'); % Configure ADALM-PLUTO Radio Firmware
% Define Transmitter Object
tx = sdrtx('Pluto','RadioID',tx_address); % CreateTransmitterSystem Object
tx.CenterFrequency = tx_fc; % Set Transmitter Center Frequency
tx.Gain = tx_gain; % Set Transmitter Gain
tx.BasebandSampleRate = fs; % Set Baseband Sampling Rate
% Define Receiver Object
rx = sdrxx('Pluto','RadioID',rx_address); % Create Receiver System Object
rx.CenterFrequency = rx_fc; % Set Receiver Center Frequency
rx.BasebandSampleRate = fs; % Set Baseband Sampling Rate
rx.SamplesPerFrame = frame_size; % Samples per Each Frame (< 2^20)
rx.GainSource = 'Manual'; % AGC Settings
rx.Gain = rx_gain; % Receiver Gain
rx.OutputDataType = 'double'; % Output Data Type
```

۴. ارسال متوالی سیگنال با فرستنده ی ADALM-PLUTO: زمانی که رادیونرم افزار ADALM-PLUTO روشن شد، حتی اگر درخواستی ارسال نشده باشد، فرستنده شروع به ارسال داده می کند و آخرین محتوای بافر خود را ارسال می نماید. این وضعیت تا زمانی که رادیونرم افزار روشن است، ادامه دارد. حال اگر بنا باشد یک داده به صورت متوالی ارسال شود، از دستور **transmitRepeat** استفاده می شود. در این جا می خواهیم یک سیگنال سینوسی به فاصله ی 1MHz نسبت به فرکانس مرکزی 325MHz تولید نماییم.

گام ۱. مانند کد زیر یک سیگنال سینوسی ارسال نمایید.

```
% Transmit Repeat
f_offset = 1e6;
t = (0:2^14-1)/fs;
sin_wave = exp(1j*2*pi*f_offset*t);
tx.transmitRepeat(sin_wave);
```

۵. دریافت سیگنال با استفاده از گیرنده **ADALM-PLUTO**: در این جا بناست پهنای باندی برابر 10MHz حول فرکانسی مرکزی 325MHz را دریافت نموده و طیف توان و نمونه های زمانی معادل باندپایه ی سیگنال نمایش داده شود. برای این بخش خروجی فرستنده را با کابل به ورودی گیرنده وصل نمایید.

گام ۱. اطمینان حاصل شود که پارامترهای رادیونرم افزار به نحوی تنظیم شده باشد که فرکانس مرکزی آن 325MHz، بهره ی گیرنده برابر 20dB، نرخ نمونه برداری برابر 10MHz، طول داده های هر فریم برابر ۴۰۹۶ نقطه، نوع داده های خروجی به صورت double و زمان توقف دریافت داده برابر ۶۰ ثانیه باشد.

گام ۲. حلقه ی تکرار برنامه ی خود را با استفاده از دستورهای **tic** و **toc** و بر اساس متغیر زمان توقف دریافت داده (**stop_time**) پیاده سازی نمایید.

گام ۳. با استفاده از فراخوانی شیء گیرنده با وارد کردن **rx()** یک فریم از **ADALM-PLUTO** را دریافت نمایید و در متغیر **pluto_data** ذخیره نمایید.

گام ۴. FFT داده های به دست آمده از رادیونرم افزار را محاسبه و طیف توان آن را بر حسب dBm به دست آورید و در متغیر **pluto_data_fft** قرار دهید.

گام ۵. بر اساس متغیر **pluto_data** و **pluto_data_fft** دو نمودار زیر هم رسم نمایید. نمودار بالا مقدار حقیقی یا موهومی داده های زمانی و نمودار پایین طیف توان لحظه ای و بیشینه ی طیف توان لحظه ای در کل بازه ی زمان دریافت (Max Hold) را نمایش می دهد که با دریافت داده ی جدید به روز می شود. (راهنمایی: برای این کار از دستورهای **refreshdata** و **drawnow** استفاده نمایید. بخش **Automatically Refresh Plot After Changing Data** در راهنمای MATLAB را مطالعه نمایید.)

آزمایش ۳-۳: پایش طیف فرکانسی با استفاده از رادیونرم افزار

۱. جاروب فرکانسی رادیو نرم افزار **ADALM-PLUTO**: برنامه ای بنویسید که کل گستره ی فرکانسی 75MHz تا 6GHz را با رادیونرم افزار **ADALM-PLUTO** جاروب نماید و طیف توان کل این بازه را در قالب یک نمودار ارایه دهد.

گام ۱. پارامترهای رادیونرم افزار باید در یک حلقه به نحوی تنظیم شود که از فرکانس مرکزی 100MHz تا 6GHz با گام های 50MHz جابه جا شود. بهره ی گیرنده برابر 30dB، نرخ نمونه برداری برابر 50MHz، طول داده های هر فریم برابر ۴۰۹۶ نقطه، نوع داده های خروجی به صورت double باشد.

گام ۲. در هر گام ۵ فریم را قرائت کرده و نادیده بگیرید. قدرمطلق FFT مربوط به ۱۰ فریم بعدی را محاسبه کرده و میانگین این ۱۰ فریم را محاسبه کنید. تعداد نمونه های طیف میانگین را با ضریب ۱۶ کاهش دهید. (راهنمایی: قبل از FFT می بایست مقدار DC هر فریم دریافتی را حذف نمود. برای کاهش تعداد نمونه از دستور **decimate** استفاده نمایید.)

گام ۳. طیف میانگین با نرخ کاهش یافته را به یک متغیر اضافه نمایید. طیف حاصل را به dBm تبدیل کرده و نمایش دهید.

۲. جستجوی ایستگاه های رادیو FM با رادیونرم افزار **ADALM-PLUTO**: سیگنال های رادیو FM در محدوده ی فرکانسی 88MHz تا 108MHz قرار دارند. سیگنال های حامل ایستگاه ها به فاصله ی 200kHz از یک دیگر قرار گرفته اند. با استفاده از برنامه های قبل انجام دهید.

گام ۱. به کمک **ADALM-PLUTO** بیان کنید در محدوده ی فرکانسی ذکر شده چه تعداد ایستگاه رادیو FM وجود دارد؟

گام ۲. فرکانس ایستگاه های قوی رادیو FM را ثبت نمایید.

گام ۳. قوی ترین سیگنال رادیو FM را پیدا نمایید و تنها این سیگنال را با استفاده از رادیونرم افزار دریافت نمایید و رفتار زمانی بخش حقیقی و موهومی معادل باندپایه ی آن را تحلیل نمایید.

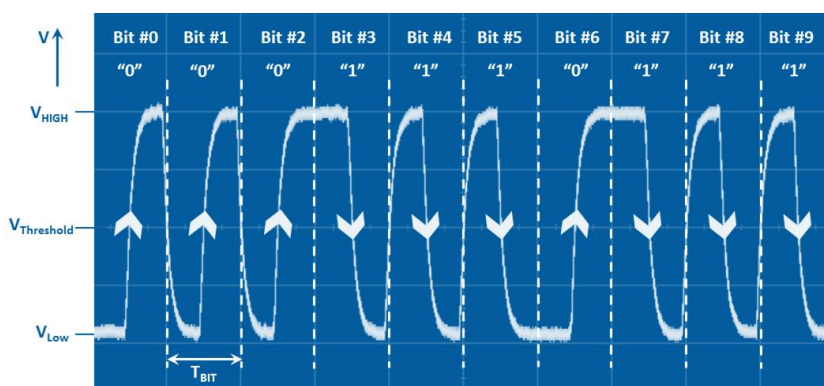
گام ۴. با استفاده از کدی که در اختیار شما قرار می گیرد، به این ایستگاه رادیویی گوش دهید.

۳. پایش طیف تلفن همراه - نسل ۲ با ADALM-PLUTO: در استاندارد GSM-900 مربوط به نسل دو پیاده سازی شده در ایران گستره‌ی فرکانسی 890MHz تا 915MHz مربوط به ارسال به ایستگاه پایه^۶ و گستره‌ی فرکانسی 935MHz تا 960MHz مربوط به دریافت از ایستگاه پایه است. در استاندارد GSM-1800 گستره‌ی فرکانسی 1710MHz تا 1785MHz برای ارسال به ایستگاه پایه و گستره‌ی فرکانسی 1805MHz تا 1880MHz برای دریافت از ایستگاه پایه در نظر گرفته شده است. در این محدوده‌های فرکانسی به دنبال سیگنال تلفن همراه نسل دو بگردید.

- گام ۱. کانال های موبایل نسل دوم دارای چه پهنای کانالی هستند و در چه فاصله‌ای از یکدیگر قرار گرفته‌اند؟
- گام ۲. تلفن همراه خود را برای ارتباط با نسل دو تنظیم نمایید و با گرفتن تماس تلفنی و ارسال پیام سعی کنید سیگنال تلفن همراه خود را مشاهده نمایید.
- گام ۳. رفتار طیف نسل دوم را با توجه به نمودار آبخاری تحلیل نمایید. این طیف با کدام طیف دسترسی چندگانه تطبیق دارد.

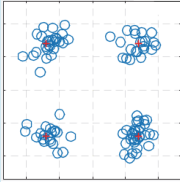
۴. پایش طیف ریموت خودرو: رادیو نرم افزار ADALM-PLUTO را در فرکانس 433.9MHz تنظیم نمایید و ریموت خودرو را فشار دهید.

- گام ۱. فرکانس نمونه برداری را برابر با 250kHz قرار داده و طول بسته‌ها را برابر 65536 در نظر بگیرید. رفتار طیف را تحلیل نمایید.
- گام ۲. با تنظیم دستی فرکانس مرکزی ADALM-PLUTO سعی کنید سیگنال را به باند پایه انتقال دهید و نمونه‌های زمانی آن را رسم کنید.
- گام ۳. با انتخاب یک سطح آستانه، اگر سیگنال از این آستانه بالاتر بود نمایش طیف و نمونه‌های زمانی متوقف نمایید و از نمونه‌های آن برای گام آینده استفاده کنید.
- گام ۴. با استفاده از کدگذاری منچستر که در زیر آمده است، کد ریموت خودرو را استخراج نمایید.



- [1] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.
- [2] T. F. Collins, R. Getz, D. Pu, and A. M. Wyglinski, *Software-defined radio for engineers*. Norwood, MA: Artech House, 2018.

⁶ base station



آزمایش چهارم

معرفی مدولاسیون دیجیتال خطی

اهداف آزمایش

سامانه‌های مخابرات دیجیتال بی‌سیم اطراف ما را فرا گرفته است و ارتباطات صدا و داده در سامانه‌های مخابرات سلولی، WiFi، Bluetooth و سایر استانداردها را فراهم می‌کند. در طی چندین آزمایش آینده بلوک‌های پایه‌ی مخابرات دیجیتال شامل الگوهای مدولاسیون، دریافت سمبل و بیت، عملکرد در حضور نویز، کدگذاری و کدگشایی تقاضایی، تبدیل فرکانس به بالا و پایین دیجیتال و ... ارائه می‌شود. در ادامه جزئیات طراحی گیرنده‌ی دیجیتال شامل همگام‌سازی فرکانس حامل و همگام‌سازی زمانی سمبل‌ها مورد بررسی قرار می‌گیرد. در این آزمایش با الگوهای مدولاسیون دیجیتال خطی و شکل‌دهی پالس آشنا می‌شویم.

در انتهای این آزمایش دانشجو می‌بایست:

- منظومه‌های مدولاسیون‌های خطی M-PAM، M-PSK و M-QAM را بشناسد.
- بتواند رشته‌های بیت را به سمبل‌های مدولاسیون‌های مختلف در حالت سمبل‌های مختلط و سمبل‌های I/Q تبدیل نماید.
- با مفهوم کدگذاری Gray آشنا شود.
- بتواند سمبل‌های دریافتی را به رشته‌های بیت تبدیل نماید.
- بتواند سیگنال را در یک کانال باندپایه ارسال نماید و اثر نویز را در تبدیل سمبل‌ها به رشته‌های بیت را بشناسد.
- با طراحی Correlator و فیلتر منطبق آشنا شود.
- بتواند گیرنده‌ی حداقل فاصله را طراحی نماید.

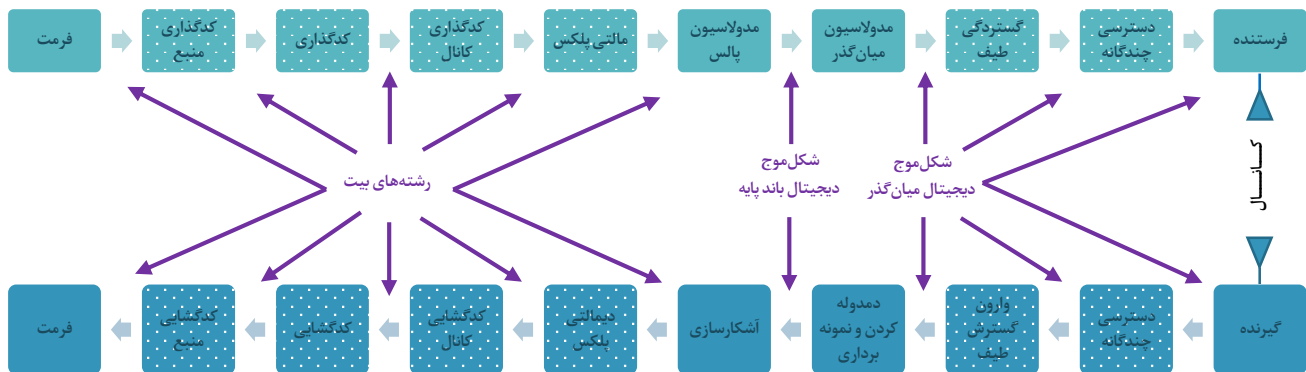
ابزارهای مورد نیاز

رایانه

نرم‌افزار MATLAB R2020b



شکل 1 شمای یک سامانه‌ی مخابرات دیجیتال متداول را به صورت کامل نشان می‌دهد. در این آزمایش و آزمایش بعد سعی داریم که بخش‌های ضروری این سامانه را برای مدولاسیون‌های خطی M-PAM، M-PSK و M-QAM مدل‌سازی کنیم. در ادامه این مدولاسیون‌ها را مرور خواهیم کرد.



شکل 1 شمای یک سامانه‌ی مخابرات دیجیتال (بخش‌های با پس‌زمینه‌ی نقطه‌دار اختیاری هستند)

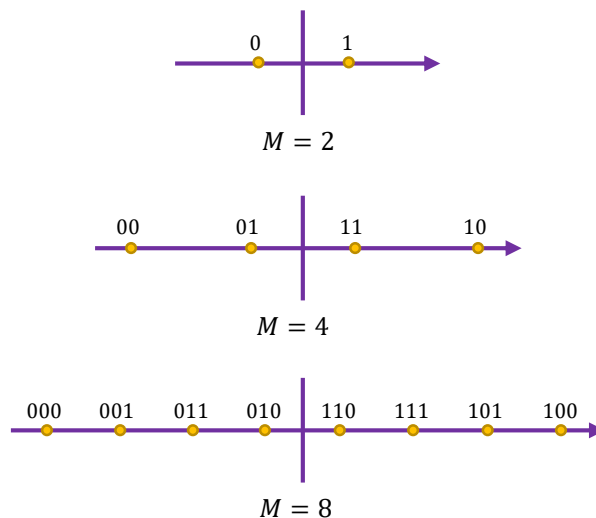
مدولاسیون‌های خطی

مدولاسیون M-PAM (Pulse Amplitude Modulation)

این مدولاسیون از جمله مدولاسیون‌های دیجیتال باند پایه می‌باشد. در این مدولاسیون اطلاعات در توان سیگنال ارسالی ذخیره می‌شود و آشکارسازی آن با تشخیص سطح سیگنال انجام می‌شود.

در این مدولاسیون سیگنال ارسالی به صورت $s_m(t) = A_m p(t)$ است. پالس ارسالی $p(t)$ می‌تواند به طور دلخواه انتخاب شود، اما از آنجا که پهنای باند سیگنال ارسالی به طور کامل به شکل پالس بستگی دارد، تعیین پالس مناسب اهمیت زیادی خواهد داشت. A_m دامنه پالس ارسالی را با توجه به پیغام تنظیم می‌کند. با فرض اینکه هر سیگنال ارسالی حاوی k بیت پیام باشد، می‌توان A_m ‌ها را از مجموعه زیر انتخاب نمود.

$$A_m \in \{\pm 1, \pm 3, \dots, \pm(M-1)\}; \quad M = 2^k$$



شکل 2 منظومه‌ی سیگنالی مربوط به مدولاسیون M-PAM برای مقادیر مختلف M

مدولاسیون PSK (Phase Shift Keying Modulation)

در این مدولاسیون، سیگنال ارسالی به صورت $s_m(t) = g(t) \cos\left(2\pi f_0 t + \frac{2\pi}{M}(m-1)\right)$ می باشد که f_0 فرکانس حامل خواهد بود. پالس $g(t)$ می تواند به طور دلخواه انتخاب شود. در اینجا نیز همانند مدولاسیون PAM می توان با انتخاب درست این پالس، در پهنای باند صرفه جویی نمود. با فرض اینکه هر سیگنال ارسالی حاوی k بیت پیغام باشد، می توان m را از مجموعه $\{1, 2, \dots, M\}$ انتخاب کرد. در این جا $M = 2^k$ می باشد. با در نظر گرفتن سیگنال ارسالی به صورت فوق، خواهیم داشت.

با انتخاب $\Phi_1(t) = \sqrt{\frac{2}{E_g}} g(t) \cos(2\pi f_0 t)$ و $\Phi_2(t) = -\sqrt{\frac{2}{E_g}} g(t) \sin(2\pi f_0 t)$ به عنوان پایه های متعامد فضای سیگنال، خواهیم داشت.

$$s_m(t) = \Phi_1(t) \cos\left(\frac{2\pi}{M}(m-1)\right) \sqrt{\frac{E_g}{2}} + \Phi_2(t) \sin\left(\frac{2\pi}{M}(m-1)\right) \sqrt{\frac{E_g}{2}}$$

در رابطه ی فوق E_g انرژی پالس $g(t)$ می باشد $(\int |g(t)|^2 dt = E_g)$. بدین ترتیب می توان معادل برداری سیگنال $s_m(t)$ را به صورت زیر در نظر گرفت.

$$\bar{s}_m = \sqrt{\frac{E_g}{2}} \begin{bmatrix} \cos\left(\frac{2\pi}{M}(m-1)\right) \\ \sin\left(\frac{2\pi}{M}(m-1)\right) \end{bmatrix}$$

با توجه به \bar{s}_m ، فضای سیگنال این مدولاسیون دارای دو بعد در باند میانی می باشد. معادل باند پایه ی این سیگنال به صورت زیر خواهد بود:

$$s_{l,m}(t) = g(t) e^{j\frac{2\pi}{M}(m-1)}$$

با در نظر گرفتن $\Phi_1(t) = \sqrt{\frac{2}{E_g}} g(t)$ به عنوان پایه متعامد فضای سیگنال، داریم:

$$s_{l,m}(t) = \sqrt{\frac{E_g}{2}} \Phi_1(t) e^{j\frac{2\pi}{M}(m-1)}$$

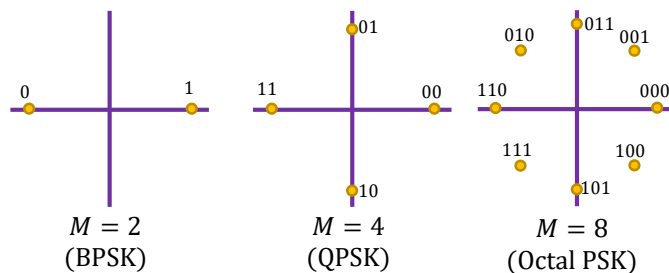
بنابراین در باند پایه، فضای سیگنال تنها یک بعد دارد. به بیان دیگر فضای سیگنال در این مدولاسیون دارای دو بعد حقیقی (باند میانی) یا یک بعد مختلط (باند پایه) خواهد بود. در باند پایه خواهیم داشت.

$$s_{l,m}(t) = g(t) \left(\cos\left(\frac{2\pi}{M}(m-1)\right) + j \sin\left(\frac{2\pi}{M}(m-1)\right) \right)$$

با متحد قرار دادن عبارت فوق با $s_{l,m}(t) = s_i(t) + js_q(t)$ ، مؤلفه های هم فاز و غیرهم فاز به دست خواهند آمد.

$$\begin{cases} s_i(t) = g(t) \cos\left(\frac{2\pi}{M}(m-1)\right) \\ s_q(t) = g(t) \sin\left(\frac{2\pi}{M}(m-1)\right) \end{cases}$$

همان گونه که ملاحظه می شود، در این مدولاسیون، فاز سیگنال ارسالی، مشخص کننده ی پیغام می باشد.



شکل 3 منظومه ی مربوط به مدولاسیون MPSK برای مقادیر مختلف M

مدولاسیون QAM (Quadrature Amplitude Modulation)

در این مدولاسیون، ابتدا سیگنال ارسالی را در باند پایه بررسی خواهیم کرد. سیگنال ارسالی مدولاسیون QAM در باند پایه به صورت زیر خواهد بود.

$$S_{l,m}(t) = (A_{m_i} + jA_{m_q})g(t)$$

$g(t)$ پالسی است که برای مدولاسیون انتخاب می شود و همانند دیگر مدولاسیون ها، بر پهنای باند مصرفی تأثیر می گذارد. بعد فضای این سیگنال در باند پایه ۱ می باشد. با در نظر گرفتن $s_{l,m}(t) = s_i(t) + js_q(t)$ ، مؤلفه های in-phase و quadrature به صورت زیر خواهند بود.

$$\begin{cases} s_i(t) = g(t)A_{m_i} \\ s_q(t) = g(t)A_{m_q} \end{cases}$$

معادل باند میانی این سیگنال به صورت زیر به دست می آید.

$$S_m(t) = \text{Re}\{S_{l,m}(t)e^{j2\pi f_0 t}\} = A_{m_i}g(t)\cos(2\pi f_0 t) - A_{m_q}g(t)\sin(2\pi f_0 t)$$

با انتخاب $\Phi_1(t) = \sqrt{\frac{2}{E_g}}g(t)\cos(2\pi f_0 t)$ و $\Phi_2(t) = -\sqrt{\frac{2}{E_g}}g(t)\sin(2\pi f_0 t)$ به عنوان پایه های متعامد فضای سیگنال، خواهیم داشت:

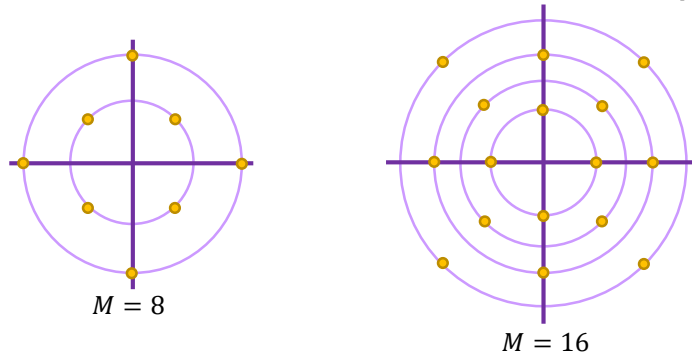
$$S_m(t) = A_{m_i}\sqrt{\frac{E_g}{2}}\Phi_1(t) + A_{m_q}\sqrt{\frac{E_g}{2}}\Phi_2(t)$$

در رابطه ی فوق نیز E_g انرژی پالس $g(t)$ می باشد $(\int |g(t)|^2 dt = E_g)$.

بدین ترتیب می توان معادل برداری سیگنال $s_m(t)$ را به صورت زیر در نظر گرفت:

$$\bar{s}_m = \sqrt{\frac{E_g}{2}} \begin{bmatrix} A_{m_i} \\ A_{m_q} \end{bmatrix}$$

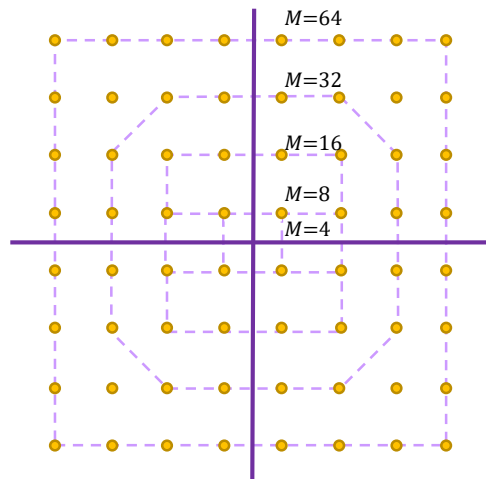
به ازای مقادیر مختلف A_{m_i} و A_{m_q} ، فضای سیگنال می تواند شکل های گوناگونی مانند شکل 4 به خود بگیرد.



شکل 4 منظومه ی مدولاسیون M-QAM برای مقادیر مختلف M

حالت خاص فضای سیگنال به ازای $A_{m_i}, A_{m_q} \in \{\pm 1, \pm 3, \dots, \pm(\sqrt{M}-1)\}$ تشکیل می شود که به QAM مستطیلی معروف است. منظومه ی این مدولاسیون در شکل 5 نشان داده شده است.

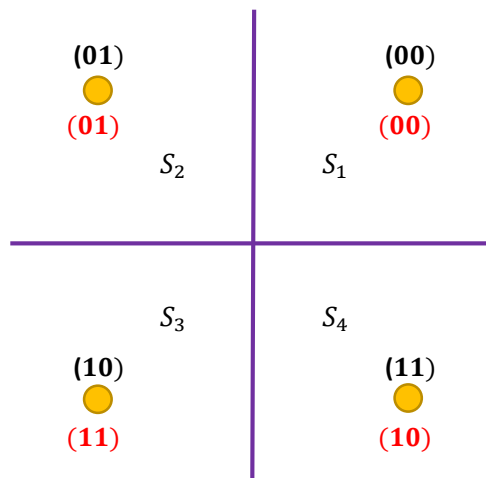
از این به بعد، هر جا که درباره مدولاسیون QAM صحبت می شود منظور QAM مستطیلی خواهد بود.



شکل 5 منظومه‌ی مدولاسیون M-QAM مستطیلی برای مقادیر مختلف M

کدگذاری Gray

بسته به مدولاسیونی که برای ارسال انتخاب می‌شود، هر k بیت از رشته بیت تولیدی به یک سمبل مشخص تبدیل می‌شود. به عنوان مثال شکل 6، نگاشت طبیعی (اعداد مشکی) و گری (اعداد قرمز) را در مدولاسیون 4QAM نشان می‌دهد. همان‌طور که در شکل مشاهده می‌شود، با استفاده از کدگذاری گری، اگر تحت تأثیر نویز، یک سمبل با سمبل مجاور خود جابه‌جا شود، فقط یک بیت به اشتباه کدگذاری می‌شود؛ اما در صورت استفاده از کدگذاری طبیعی امکان دارد که هر دو بیت به اشتباه کدگذاری شود. از این رو معمولاً از کدگذاری گری استفاده کنیم.



شکل 6 نگاشت طبیعی (عددهای مشکی) و نگاشت گری (عددهای قرمز)

در کد گری، هر دو دنباله‌ی متوالی فقط در یک بیت با یکدیگر اختلاف دارند. برای تولید دنباله‌ی n بیتی این کد به روش زیر عمل

می‌کنیم.

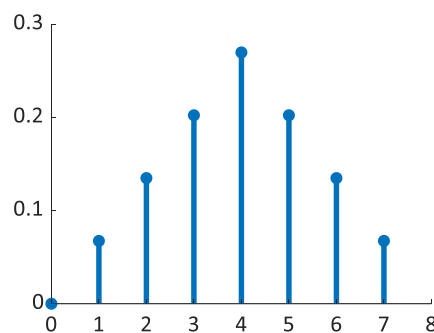
| $n = 1$ | $n = 2$ | $n = 3$ | $n = 4$ | ... |
|---|---|---|--|-----|
| $A_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}_{2 \times 1}$ | $A_2 = \begin{bmatrix} 0_{2 \times 1} & A_1 \\ 1_{2 \times 1} & A_1^B \end{bmatrix}_{4 \times 2}$ | $A_3 = \begin{bmatrix} 0_{4 \times 1} & A_2 \\ 1_{4 \times 1} & A_2^B \end{bmatrix}_{8 \times 3}$ | $A_4 = \begin{bmatrix} 0_{8 \times 1} & A_3 \\ 1_{8 \times 1} & A_3^B \end{bmatrix}_{16 \times 4}$ | ... |

در روند ذکر شده، ماتریس A_i^B از معکوس کردن هر یک از ستون‌های A_i به دست می‌آید. به عبارتی عملگر ماتریسی B به صورتی که در ادامه آمده است، عمل می‌نماید.

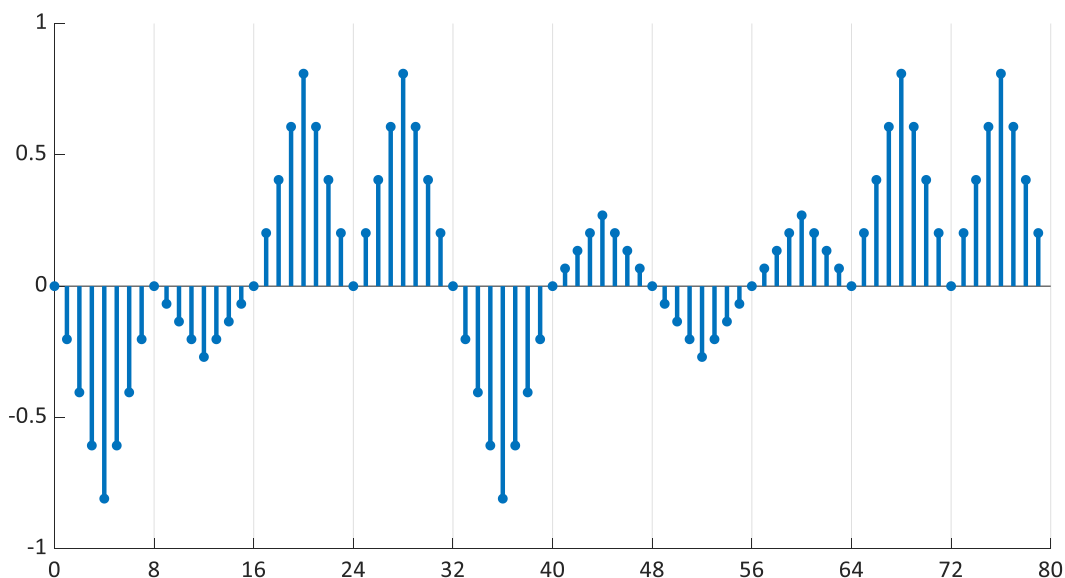
$$A_i = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,i} \\ a_{2,1} & a_{2,2} & \dots & a_{2,i} \\ \vdots & \vdots & \ddots & \vdots \\ a_{2^{i-1},1} & a_{2^{i-1},2} & \dots & a_{2^{i-1},i} \end{bmatrix} \Rightarrow A_i^B = \begin{bmatrix} a_{2^i,1} & a_{2^i,2} & \dots & a_{2^i,i} \\ a_{2^{i-1},1} & a_{2^{i-1},2} & \dots & a_{2^{i-1},i} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,1} & a_{1,2} & \dots & a_{1,i} \end{bmatrix}$$

اعمال شکل دهی پالس

پس از انتخاب سمبل‌ها لازم است که از شکل دهی پالس برای ارسال سمبل‌ها استفاده کنیم. سیگنال شکل دهنده‌ی پالس را با $p(t)$ نشان می‌دهیم. شکل 7، $p(t)$ مثلثی را نشان می‌دهد. فرض کنید که می‌خواهیم یک دنباله‌ی ۱۰ تایی از سمبل‌های 4PAM را توسط $p(t)$ نشان داده در شکل 7 ارسال کنیم. در صورتی که دنباله‌ی سمبل‌ها به صورت $s = -3, -1, 3, 3, -3, 1, -1, 1, 3, 3$ باشد، سیگنال ارسالی به صورت شکل 8 خواهد بود. برای تشکیل سیگنال ارسالی، دو روش ضرب کرونیگر^۷ و استفاده از کانولوشن وجود دارد که در ادامه به توضیحاتی در مورد آن‌ها می‌پردازیم.



شکل 7 شکل دهنده‌ی پالس مثلثی



شکل 8 سیگنال ارسالی با استفاده از شکل دهنده‌ی پالس مثلثی

تولید سیگنال ارسالی با استفاده از ضرب کرونیگر

ضرب کرونیگر دو ماتریس A و B را که با $A \oplus B$ نشان می‌دهیم، به صورت زیر تعریف می‌شود.

^۷ Kronecker Product

$$\mathbf{A}_{m \times n} \oplus \mathbf{B}_{p \times q} = \begin{bmatrix} A_{11}\mathbf{B} & \dots & A_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{m1}\mathbf{B} & \dots & A_{mn}\mathbf{B} \end{bmatrix}$$

در صورتی که \mathbf{A} و \mathbf{B} هر دو بردارهای سطری باشند، حاصل این عملگر به صورت زیر خواهد بود.

$$\mathbf{A}_{1 \times n} \oplus \mathbf{B}_{1 \times q} = [A_1\mathbf{B} \quad A_2\mathbf{B} \quad \dots \quad A_n\mathbf{B}]$$

بنابراین در اینجا می‌توان سیگنال ارسالی شکل 8 را با استفاده از ضرب کرونیگر تولید کرد.

تولید سیگنال ارسالی با استفاده از کانولوشن

دنباله‌ی سمبل‌ها را می‌توان به صورت $s = \sum_{i=0} s_i \delta[n - i]$ نشان داد. s را می‌توان به صورت زیر Zero Pad نمود.

$$s' = \sum_{i=0} s_i \delta[n - iL], \quad L = \text{length}(p)$$

حاصل کانولوشن سیگنال s' و p به صورت زیر خواهد بود که همان نتیجه‌ی مطلوب را به دست می‌دهد.

$$s' * p = \left(\sum_{i=0} s_i \delta[n - iL] \right) * p = \sum_{i=0} s_i p[n - iL]$$

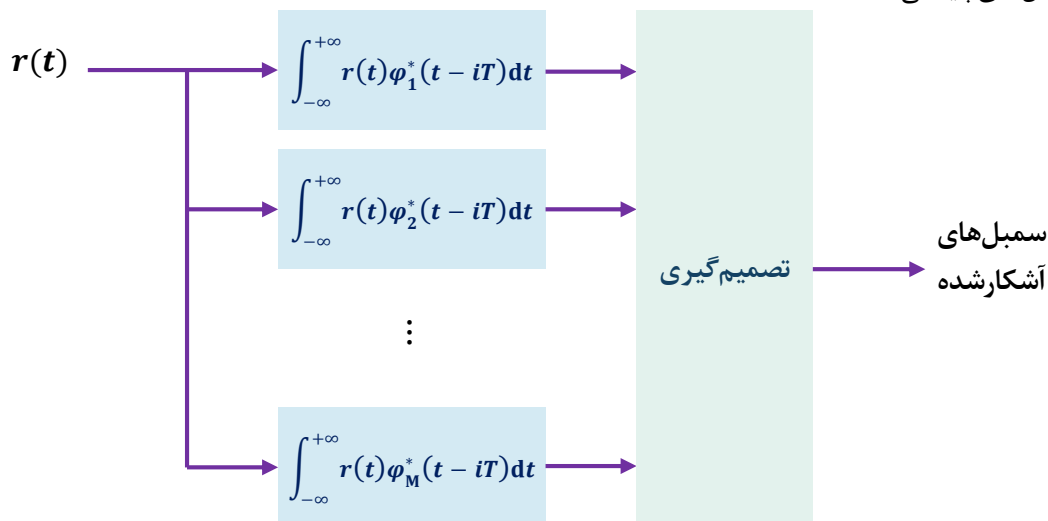
نکته: در صورتی که پالس‌های ارسالی با یکدیگر تداخل داشته باشند، روش ضرب کرونیگر سیگنال صحیح را تولید نمی‌کند.

تشخیص تصویر سیگنال دریافتی در راستای سیگنال‌های پایه

از آنجا که سمبل‌های ارسالی توسط سیگنال‌های پایه ارسال شده‌اند، می‌بایست در گیرنده، قدرت سیگنال در راستای هر یک از پایه‌ها به دست آید تا سیگنال دریافتی را آشکار کنیم. این کار با استفاده از Correlator و یا فیلتر منطبق انجام می‌شود.

Correlator

شکل 9 شمای کلی یک گیرنده را با استفاده از Correlator (بلوک‌های آبی رنگ) نشان می‌دهد. خروجی هر یک از بلوک‌های خاکستری، قدرت سیگنال را در راستای یکی از سیگنال‌های پایه نشان می‌دهد. این مقادیر پس از ورود به بلوک تصمیم‌گیری، با منظومه‌ی سیگنالی مدولاسیون مقایسه می‌شوند و نزدیک‌ترین سمبل به آن به عنوان سمبل ارسالی انتخاب می‌شود. در بلوک‌های مشخص شده، T طول زمانی سیگنال‌های پایه می‌باشد.



شکل 9 استفاده از Correlator برای اندازه‌گیری قدرت سیگنال در راستای هر سیگنال پایه

فیلتر منطبق

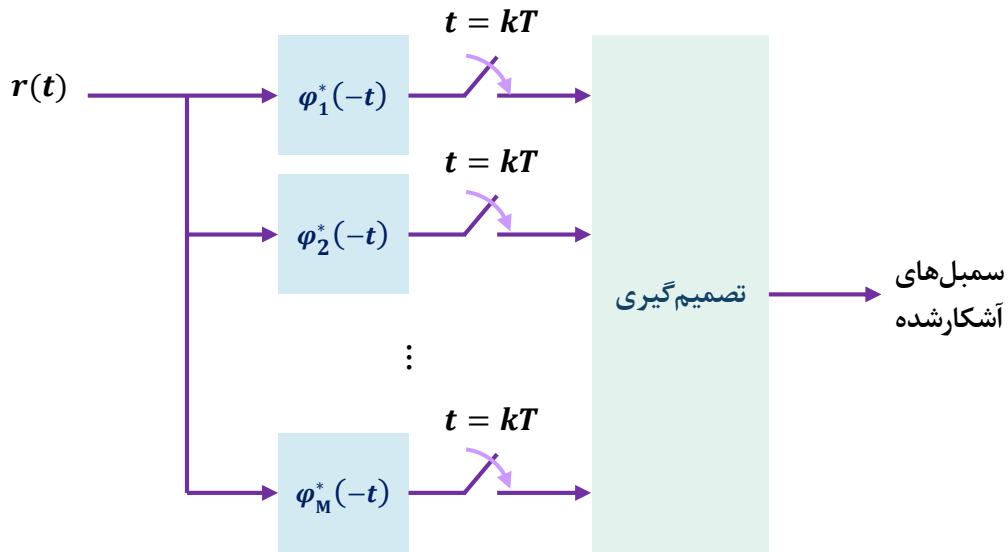
اگر بنا باشد به جای استفاده از انتگرال‌گیر، از فیلتری استفاده شود که خروجی آن شبیه Correlator است، به صورت زیر عمل

می‌شود.

⁸ Matched Filter

$$\int_{-\infty}^{\infty} r(t) \cdot \phi_m^*(t - iT) dt = \int_{-\infty}^{\infty} r(\tau) \cdot \phi_m^*(\tau - iT) d\tau = r(t) * \phi_m^*(-t)|_{t=iT}$$

با توجه به رابطه‌ی فوق می‌توان به جای استفاده از انتگرال گیر، فیلتر $\phi_m^*(-t)$ را بر سر راه سیگنال دریافتی قرار داد و در نقاط iT از آن نمونه‌گیری کرد. شکل 10، ساختار کلی یک گیرنده را با استفاده از Matched Filter نشان می‌دهد.



شکل 10 استفاده از فیلتر منطبق برای اندازه‌گیری قدرت سیگنال در راستای هر سیگنال پایه

گیرنده‌ی حداقل فاصله

به گیرنده‌ای که سمبل دریافتی را با منظومه‌ی سیگنالی یک مدولاسیون مقایسه و نزدیکترین سمبل به سیگنال دریافتی را از میان سمبل‌های موجود در منظومه‌ی سیگنالی انتخاب کند، گیرنده‌ی حداقل فاصله گفته می‌شود.

شرح آزمایش

آزمایش ۱-۴: پیاده‌سازی فرستنده مدولاسیون PAM

در این آزمایش بنا داریم تعدادی بسته‌ی داده را برای ارسال از طریق مدولاسیون 4PAM آماده نماییم. از آن جا که در آزمایش‌های آینده، بناست این فرستنده با استفاده از رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی شود، بنابراین می‌بایست با استفاده از پارامترها و ملاحظات عملی، این کار انجام شود.

پارامترهایی که در این پیاده‌سازی اهمیت دارد شامل نرخ ارسال بیت (**bit_rate**)، ضریب افزایش نرخ نمونه‌برداری سمبل‌ها یا تعداد نمونه‌های هر سمبل (**smpl_per_syml**)، تعداد سمبل‌های ارسالی در هر بسته‌ی داده (**pkt_size**) و مدت زمان ارسال داده (**stop_time**) می‌باشد. در این آزمایش مقدار پارامترهای مورد نیاز به صورت جدول 3 می‌باشد.

جدول 3 پارامترهای آزمایش ۱-۴

| پارامتر | smpl_per_syml | pkt_size |
|---------|---------------|----------|
| مقدار | ۸ | ۱۰ |

۱. تولید بیت: با استفاده از تابع **bit_gen** تعدادی بیت ۰ و ۱ را متناسب با طول بسته‌ی داده و با در نظر گرفتن پارامترهای مدولاسیون 4PAM تولید نموده و آن را درون ماتریس **b_tx** ذخیره نمایید. ماتریس **b_tx** یک ماتریس با اندازه‌ی $\text{pkt_size} \times (1 \times 2)$ می‌باشد.

۲. نگاشت بیت به سمبل:

گام ۱. تولید دنباله‌ی کد گری به صورت باینری: تابعی بنویسید که آرگومان ورودی آن تعداد بیت‌های دنباله باشد و خروجی آن کد گری مربوط به این دنباله باشد. سطر اول این تابع می‌بایست به صورت زیر باشد.

```
function [b_gray] = gray_code(k)
```

گام ۲. کدگذاری گری بیت‌های تولیدی: با استفاده از تابع **gray_code**، ماتریس مربوط به کدگذاری گری باینری مدولاسیون 4PAM را تولید کنید و آن را در ماتریس **b_gray** ذخیره کنید. بردار جدیدی متناظر با ماتریس **b_tx** و با نام **sym_idx** تولید نمایید و نگاشتی یک‌به‌یک بین سطرهای ماتریس **b_tx** و ماتریس **b_gray** برقرار کنید. به عبارتی سطر **sym_idx** بردار **sym_idx**، شماره‌ی سطری از **b_gray** را نشان می‌دهد که برابر با سطر **sym_idx** ماتریس **b_tx** باشد.

گام ۳. تولید سمبل‌های ارسالی: با استفاده از تابع **constellation**، تمامی سمبل‌های ارسالی یک مدولاسیون 4PAM را تولید کرده و سمبل‌های متناظر با هر سطر بردار **sym_idx** را تولید کنید. این سمبل‌ها را در بردار **mod_sym** ذخیره نمایید.

۳. شکل‌دهی پالس ارسالی: همان‌طور که در بخش تئوری بیان گردید، سیگنال ارسالی را می‌توان با دو روش ضرب کرونیگر و کانولوشن تولید و پالس ارسالی را شکل‌دهی نمود. تابعی بنویسید که بردار شماره‌ی سمبل‌های تولیدشده (**sym_idx**)، مدولاسیون (**modulation**)، مرتبه‌ی مدولاسیون (**M**)، نرخ نمونه‌برداری (**fs**)، تعداد نمونه‌های هر سمبل (**smpl_per_syml**)، نام تابع شکل‌دهنده‌ی پالس (**pulse_name**) و روش تولید شکل‌دهی پالس (**mode**) را دریافت کند و نمونه‌های سیگنال ارسالی متناظر با یک بسته و منظومه‌ی سیگنالی را تولید کند. سطر اول این تابع می‌بایست به صورت زیر باشد.

```
function [tx_smpl, cons] = pulse_modulation(sym_idx, modulation, M, fs, smpl_per_syml, pulse_name, pulse_shape_mode, varargin)
```

توجه داشته باشید که گام ۳ سوال قبل را مجدد در قالب یک تابع منسجم انجام داده‌ایم. در این جا **pulse_shape_mode** روش شکل‌دهی پالس می‌باشد که یکی از دو حالت 'kron' و 'conv' را اختیار می‌کند.

با استفاده از این تابع نمونه‌های ارسالی مربوط به یک بسته‌ی داده را آماده‌ی ارسال کرده و حاصل را در بردار `tx_out` ذخیره نمایید. تابع شکل‌دهی پالس در این آزمایش پالس مثلثی است. اثر شکل پالس‌ها در آزمایش بعد مورد بررسی قرار می‌گیرد.

(راهنمایی ۱: ضرب کرونیگر در برنامه‌ی MATLAB با استفاده از دستور `kron` انجام می‌شود که شیوه‌ی استفاده از آن به صورت `tx_out = kron(s, p)` می‌باشد.)

(راهنمایی ۲: برای انجام عمل Zero Padding از تابع `upsmpl` خود استفاده کنید. دقت نمایید که تابع شما نباید بعد از آخرین سمبل صفر اضافه نماید. حال سیگنال ارسالی را توسط عملگر کانولوشن و به صورت `tx_out = conv(s_zero_pad, p)` تولید نمایید.)

آزمایش ۲-۴: پیاده‌سازی گیرنده‌ی مدولاسیون PAM

در ادامه‌ی می‌خواهیم گیرنده‌ی یک مدولاسیون PAM را پیاده‌سازی نماییم.

۱. تشخیص قدرت سیگنال دریافتی در راستای سیگنال‌های پایه: تابعی بنویسید که دو بردار `rx_smpl` و `p` را دریافت کند و قدرت سیگنال را در راستای `p` محاسبه کند. این تابع باید بتواند قدرت سیگنال را در راستای `p`، با استفاده از هر دو روش correlator و فیلتر منطبق محاسبه کند. سطر اول این تابع می‌بایست به صورت زیر باشد.

```
function [rx_sym] = corr_match(rx_smpl, p, smpl_per_symbl, rx_mode)
```

در این تابع `rx_mode` می‌تواند یکی از دو حالت `'correlator'` و یا `'matched_filter'` را اختیار کند.

۲. گیرنده‌ی حداقل فاصله: تابعی بنویسید که منظومه‌ی سیگنالی (`constellation`) و بردار سمبل‌های دریافتی (`rx_sym`) را به عنوان ورودی بگیرد و در خروجی، برداری به طول `r` تولید کند. المان `i`ام بردار سمبل‌های آشکارشده‌ی `det_sym`، نزدیکترین مقدار `constellation` به المان `i`ام بردار `rx_sym` خواهد بود. سطر اول این تابع به صورت زیر می‌باشد.

```
function [det_sym] = min_dist_detector(rx_sym, constellation);
```

۳. دمدولاسیون یا آشکارسازی پالس: نمونه‌های ارسالی شده (`tx_smpl`) را به عنوان نمونه‌های دریافتی درون بردار `rx_smpl` قرار دهید. در این جا می‌خواهیم تابعی بنویسیم که دو عمل قبل را به صورت منسجم در قالب یک تابع انجام دهد. ورودی‌های این تابع نمونه‌های سیگنال دریافتی (`rx_signal`)، نام مدولاسیون (`modulation`)، مرتبه‌ی مدولاسیون (`M`)، نرخ نمونه‌برداری (`fs`)، تعداد نمونه‌های هر سمبل (`smpl_per_symbl`)، نام تابع شکل‌دهنده‌ی پالس (`pulse_name`) و روش تشخیص قدرت سیگنال دریافتی در راستای سیگنال‌های پایه (`mode`) را دریافت کند و خروجی آن سمبل‌های دریافتی (`rx_sym`) و اندیس سمبل‌های آشکارشده (`det_sym_idx`) باشد.

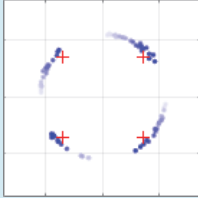
```
function [det_sym_idx, rx_sym] = pulse_demodulation(rx_smpl, modulation, M, fs, smpl_per_symbl, pulse_name, rx_mode, varargin)
```

۴. تبدیل از کدگذاری گری به کدگذاری باینری: به کمک بردار `det_sym_idx` و ماتریس `b_gray` اندیس سمبل‌های آشکارشده‌ی با کدگذاری باینری را به دست آوردید و درون متغیر `det_bit` قرار دهید. برای این منظور متناظر با محتوای هر سطر `det_sym_idx` سطری متناظر با آن محتوا از `b_gray` انتخاب می‌شود.



- [1] J. G. Proakis and M. Salehi, *Digital Communications*. Boston: McGraw-Hill, 2008.
- [2] M. Rice, *Digital Communications: A Discrete-Time Approach*. Upper Saddle River, NJ: Pearson Education, 2009.
- [3] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.





آزمایش پنجم

کاوش در مدولاسیون دیجیتال خطی

اهداف آزمایش

سامانه‌های مخابرات دیجیتال بی‌سیم اطراف ما را فرا گرفته است و ارتباطات صدا و داده در سامانه‌های مخابرات سلولی، WiFi، Bluetooth و سایر استانداردها را فراهم می‌کند. در طی چندین آزمایش آینده بلوک‌های پایه‌ی مخابرات دیجیتال شامل الگوهای مدولاسیون، دریافت سمبل و بیت، عملکرد در حضور نویز، کدگذاری و کدگشایی تقاضای، تبدیل فرکانس به بالا و پایین دیجیتال و ... ارایه می‌شود. در این آزمایش در ادامه‌ی آزمایش قبل با الگوهای مدولاسیون دیجیتال خطی و شکل‌دهی پالس آشنا می‌شویم و به کاوش بر روی پارامترهای مختلف این مدولاسیون‌ها می‌پردازیم. هم‌چنین تأثیراتی که کانال می‌تواند بر روی فرآیند آشکارسازی بگذارد نیز مورد بررسی قرار می‌گیرد.

در انتهای این آزمایش دانشجو می‌بایست:

- منظومه‌های مدولاسیون‌های خطی M-PAM، M-PSK و M-QAM را بشناسد.
- بتواند رشته‌های بیت را به سمبل‌های مدولاسیون‌های مختلف در حالت سمبل‌های مختلط و سمبل‌های I/Q تبدیل نماید.
- بر روی کدگذاری Gray برای مدولاسیون‌های مختلف مسلط باشد.
- با علت استفاده از شکل‌دهی پالس در سمبل‌های ارسالی آشنا شود.
- با اثراتی که شکل‌دهی پالس بر روی پهنای باند طیف ارسالی آشنا شود.
- بتواند سیگنال را در یک کانال باندپایه ارسال نماید و اثر نویز را در تبدیل سمبل‌ها به رشته‌های بیت را بشناسد.
- بتواند به کانال باندپایه، تأخیر و اختلاف فاز اضافه نماید و با اثر این عوامل بر روی منظومه‌ی سیگنالی و خطای آشکارسازی آشنا شود.
- با طراحی Correlator و فیلتر منطبق آشنا شود.
- بتواند سمبل‌های دریافتی مدولاسیون‌های مختلف را به رشته‌های بیت تبدیل نماید.
- بتواند گیرنده‌ی حداقل فاصله را طراحی نماید.

ابزارهای مورد نیاز

- رایانه
- نرم‌افزار MATLAB R2020b
- رادیو نرم‌افزار ADALM-PLUTO
- کابل کوکسیال و آنتن



تئوری

تعیین نسبت انرژی متوسط بیت به چگالی نویز

به منظور بررسی رفتار و عملکرد مدولاسیون‌های دیجیتال خطی در عبور از یک کانال با نویز سفید جمع‌شونده می‌بایست پارامتر نسبت انرژی متوسط بیت به چگالی نویز (E_b/N_0) به درستی تعریف شود. انرژی متوسط بیت را می‌توان بر اساس تعداد بیت‌های معرف یک سمبل و انرژی متوسط سمبل یک مدولاسیون دیجیتال تعیین نمود. با داشتن نسبت $\frac{E_b}{N_0}$ می‌توان N_0 را محاسبه نمود. البته ذکر این نکته حایز اهمیت است که در سامانه‌های عملی، ابتدا مقدار N_0 که به پارامترهای گیرنده ارتباط دارد محاسبه شده و سپس توان فرستنده برای رسیدن به انرژی متوسط بیت مورد نظر تعیین می‌گردد. از آن‌جا که برای محاسبه‌ی عملکرد یک مدولاسیون در محیط AWGN تنها نسبت E_b/N_0 اهمیت دارد، در شبیه‌سازی استفاده از هر یک از دو رویکرد فوق چندان تفاوتی ندارد.



شرح آزمایش

آزمایش ۱-۵: مقداردهی‌های اولیه

در این آزمایش بنا داریم تعدادی بسته‌ی داده را برای ارسال از طریق مدولاسیون‌های خطی آماده نماییم. از آن‌جا که در این آزمایش بناسات این فرستنده با استفاده از رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی شود، می‌بایست با استفاده از پارامترها و ملاحظات عملی، این کار انجام شود.

پارامترهایی که در این پیاده‌سازی اهمیت دارد شامل نرخ نمونه‌برداری (fs)، ضریب افزایش نرخ نمونه‌برداری سمبل‌ها یا تعداد نمونه‌های هر سمبل ($smpl_per_syml$)، تعداد سمبل‌های ارسالی در هر بسته‌ی داده (pkt_size) و مدت زمان ارسال داده ($stop_time$) می‌باشد. در این آزمایش مقدار پارامترهای مورد نیاز به صورت جدول ۳ می‌باشد.

جدول ۴ پارامترهای آزمایش ۵

| پارامتر | fs | $smpl_per_syml$ | pkt_size | $stop_time$ |
|---------|-------|-------------------|--------------------------------------|--------------|
| مقدار | 10MHz | ۸ | شبیه‌سازی: ۱۰۰۰۰۰ سخت‌افزار: ۱۰۰۰ | ۱۰۰ ثانیه |

دقت نظر داشته باشید که در هر آزمایش تنها می‌خواهیم برنامه‌هایی که در آزمایش قبل نوشته شده است را تکمیل نماییم. هم‌چنین ابتدای برنامه‌نویسی pkt_size را مقدار کوچکی مانند ۱۰ قرار دهید و پس از اطمینان از برنامه مقدار آن را افزایش دهید.

۱. تولید M-File مربوط به مقدار دهی اولیه: در آزمایش‌ها می‌خواهیم، مدولاسیون‌های مختلف، عوامل غیرایده‌آل کانال، استفاده یا عدم استفاده از سخت‌افزار، شکل پالس‌های مختلف و ... را مورد بررسی قرار دهیم. از این رو بهتر است برنامه به صورت یک کل نوشته شود تا با تغییر چند پارامتر در فایل مقداردهی اولیه، تغییرات مد نظر در کل برنامه اعمال شود. در ادامه قالب کلی این پارامترها آورده شده است. فایل `dcl_init.m` را ایجاد نموده و این موارد را در آن وارد نمایید. سپس می‌بایست این فایل را از برنامه‌ی اصلی خود فراخوانی نمایید.

```
%% Simulation Parameters
flg_hrdwr_usg = 0;
stop_time = 100;
```

```
%% Receiver Parameters
fs = 10e6;
ts = 1/fs;
pkt_size = 1e6;
rx_alg = 0;
```

```
% Baseband Sampling Rate (65105 to 61.44e6 Hz)
% Baseband Sampling Time
% Number of Symbol in Each Packet
% Receiver Detection Algorithm
```



```

cmpnst_mode = 0; % Compensate Mode (0: No Compensation, 1:
Amplitude Compensation, 2: Phase Compensation, 3: Compensation)
%% Modulation Parameters
modulation = 'psk'; % Modulation Name ('psk', 'pam', 'qam', 'fsk')
k = 2; % Bit Per Symbol
M = 2^k; % Modulation Order

smpl_per_symb1 = 8; % Sample Per Symbol
Ts = smpl_per_symb1*ts; % Symbol Time

flg_gray_encode = 1; % Gray Code Usage Flag
mod_det_opt = 'coherent'; % Modulation Detection Option ('coherent',
'noncoherent')

% Pulse Shape Parameters
pulse_name = 'triangular'; % Name of Pulse Shaping Function
beta = 0.99; % Parameter for RC, RRC and Gaussian Pulse Shape
span_in_symb1 = 0; % Truncation Length for RC, RRC and Gaussian
Pulse Shape (Multiple of Symbol Time)

% Header Option
flg_add_hdr = 0; % Flag For Having Packets with Header

% SNR Bound for BER Plots
snr_min = 0; % Minimum SNR (dB)
snr_max = 10; % Maximum SNR (dB)
snr_step = 1; % SNR Step (dB)
snr_db = snr_min:snr_step:snr_max; % SNR Vector (dB)

%% Channel Parameters
chnl_delay_in_smpl = 0; % Channel Delay in Sample
chnl_phase_offset = 0 * pi/180; % Channel Phase Offset
chnl_freq_offset = 0; % Channel Frequency Offset

%% Hardware Parameters
% Transmitter Parameters
tx_fc = 2400e6; % Set Transmitter Center Frequency (AD9363: 325-
3800MHz) (AD9364: 70-6000MHz)
tx_gain = 0; % Set Transmitter Attenuation As a Negative Gain
(-89.75 to 0 dB)
tx_address = 'usb:0'; % Set Transmitter Identification Number

% Receiver Parameters
rx_fc = 2400e6; % Set Receiver Center Frequency (AD9363: 325-
3800MHz) (AD9364: 70-6000MHz)
rx_gain = 20; % Set Receiver Gain (-4dB to 71dB)
rx_address = 'usb:0'; % Set Receiver Identification Number

% Initialize ADALM-PLUTO
if flg_hrdwr_usg
    dev = sdrdev('Pluto'); % Create Radio Object for ADALM-PLUTO
    setupSession(dev)
    configurePlutoRadio('AD9364'); % Configure ADALM-PLUTO Radio Firmware
end

```

آزمایش ۲-۵: پیاده‌سازی فرستنده مدولاسیون

۱. تولید بیت: با استفاده از تابع **bit_gen** تعدادی بیت ۰ و ۱ را متناسب با طول بسته‌ی داده و با در نظر گرفتن پارامترهای مدولاسیون 4PSK تولید نموده و آن را درون ماتریس **b_tx** ذخیره نمایید. ماتریس **b_tx** یک ماتریس با اندازه‌ی $\text{pkt_size} \times k$ می‌باشد.

۲. نگاشت بیت به سمبل:

گام ۱. کدگذاری گری بیت‌های تولیدی: با استفاده از تابع `gray_code`، ماتریس مربوط به کدگذاری گری مدولاسیون 4PSK را تولید کنید و آن را در ماتریس `b_gray` ذخیره کنید. بردار جدیدی متناظر با ماتریس `b_tx` و با نام `sym_idx` تولید نمایید و نگاشتی یک‌به‌یک بین سطرهای ماتریس `b_tx` و ماتریس `b_gray` برقرار کنید. به عبارتی سطر `i`ام بردار `sym_idx`، شماره‌ی سطر `i`ام از `b_gray` را نشان می‌دهد که برابر با سطر `i`ام ماتریس `b_tx` باشد. (راهنمایی: بهتر است سطرهای ماتریس `b_gray` را به یک عدد صحیح تبدیل نمایید.)

گام ۲. کدگذاری طبیعی بیت‌های تولیدی: با استفاده از فلگ `flg_gray_encode`، قابلیت‌ی را به برنامه‌ی خود اضافه نمایید که بتوان کدگذاری گری را اعمال نمود.

گام ۳. تولید سمبل‌های ارسالی: با استفاده از تابع `constellation`، تمامی سمبل‌های ارسالی مدولاسیون 4PSK را تولید کرده و سمبل‌های متناظر با هر سطر بردار `sym_idx` را تولید کنید. این سمبل‌ها را در بردار `mod_sym` ذخیره نمایید.

۳. شکل‌دهی پالس ارسالی: با استفاده از تابع `pulse_modulation` نمونه‌های ارسالی مربوط به یک بسته‌ی داده را آماده‌ی ارسال کرده و حاصل را در بردار `tx_smp1` ذخیره نمایید. تابع شکل‌دهی پالس را فعلاً مثلاً در نظر بگیرید.

آزمایش ۳-۵: مدل‌سازی کانال

۱. افزودن تأخیر در کانال: برای شبیه‌سازی تأخیر در کانال به اندازه‌ی پارامتر `chnl_delay_in_smp1` به ابتدای بردار `tx_smp1` صفر اضافه نمایید. حاصل را درون بردار `tx_smp1_delayed` قرار دهید. دقت نمایید که طول بردار افزایش می‌یابد و می‌بایست در ادامه‌ی برنامه اثر این افزایش لحاظ شود. در این جا فعلاً مقدار تأخیر برابر با صفر قرار دهید.

۲. اعمال اختلاف فاز در کانال: برای اعمال اختلاف فاز در کانال می‌بایست بردار `tx_smp1_delayed` را در `exp(1i*chnl_phase_offset)` ضرب شود. حاصل را در بردار `rx_smp1` قرار دهید. در این جا نیز آفست فاز را برابر با صفر قرار دهید.

۳. شبیه‌سازی کانال با نویز سفید گاوسی

گام ۱. تعیین واریانس نویز بر اساس نسبت سیگنال به نویز (E_b/N_0): ابتدا بر اساس خروجی تابع `constellation`، مقدار متوسط انرژی سمبل را به دست آورید (`Es_avg`). حال با استفاده از متوسط انرژی سمبل، مقدار انرژی متوسط بیت را به دست آورده و برابر متغیر `Eb` قرار دهید. مقدار E_b/N_0 را برابر با 3dB در نظر بگیرید (به عبارتی پارامترهای `snr_min` و `snr_max` برابر 3 تنظیم می‌شوند). سپس واریانس نویز را بر اساس نسبت E_b/N_0 به دست آورید و آن را درون متغیر `var_noise` قرار دهید.

گام ۲. افزودن نویز به سیگنال: با استفاده از تابع `randn` یک بردار نویز مختلط با واریانس `var_noise` و ابعاد برابر با `tx_smp1` تولید نمایید و آن را `noise_smp1` نامگذاری نمایید. سپس این بردار را با بردار `rx_smp1` جمع نموده و حاصل را `rx_smp1_noise` بنامید.

آزمایش ۴-۵: پیاده‌سازی گیرنده‌ی مدولاسیون

در ادامه‌ی فایل آزمایش قبل می‌خواهیم گیرنده‌ی یک مدولاسیون خطی را پیاده‌سازی نماییم.

۵. دمدولاسیون یا آشکارسازی پالس: نمونه‌های دریافت شده به همراه نویز (`rx_smp1_noise`) را که اثر تأخیر کانال در آن جبران شده را درون بردار `rx_smp1` قرار دهید. با استفاده از تابع `pulse_demodulation` با ورودی‌های نمونه‌های سیگنال دریافتی (`rx_smp1`)، نام مدولاسیون (`modulation`)، مرتبه‌ی مدولاسیون (`M`)، نرخ نمونه‌برداری (`fs`)، تعداد نمونه‌های هر سمبل (`smp1_per_symbl`)، نام تابع شکل‌دهنده‌ی پالس (`pulse_name`) و روش تشخیص قدرت سیگنال دریافتی در راستای سیگنال‌های پایه (`mode`)، سمبل‌های دریافتی (`rx_sym`) و اندیس سمبل‌های آشکار شده (`det_sym_idx`) را به دست آورید.



۶. تبدیل از کدگذاری گری به کدگذاری باینری: به کمک بردار `det_sym_idx` و ماتریس `b_gray` اندیس سمبل‌های آشکارشده‌ی با کدگذاری باینری را به دست آوردید و درون متغیر `det_bit` قرار دهید. برای این منظور متناظر با محتوای هر سطر `det_sym_idx` سطری متناظر با آن محتوا از `b_gray` انتخاب می‌شود. این عمل را می‌بایست برای حالتی که از کدگذاری باینری نیز استفاده می‌شود و `flg_gray_encode = 0` است نیز مدیریت شود.

۷. محاسبه‌ی خطای سمبل: با مقایسه‌ی تعداد اختلاف‌های بردارهای `det_sym_idx` و `sym_idx` و محاسبه نسبت اختلاف این دو بردار به تعداد کل سمبل‌ها، خطای سمبل را به دست آورده و آن را در متغیر `ser` قرار دهید.

۸. تبدیل سمبل به بیت و محاسبه‌ی خطای بیت: با مقایسه‌ی تعداد اختلاف ماتریس‌های `det_bit` و `b_tx` و محاسبه نسبت تعداد اختلاف این دو ماتریس به تعداد کل بیت‌ها، خطای بیت را به دست آورده و آن را در متغیر `ber` قرار دهید.

آزمایش ۵-۵: خواسته‌های کلی

۱. رسم نمودار نرخ خطای بیت: با تغییر نسبت سیگنال به نویز بین 0 تا 10dB نرخ خطای بیت را برای مدولاسیون‌های 16QAM، 8PSK و 4PSK را به دست آورده و در یک نمودار رسم نمایید و آن را با خروجی تابع `berawgn` نرم‌افزار MATLAB مقایسه نمایید. این کار را برای شکل موج مثلثی و `root raised cosine` انجام دهید. هم‌چنین اثر اعمال کدگذاری گری و کدگذاری باینری را نیز بر روی نمودار نرخ خطای بیت نشان دهید.

۲. اثر تأخیر در کانال:

در این جا فرض نمایید مدولاسیون 4PSK است و نسبت سیگنال به نویز (E_b/N_0) برابر با 10dB است.

گام ۱. اثر تأخیر در کانال بر روی نرخ خطای بیت و منظومه‌ی سیگنالی: تأخیر کانال را برابر با گرد شده‌ی 0.1، 0.5 و 0.8 برابر طول یک سمبل قرار دهید (یک عدد صحیح معادل تعداد نمونه‌های زمانی تأخیر یافته). نرخ خطای بیت را به دست آورد و با حالت بدون تأخیر مقایسه نمایید. منظومه سیگنالی حالت‌های فوق را نیز با یکدیگر مقایسه کرده و علت مشاهدات خود را توضیح دهید. (`flg_hrdwr_usg = 0` و `rx_alg = 0`)

گام ۲. اضافه کردن هدر: برای یافتن نقطه‌ی شروع درست داده‌ها، تعدادی داده‌ی مشخص به عنوان هدر در ابتدای رشته بیت‌های ارسالی قرار دهید. نمونه‌های زمانی مربوط به هدر را از بردار `tx_smpl` جدا کرده و در متغیر `hdr_smpl` ذخیره نمایید. بیت‌های هدر می‌بایست به صورت زیر باشد. این رشته بیت خواص خودهمبستگی خوبی دارد.

```
hdr_bit = repmat(reshape(de2bi(hex2dec(['1C6387FF5DA4FA325C895958DC5'])))', [1, 1], 1, k);
```

گام ۳. تخمین زمان شروع بسته و جبران اثر تأخیر کانال: با استفاده از همبستگی بردار `rx_smpl_noise` و نمونه‌های زمانی هدر `hdr_smpl`، نقطه‌ی شروع بسته‌ی داده را تخمین بزنید. با مقایسه‌ی نرخ خطای بیت حالت بدون تأخیر و حالتی که نقطه‌ی شروع داده‌ها تخمین زده می‌شود، از عملکرد برنامه‌ی خود اطمینان حاصل نمایید. (`flg_hrdwr_usg = 0` و `rx_alg = 1`)

۳. اعمال فاز در کانال و تخمین فاز:

در این جا فرض نمایید مدولاسیون 4PSK است و نسبت سیگنال به نویز (E_b/N_0) برابر با 10dB است.

گام ۱. اثر فاز اضافی کانال بر نرخ خطای بیت: اختلاف فاز کانال را برابر ۳۰ درجه قرار دهید. ابتدا برای مدولاسیون 4PSK نرخ خطای بیت را به دست آورده و با حالت بدون اختلاف فاز مقایسه نمایید. منظومه سیگنالی هر دو حالت را نیز با یکدیگر مقایسه نمایید. علت مشاهدات خود را توضیح دهید.

گام ۲. تخمین فاز اضافی کانال: اگر به ابتدای بسته‌ی ارسالی هدر بخش قبل اضافه شود، با استفاده از همبستگی بردار `rx_smpl_noise` و نمونه‌های زمانی هدر `hdr_smpl`، فازی را که کانال به سیگنال اضافه کرده است را به

دست آورید. (راهنمایی: به فاز در قله‌ی خروجی همبستگی دقت نمایید. قله از روی مقدار مطلق همبستگی به دست می‌آید ولی مقدار خود همبستگی در این اندیس به دست آمده مورد استفاده قرار می‌گیرد. به عملکرد تابع **max** بر روی داده‌های مختلط توجه نمایید.)

گام ۳. جبران‌سازی فاز کانال: تلاش کنید با اعمال ضریب مناسب اثر فاز اضافی کانال را جبران نمایید.

(**cmpnst_mode = 2**)

آزمایش ۵-۵: پیاده‌سازی سخت‌افزاری

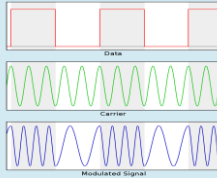
در این بخش **rx_alg = 1** و **flg_hrdwr_usg = 1** می‌باشد.

۱. ارسال نمونه‌ها با استفاده از ADALM-PLUTO: بردار **tx_smp1** مربوط به مدولاسیون 4PSK (هدر اضافه شود) را با استفاده از دستور **transmitRepeat** به مدت زمان **stop_time** ثانیه به صورت پی‌درپی در فضا ارسال نمایید. این دستور داده‌ها را از طریق USB به رادیو نرم‌افزار ارسال می‌نماید و در حافظه‌ی سخت‌افزار ذخیره می‌نماید. فرستنده داده‌ها را از حافظه‌ی رادیو نرم‌افزار قرائت کرده و مدام ارسال می‌نماید. برای این منظور ابتدا می‌بایست پس از پیکربندی رادیو نرم‌افزار یک شیء فرستنده ایجاد نموده و فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را برابر با 0dB تنظیم نمایید.
۲. دریافت سیگنال با استفاده از ADALM-PLUTO: ابتدا شیء مربوط به گیرنده را تعریف نمایید. فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را به صورت Manual و برابر 20dB تنظیم نمایید. به علت اطمینان از دریافت یک بسته‌ی کامل، تعداد نمونه‌های زمانی دریافتی را برابر با دو برابر تعداد نمونه‌های زمانی بسته در نظر بگیرید. به منظور دریافت داده، شیء گیرنده را فراخوانی نمایید و حاصل را درون متغیر **rx_smp1** قرار دهید.
۳. اثر شکل پالس بر روی پهنای باند سیگنال ارسالی: طیف فرکانسی سیگنال **rx_smp1** دریافتی از ADALM-PLUTO را برای مدولاسیون 4PSK و به ازای شکل پالس‌های مثلثی و **root raised cosine** رسم نمایید. در مورد پهنای باند و سطح باندهای جانبی آن توضیحاتی ارائه دهید. برای این منظور از برنامه‌های قبلی استفاده نمایید.
۴. دمدولاسیون و مشاهده‌ی منظومه‌ی سیگنالی: مشابه قبل عمل دمدولاسیون را انجام داده و منظومه‌ی سیگنالی مربوط به **rx_sym** را رسم نمایید و مشاهدات خود را یادداشت نمایید. منظومه‌ی سیگنالی باید بتواند به صورت به‌لحظه به‌روز شود.
۵. جبران‌سازی دامنه، فاز و تأخیر و مشاهده‌ی منظومه‌ی سیگنالی: با یافتن ابتدای بسته‌ها و جبران فاز، دامنه و تأخیر، منظومه‌ی سیگنالی مربوط به **rx_sym** را رسم نمایید. احتمال خطای بیت را نیز در این حالت محاسبه نمایید.



- [1] J. G. Proakis and M. Salehi, *Digital Communications*. Boston: McGraw-Hill, 2008.
- [2] M. Rice, *Digital Communications: A Discrete-Time Approach*. Upper Saddle River, NJ: Pearson Education, 2009.
- [3] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.





آزمایش ششم

مدولاسیون FSK همدوس

اهداف آزمایش

- پس از آشنایی با مدولاسیون‌های خطی و اثر یک کانال AWGN بر روی آن، در این آزمایش با مدولاسیون FSK همدوس آشنا می‌شویم و رفتار آن را دوباره در کانال AWGN مورد بررسی قرار می‌دهیم.
- در انتهای این آزمایش دانشجو می‌بایست:
- با مفهوم مدولاسیون‌های متعامد آشنا شود.
 - منظومه‌ی سیگنالی مدولاسیون FSK را بشناسد.
 - با شکل موج مدولاسیون FSK آشنا باشد.
 - بر اثر فاصله‌ی فرکانسی بین پایه‌های مدولاسیون FSK مسلط شود.
 - بتواند یک سیستم کامل مدولاسیون FSK را پیاده‌سازی کند.
 - احتمال خطای سمبل و بیت را در این مدولاسیون به دست آورد.
 - یه سامانه‌ی کامل مدولاسیون FSK همدوس را به کمک رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی نماید.

ابزارهای مورد نیاز

- رایانه
- نرم‌افزار MATLAB R2020b
- رادیو نرم‌افزار ADALM-PLUTO
- آنتن



مدولاسیون‌های متعامد

در مدولاسیون‌های متعامد، بُعد فضا برابر با تعداد نقاط منظومه‌ی سیگنالی می‌باشد. به عبارتی، در صورتی که N نشان‌دهنده‌ی بُعد فضا باشد، $N = M$ خواهد بود که $S_1(t), S_2(t), \dots, S_M(t)$ سیگنال‌های ارسالی می‌باشند. برای این مدولاسیون‌ها داریم:

$$\langle S_i(t), S_j(t) \rangle = \begin{cases} 0 & i \neq j \\ E_s & i = j \end{cases}$$

بدیهی است که سیگنال‌های پایه برای این مدولاسیون‌ها به صورت زیر خواهند بود:

$$\phi_i(t) = \frac{S_i(t)}{\|S_i(t)\|} = \frac{S_i(t)}{\sqrt{E_s}}$$

حالت خاصی از مدولاسیون‌های متعامد، مدولاسیون FSK⁹ می‌باشد که سیگنال‌های ارسالی به صورت زیر می‌باشند:

$$S_m(t) = \sqrt{\frac{2E_s}{T}} \cos(2\pi(f_0 + (m-1)\Delta f)t) \quad 0 < t \leq T, \quad m = 1, 2, \dots, M$$

همان‌گونه که مشاهده می‌شود، شکل‌دهنده‌ی پالس در اینجا یک مستطیل به طول T خواهد بود. معادل باندپایه برای این سیگنال‌ها به صورت زیر خواهد بود.

$$S_{\ell,m}(t) = \sqrt{\frac{2E_s}{T}} e^{j2\pi(m-1)\Delta f t}$$

حال می‌خواهیم Δf را به گونه‌ای تنظیم کنیم که سیگنال‌های ارسالی بر یکدیگر عمود باشند.

$$\begin{aligned} \langle S_{\ell,m}(t), S_{\ell,n}(t) \rangle &= \int_{-\infty}^{\infty} S_{\ell,m}(t) S_{\ell,n}^*(t) dt = \int_0^T \sqrt{\frac{2E_s}{T}} e^{j2\pi(m-1)\Delta f t} \sqrt{\frac{2E_s}{T}} e^{-j2\pi(n-1)\Delta f t} dt = \frac{2E_s}{T} \int_0^T e^{j2\pi(m-n)\Delta f t} dt \\ &= \frac{2E_s}{T} e^{j\pi(m-n)\Delta f T} \times \frac{\sin(\pi(m-n)\Delta f T)}{\pi(m-n)\Delta f} \end{aligned}$$

از طرفی داریم:

$$\langle S_m(t), S_n(t) \rangle = \frac{1}{2} \text{Re}\{\langle S_{\ell,m}(t), S_{\ell,n}(t) \rangle\}$$

بنابراین:

$$\langle S_m(t), S_n(t) \rangle = \frac{1}{2} \text{Re}\{\langle S_{\ell,m}(t), S_{\ell,n}(t) \rangle\} = E_s \frac{(\cos(\pi(m-n)\Delta f T) \times \sin(\pi(m-n)\Delta f T))}{\pi(m-n)\Delta f T} = E_s \text{sinc}(2(m-n)\Delta f T)$$

به منظور برقراری شرط تعامد لازم است که:

$$\text{sinc}(2(m-n)\Delta f T) = 0$$

بنابراین برای اینکه تعامد در باند میانی برقرار شود لازم است که شرط $\Delta f = \frac{k}{2(m-n)T}$; $k \in \mathbb{Z}$ برقرار باشد. حداقل مقدار Δf که

تعامد را برقرار می‌سازد برابر مقدار زیر است.

$$\Delta f_{\min} = \frac{1}{2T}$$

از آنجا که پهنای باند مصرفی با Δf متناسب است، معمولاً در عمل سعی می‌شود که از Δf_{\min} استفاده شود. لازم به ذکر است که

پایه‌های فضای سیگنال در باند میانی به صورت زیر می‌باشند:

$$\Phi_m(t) = \sqrt{\frac{2}{T}} \times \cos(2\pi(f_0 + (m-1)\Delta f)t), \quad 0 < t \leq T$$

⁹ Frequency Shift Keying



شرح آزمایش

آزمایش ۱-۶: مقداردهی‌های اولیه

در این آزمایش بنا داریم تعدادی بسته‌ی داده را برای ارسال از طریق مدولاسیون FSK آماده نماییم. از آن‌جا که در این آزمایش بناسست این فرستنده با استفاده از رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی شود، می‌بایست با استفاده از پارامترها و ملاحظات عملی، این کار انجام شود.

پارامترهایی که در این پیاده‌سازی اهمیت دارد شامل نرخ نمونه‌برداری (f_s)، تعداد نمونه‌های هر سمبل (smp1_per_sybl)، تعداد سمبل‌های ارسالی در بسته‌ی داده (pkt_size) و مدت زمان ارسال داده (stop_time) می‌باشد. در این آزمایش مقدار پارامترهای مورد نیاز به صورت جدول ۵ می‌باشد.

جدول ۵ پارامترهای آزمایش ۶

| پارامتر | f_s | smp1_per_sybl | pkt_size | stop_time |
|---------|-------|--------------------------|--------------------------------------|---------------------|
| مقدار | 10MHz | 64 | شبیه‌سازی: ۱۰۰۰۰۰ سخت‌افزار: ۱۰۰۰ | ۱۰۰ ثانیه |

دقت نظر داشته باشید که در این آزمایش نیز تنها می‌خواهیم برنامه‌هایی که در آزمایش قبل نوشته شده است را تکمیل نماییم.

۱. تولید M-File مربوط به مقدار دهی اولیه: قالب کلی پارامترهای مورد نیاز در این آزمایش، مانند آزمایش قبل است و فایل `dcl_init.m` می‌بایست مدولاسیون fsk را نیز بشناسد. این فایل می‌بایست از برنامه‌ی اصلی فراخوانی نماییم.

آزمایش ۲-۶: پیاده‌سازی فرستنده مدولاسیون FSK

۱. تولید بیت: با استفاده از تابع `bit_gen` تعدادی بیت ۰ و ۱ را متناسب با طول بسته‌ی داده و با در نظر گرفتن پارامترهای مدولاسیون 4FSK تولید نموده و آن را درون ماتریس `b_tx` ذخیره نماییم. ماتریس `b_tx` یک ماتریس با اندازه‌ی $\text{pkt_size} \times k$ می‌باشد.

۲. نگاشت بیت به شماره‌ی سمبل: بردار جدیدی متناظر با ماتریس `b_tx` و با نام `sym_idx` تولید نماییم و متناظر با هر سطر ماتریس `b_tx` یک عدد صحیح معادل با آن بیت‌ها در `sym_idx` قرار دهیم. دقت نماییم در مدولاسیون FSK نیازی به کدگذاری گری نیست. بنابراین در برنامه از پرچم `flag_gray_encode` برای اعمال یا عدم اعمال کدگذاری گری استفاده نماییم.

۳. تولید نمونه‌های زمانی سمبل ارسالی: با استفاده از تابع `pulse_modulation` ابتدا نمونه‌های زمانی متناظر با هر شماره‌ی سمبل را تولید و در نهایت نمونه‌های زمانی یک بسته‌ی داده را آماده‌ی ارسال کرده و حاصل را در بردار `tx_smp1` ذخیره نماییم. دقت نماییم که انرژی هر سمبل برابر با $E_s = 1$ باشد تا انرژی متوسط سمبل‌ها نیز برابر با $E_{s,avg} = 1$ شود. توضیحات تکمیلی: در این آزمایش نمونه‌های سیگنال در باندپایه ایجاد می‌شود. چون سیگنال در باند میانی تولید نمی‌شود نیازی به دانستن f_0 نیست. از این رو می‌بایست از فرمول معادل باندپایه‌ی سیگنال مدولاسیون FSK استفاده نمود. فرمول‌هایی که در بخش توضیحات این آزمایش نوشته شده مربوط به حالت سیگنال پیوسته است. در حالت گسسته ابتدا یک بردار زمان به صورت $t = (0:\text{smp1_per_sybl}-1) * t_s$ ایجاد نمود که در آن $t_s = 1/f_s$ زمان نمونه‌برداری می‌باشد. سپس با استفاده از این بردار که در واقع نقاط نمونه‌برداری از معادل پایین‌گذر است، شکل گسسته‌ی هر کدام از سمبل‌ها به دست می‌آید. ضریب مناسبی به جای $\sqrt{\frac{2E_s}{T}}$ می‌بایست گذاشته شود تا انرژی هر سمبل واحد شود. حال نمونه‌های سمبل‌های مختلف را پشت سر هم قرار می‌دهیم. در واقع اگر تعداد سمبل‌ها برابر ۱۰ باشد و `smp1_per_sybl` برابر

با باشد، بردار `tx_smpl` یک بردار با ۶۴۰ سطر و یک ستون خواهد بود. با استفاده از یک حلقه `for` می‌توان پس از تولید هر یک از سمبل‌ها آن را در موقعیت مناسب در بردار `tx_smpl` قرار داد.

آزمایش ۳-۶: مدل‌سازی کانال

۱. افزودن تأخیر در کانال: برای شبیه‌سازی تأخیر در کانال به اندازه‌ی پارامتر `chnl_delay_in_smpl` به ابتدای بردار `tx_smpl` صفر اضافه نمایید. حاصل را درون بردار `tx_smpl_delayed` قرار دهید. دقت نمایید که طول بردار افزایش می‌یابد و می‌بایست در ادامه‌ی برنامه اثر این افزایش لحاظ شود. در این جا فعلاً مقدار تأخیر برابر با صفر قرار دهید.
 ۲. اعمال اختلاف فاز در کانال: برای اعمال اختلاف فاز در کانال می‌بایست بردار `tx_smpl_delayed` را در `exp(1i*chnl_phase_offset)` ضرب شود. حاصل را در بردار `rx_smpl` قرار دهید. در این جا نیز آفست فاز را برابر با صفر قرار دهید.
 ۳. شبیه‌سازی کانال با نویز سفید گاوسی
- گام ۱. تعیین واریانس نویز بر اساس نسبت سیگنال به نویز (E_b/N_0): با توجه به تولید هر سمبل با انرژی واحد، مقدار انرژی متوسط بیت را به دست آورده و برابر متغیر `Eb` قرار دهید. مقدار E_b/N_0 را برابر با 10dB در نظر بگیرید (به عبارتی پارامترهای `snr_min` و `snr_max` برابر 10 تنظیم می‌شوند). سپس واریانس نویز را بر اساس نسبت E_b/N_0 به دست آورید و آن را درون متغیر `var_noise` قرار دهید.
- گام ۲. افزودن نویز به سیگنال: با استفاده از تابع `randn` یک بردار نویز مختلط با واریانس `var_noise` و ابعاد برابر با `tx_smpl` تولید نمایید و آن را `noise_smpl` نامگذاری نمایید. سپس این بردار را با بردار `rx_smpl` جمع نموده و حاصل را `rx_smpl_noise` بنامید.

آزمایش ۴-۶: پیاده‌سازی گیرنده‌ی مدولاسیون FSK

- در این جا می‌خواهیم گیرنده‌ی مدولاسیون FSK خطی را پیاده‌سازی نماییم.
۱. دمدولاسیون یا آشکارسازی پالس: نمونه‌های ارسال شده به همراه نویز (`tx_smpl_noise`) را به عنوان نمونه‌های دریافتی درون بردار `rx_smpl` قرار دهید. با استفاده از تابع `pulse_demodulation` با ورودی‌های نمونه‌های سیگنال دریافتی (`rx_smpl`)، نام مدولاسیون (`modulation`)، مرتبه‌ی مدولاسیون (`M`)، نرخ نمونه‌برداری (`fs`)، تعداد نمونه‌های هر سمبل (`smpl_per_symbol`)، نام تابع شکل‌دهنده‌ی پالس (`pulse_name`) و روش تشخیص قدرت سیگنال دریافتی در راستای سیگنال‌های پایه (`mode`)، خروجی سمبل‌های خروجی (`rx_sym`) و اندیس سمبل‌های آشکار شده (`det_sym_idx`) را به دست آورید. (در این جا به پارامترهای ورودی `mode`، `pulse_name` نیازی نیست).
 ۲. محاسبه‌ی خطای سمبل: با مقایسه‌ی تعداد اختلاف‌های بردارهای `det_sym_idx` و `sym_idx` و محاسبه نسبت اختلاف این دو بردار به تعداد کل سمبل‌ها، خطای سمبل را به دست آورده و آن را در متغیر `ser` قرار دهید.
 ۳. تبدیل سمبل به بیت و محاسبه‌ی خطای بیت: بیت‌های متناظر با سطرهای `det_sym_idx` را به دست آورید و آن را درون متغیر `det_bit` قرار دهید. با مقایسه‌ی تعداد اختلاف ماتریس‌های `det_bit` و `b_tx` و محاسبه نسبت تعداد اختلاف این دو ماتریس به تعداد کل بیت‌ها، خطای بیت را به دست آورده و آن را در متغیر `ber` قرار دهید.

آزمایش ۵-۶: خواسته‌های کلی

۱. رسم نمودار نرخ خطای بیت: با تغییر نسبت سیگنال به نویز بین 0 تا 10dB نرخ خطای بیت را برای مدولاسیون‌های 2FSK، 4FSK و 8FSK را به دست آورده و در نمودارهایی جداگانه رسم نمایید و آن را با خروجی تابع `berawgn` مقایسه نمایید.

۲. اثر تأخیر در کانال:

در این جا فرض نمایید مدولاسیون 2FSK است و نسبت سیگنال به نویز (E_b/N_0) را از 0dB تا 10dB تغییر دهید.

گام ۱. اثر تأخیر در کانال بر روی نرخ خطای بیت و منظومه‌ی سیگنالی: تأخیر کانال را برابر با گرد شده‌ی 0.1، 0.5 و 0.8 برابر طول یک سمبل قرار دهید (یک عدد صحیح معادل تعداد نمونه‌های زمانی تأخیر یافته). نمودار نرخ خطای بیت را به دست آورد و با حالت بدون تأخیر مقایسه نمایید. منظومه سیگنالی هر دو حالت فوق را نیز با یکدیگر مقایسه نمایید. علت مشاهدات خود را توضیح دهید.

گام ۲. اضافه کردن هدر: برای یافتن نقطه‌ی شروع درست داده‌ها، تعدادی داده‌ی مشخص به عنوان هدر در ابتدای رشته بیت‌های ارسالی قرار دهید. نمونه‌های زمانی مربوط به هدر را از بردار `tx_smp1` جدا کرده و در متغیر `hdr_smp1` ذخیره نمایید. بیت‌های هدر می‌بایست به صورت زیر باشد. این رشته بیت خواص خودهمبستگی خوبی دارد.

```
hdr_bit = repmat(reshape(de2bi(hex2dec(['1C6387FF5DA4FA325C895958DC5'])))', [1, 1], 1, k);
```

گام ۳. تخمین زمان شروع بسته و جبران اثر تأخیر کانال: با استفاده از همبستگی بردار `tx_smp1_noise` و نمونه‌های زمانی هدر `hdr_smp1`، نقطه‌ی شروع بسته‌ی داده را تخمین بزنید. با مقایسه‌ی نرخ خطای بیت حالت بدون تأخیر و حالتی که نقطه‌ی شروع داده‌ها تخمین زده می‌شود، از عملکرد برنامه‌ی خود اطمینان حاصل نمایید.

۳. اعمال فاز در کانال و تخمین فاز:

در این جا فرض نمایید مدولاسیون 2FSK است و نسبت سیگنال به نویز (E_b/N_0) را از 0dB تا 10dB تغییر دهید.

گام ۱. اثر فاز اضافی کانال بر نرخ خطای بیت: اختلاف فاز کانال را برابر ۳۰ درجه قرار دهید. ابتدا نمودار نرخ خطای بیت را به دست آورده و با حالت بدون اختلاف فاز مقایسه نمایید. منظومه سیگنالی هر دو حالت فوق را نیز با یکدیگر مقایسه نمایید. علت مشاهدات خود را توضیح دهید.

گام ۲. تخمین فاز اضافی کانال: اگر به ابتدای بسته‌ی ارسالی هدر بخش قبل اضافه شود، با استفاده از همبستگی بردار `rx_smp1_noise` و نمونه‌های زمانی هدر `hdr_smp1`، فازی را که کانال به سیگنال اضافه کرده است را به دست آورید. (راهنمایی: به فاز در قله‌ی خروجی همبستگی دقت نمایید.)

گام ۳. جبران‌سازی فاز کانال: تلاش کنید با اعمال ضریب مناسب اثر فاز اضافی کانال را جبران نمایید.
(`cmpnst_mode = 2`)

آزمایش ۶-۷: پیاده‌سازی سخت‌افزاری گیرنده و فرستنده‌ی FSK

در این بخش `flg_hrdwr_usg = 1` و `rx_alg = 1` می‌باشد.

۱. ارسال نمونه‌ها با استفاده از ADALM-PLUTO: بردار `tx_smp1` مربوط به مدولاسیون 2FSK (هدر اضافه شود) را با استفاده از دستور `transmitRepeat` به مدت زمان `stop_time` ثانیه به صورت پی‌درپی در فضا ارسال نمایید. این دستور داده‌ها را از طریق USB به رادیو نرم‌افزار ارسال می‌نماید و در حافظه‌ی سخت‌افزار ذخیره می‌نماید. فرستنده داده‌ها را از حافظه‌ی رادیو نرم‌افزار قرائت کرده و مدام ارسال می‌نماید. برای این منظور ابتدا می‌بایست پس از پیکربندی رادیو نرم‌افزار یک شیء فرستنده ایجاد نموده و فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را برابر با 0dB تنظیم نمایید.

۲. دریافت سیگنال با استفاده از ADALM-PLUTO: ابتدا شیء مربوط به گیرنده را تعریف نمایید. فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را به صورت Manual و برابر 20dB تنظیم نمایید. به علت اطمینان از دریافت یک بسته‌ی کامل، تعداد نمونه‌های زمانی دریافتی را برابر با دو برابر تعداد نمونه‌های زمانی بسته در نظر بگیرید. به منظور دریافت داده، شیء گیرنده را فراخوانی نمایید و حاصل را درون متغیر `rx_smp1` قرار دهید.

۳. طیف سیگنال ارسالی: طیف فرکانسی سیگنال **rx_smp1** دریافتی از ADALM-PLUTO را برای مدولاسیون 2FSK را رسم نماید. در مورد پهنای باند و سطح باندهای جانبی آن توضیحاتی ارائه دهید. برای این منظور از برنامه‌های قبلی استفاده نمایید.

۴. دمدولاسیون و مشاهده‌ی منظومه‌ی سیگنالی:

- گام ۱. مشابه قبل عمل دمدولاسیون را انجام داده و منظومه‌ی سیگنالی مربوط به **rx_sym** را رسم نمایید. منظومه‌ی سیگنالی باید بتواند به صورت به‌لحظه به‌روز شود.
- گام ۲. علت تفاوت منظومه‌ی سیگنالی در هنگام کار با سخت‌افزار را با حالت شبیه‌سازی را با مشاهده‌ی تفاوت شکل موج سیگنال دریافتی در حالت کار با سخت‌افزار و شبیه‌سازی توضیح دهید.
- گام ۳. با تغییر ویژگی‌های زیر برای سخت‌افزار مجدد مشاهدات خود را بیان نمایید. (توضیح بیشتر با جستجوی عبارت DC Offset Tracking در راهنمای نرم‌افزار MATLAB)

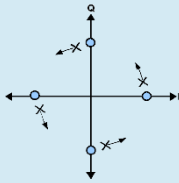
```
rx.ShowAdvancedProperties = true;
rx.EnableBasebandDCCorrection = false;
```

۵. جبران‌سازی دامنه، فاز و تأخیر و مشاهده‌ی منظومه‌ی سیگنالی: با یافتن ابتدای بسته‌ها و جبران فاز، دامنه و تأخیر، منظومه‌ی سیگنالی مربوط به **rx_sym** را رسم نمایید. احتمال خطای بیت را نیز در این حالت محاسبه نمایید.

۶. اثر Δf بر روی پهنای باند ارسالی: طیف فرکانسی سیگنال **rx_smp1** دریافتی از ADALM-PLUTO را برای مدولاسیون 4FSK رسم نماید. در مورد پهنای باند و سطح باندهای جانبی آن توضیحاتی ارائه دهید. با تغییر Δf از $\frac{1}{2T}$ به $\frac{2}{T}$ پهنای باند سیگنال دریافتی هر حالت را به دست آورده و با یکدیگر مقایسه نمایید.



- [1] J. G. Proakis and M. Salehi, *Digital Communications*. Boston: McGraw-Hill, 2008.
- [2] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.








آزمایش هفتم

آشکارسازی ناهمدوس






اهداف آزمایش

در طول آزمایش‌های قبل تمرکز بر روی آشکارسازی همدوس بوده است. در صورت وجود تأخیر کانال و تغییر فاز سیگنال دریافتی، ابتدا این فاز جبران و سپس آشکارسازی صورت می‌پذیرفت. در این آزمایش با مدولاسیون‌هایی آشنا می‌شویم که با وجود جبران نکردن فاز می‌توان عمل آشکارسازی را انجام داد که به این آشکارسازی، آشکارسازی ناهمدوس گفته می‌شود.

در انتهای این آزمایش دانشجو می‌بایست:

- با مفهوم آشکارسازی ناهمدوس آشنا شود. 
- آشکارسازی ناهمدوس مدولاسیون FSK را انجام دهد. 
- با نحوه‌ی تولید و آشکارسازی مدولاسیون DBPSK آشنا شود. 
- با شبیه‌سازی کانال با فاز تصادفی آشنا باشد. 
- این نوع آشکارسازی را با سخت‌افزار پیاده‌سازی نماید. 

ابزارهای مورد نیاز

- رایانه 
- نرم‌افزار MATLAB R2020b 
- رادیو نرم‌افزار ADALM-PLUTO 
- کابل کوکسیال 
- آنتن 



آشکارسازی ناهمدوس

سیگنال پس از عبور از کانال دچار تأخیر زمانی می‌شود. این تأخیر زمانی باعث ایجاد ابهام فاز در سیگنال دریافتی می‌گردد. به عنوان مثال، سیگنال مربوط به مدولاسیون BPSK به صورت زیر است.

$$S_m(t) = \pm g(t) \cos(2\pi f_0 t) \equiv A_m g(t) \cos(2\pi f_0 t), \quad A_m \in \{\pm 1\}$$

در این جا بُعد فضا برابر با ۱ است و پایه‌ی این فضا به صورت زیر می‌باشد.

$$\Phi(t) = \sqrt{\frac{2}{E_g}} g(t) \cos(2\pi f_0 t)$$

پس از اضافه شدن نویز، سیگنال دریافتی در گیرنده به صورت زیر خواهد بود.

$$r(t) = S_m(t) + n(t) = \pm g(t) \cos(2\pi f_0 t) + n(t)$$

بنابراین برای آشکارسازی کافی است که در هر بازه‌ی T_s ، ضرب داخلی $\phi(t)$ و $r(t)$ را محاسبه شده و با توجه به علامت آن تصمیم‌گیری انجام شود. این نحوه‌ی تصمیم‌گیری در ادامه آمده است.

$$\hat{A}_m = \text{sgn}(\langle r(t), \Phi(t) \rangle) = \text{sgn}\left(\int r(t)\Phi(t)dt\right)$$

در بسیاری از موارد، سیگنال ارسالی پس از عبور از کانال دچار تأخیر زمانی می‌شود و سیگنال دریافتی در گیرنده به صورت زیر خواهد بود.

$$r(t) = S_m(t - \tau) + n(t) = \pm g(t) \cos(2\pi f_0 t + \phi) + n(t)$$

در رابطه‌ی فوق $\phi = -2\pi f_0 \tau$ می‌باشد.

معمولاً این تأخیر، در مقایسه با پهنای پالس ارسالی ناچیز است. اما تأثیر این تأخیر بر روی فاز کاملاً قابل ملاحظه است. در صورتی که از تأثیر این تأخیر زمانی بر روی $g(t)$ صرف نظر کنیم، سیگنال پایه‌ی فضا به صورت زیر خواهد بود.

$$\Phi(t) = \sqrt{\frac{2}{E_g}} g(t) \cos(2\pi f_0 t + \phi)$$

بنابراین برای تشکیل بردار پایه می‌بایست مقدار ϕ را داشته باشیم که نیاز به تخمین τ دارد. فرض کنید که مقدار $\tau = 10^{-5} \text{ s}$

باشد. فرآیند تخمین زدن همواره با خطا همراه است و این خطا را به عنوان مثال ۱٪ در نظر می‌گیریم. بنابراین مقدار τ تخمین زده شده برابر است با

$$\hat{\tau} = 10^{-5} \pm 10^{-7} \text{ s}$$

فرکانس حامل (f_0) در سامانه‌های مختلف، مقادیر متفاوتی دارد. به عنوان مثال در شبکه‌ی سلولی مقدار فرکانس حامل از مرتبه‌ی

$10^9 \text{ Hz} \equiv 1 \text{ GHz}$ می‌باشد. بدین ترتیب ϕ تخمین زده شده در گیرنده به صورت زیر خواهد بود.

$$\hat{\phi} = -2\pi f_0 \hat{\tau} = -2\pi f_0 (10^{-5} \pm 10^{-7}) = \phi \pm 2\pi \times 10^2$$

همان گونه که ملاحظه می‌شود، با وجود خطای بسیار کوچک در تخمین، خطای قابل ملاحظه‌ای در ϕ حاصل خواهد شد.

در چنین مواردی بهتر است به جای تخمین پارامتر مبهم، آن را به صورت یک پارامتر غیر یقینی وارد محاسبات کرده و آن را در

تصمیم‌گیری وارد نمود. برای مثال می‌توان ϕ را به عنوان یک متغیر تصادفی با توزیع یکنواخت بین $[0, 2\pi]$ در نظر گرفت و تصمیم‌گیری بهینه را انجام داد. به چنین فرآیندی آشکارسازی ناهمدوس گفته می‌شود.

فرض کنید که مجموعه سیگنال‌های ارسالی به صورت $S_m(t) = g_m(t) \cos(2\pi f_0 t)$ باشد. $S_m(t, \phi)$ را به صورت زیر تعریف

می‌کنیم

$$S_m(t, \phi) \triangleq g_m(t) \cos(2\pi f_0 t + \phi)$$

معادل باند پایه سیگنال $S_m(t)$ را با $S_{\ell, m}(t)$ نشان می‌دهیم. بنابراین:

$$S_m(t) = \Re\{S_{\ell, m}(t)e^{j2\pi f_0 t}\} \leftrightarrow S_m(t, \phi) = \Re\{S_{\ell, m}(t)e^{j2\pi f_0 t + j\phi}\}$$

معادل باند پایه را برای این سیستم به صورت زیر در نظر می‌گیریم:

$$r_l(t) = S_{\ell, m}(t)e^{j\phi} + n_l(t)$$

با فرض اینکه ϕ یک متغیر تصادفی با توزیع یکنواخت بین $[0, 2\pi]$ باشد، خواهیم داشت.



$$\hat{m} = \arg \max \left(\pi_m e^{\frac{-E_{\ell,m}}{2N_0}} I_0 \left(\frac{|r_{\ell} \cdot \underline{S}_{\ell,m}|}{N_0} \right) \right)$$

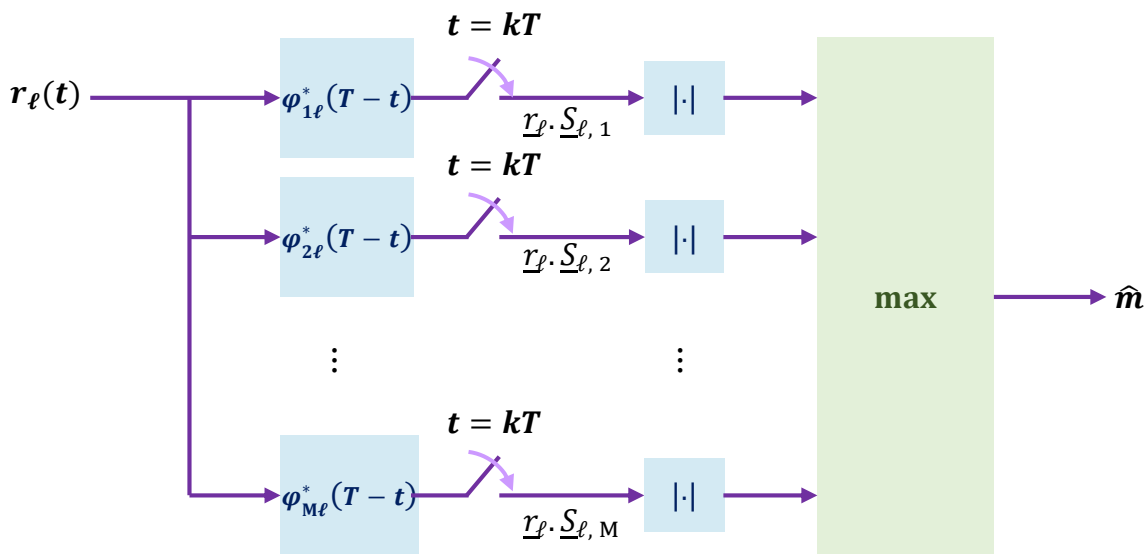
در رابطه‌ی فوق r_{ℓ} معادل برداری سیگنال دریافتی در باند پایه، $\underline{S}_{\ell,m}$ معادل برداری سیگنال ارسالی m ام در باند پایه و π_m احتمال ارسال سمبل m ام می‌باشد. $I_0(x)$ نیز تابع بسل اصلاح شده مرتبه 0 است. این تابع نسبت به x متقارن است و به ازای مقادیر مثبت x صعودی نیز می‌باشد. عملگر (\cdot) معادل ضرب داخلی دو بردار است که به صورت زیر تعریف می‌شود.

$$\underline{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \underline{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \rightarrow \underline{X} \cdot \underline{Y} = \underline{Y}^H \underline{X}$$

بدین ترتیب اگر احتمال ارسال همه سمبل‌ها برابر و انرژی سیگنال ارسالی متناظر با هر کدام از آن‌ها یکسان باشد، عبارت فوق به صورت زیر ساده خواهد شد

$$\hat{m} = \arg \max (|r_{\ell} \cdot \underline{S}_{\ell,m}|)$$

گیرنده متناظر با این تصمیم‌گیری به صورت زیر خواهد بود.

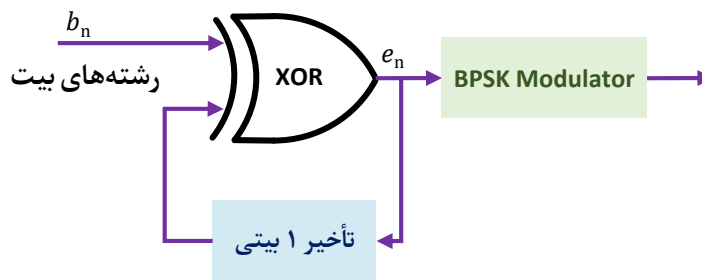


شکل 11 شمای گیرنده‌ی ناهمدوس

مدولاسیون DPSK (Differential PSK)

سیگنال ارسالی در مدولاسیون PSK در حالت کلی به صورت $g(t) \cos \left(2\pi f_0 t + \frac{2\pi}{M} (m-1) \right)$ است. بنابراین تأخیر زمانی ایجاد شده به دلیل غیرایده‌آل بودن کانال، باعث می‌شود که سیگنال دریافتی به صورت $g(t) \cos \left(2\pi f_0 t + \frac{2\pi}{M} (m-1) + \phi \right)$ باشد. بنابراین نیاز است که آشکارسازی به صورت ناهمدوس صورت بگیرد. برای سادگی کار فرض می‌کنیم که مدولاسیون به ازای $M = 2$ (BPSK) انجام پذیرفته باشد. احتمال خطای آشکارسازی ناهمدوس برای مدولاسیون BPSK برابر $\frac{1}{2}$ است. این احتمال خطا نشان‌دهنده این واقعیت است که در صورت وقوع ابهام فاز در مدولاسیون BPSK، نمی‌توان در مورد سمبل ارسالی تصمیم‌گیری نمود و این مدولاسیون عملاً ناکارآمد خواهد شد.

برای مقابله با چنین اتفاقی، مدولاسیون DPSK (در حالت خاص DBPSK) پیشنهاد می‌شود. سمبل‌های ارسالی ما به صورت 0 و 1 (± 1) خواهد بود. در این مدولاسیون، به ازای وقوع هر 1، فاز سیگنال ارسالی π رادیان و به ازای وقوع هر 0، فاز سیگنال ارسالی 0 رادیان تغییر خواهد کرد. بدین ترتیب در گیرنده برای آشکارسازی می‌بایست اختلاف فاز سیگنال فعلی را با سیگنال قبل به دست آوریم و تصمیم‌گیری انجام دهیم. این کار باعث می‌شود که ابهام ϕ در فاز حذف شود و عملکرد مدولاسیون BPSK بهبود یابد. همان‌گونه که ملاحظه می‌شود، این مدولاسیون یک مدولاسیون حافظه‌دار است که برای تصمیم‌گیری برای سمبل فعلی، به سمبل قبلی نیاز خواهد بود. در عمل می‌توان برای پیاده‌سازی مدولاتور این مدولاسیون به صورت زیر عمل کرد.

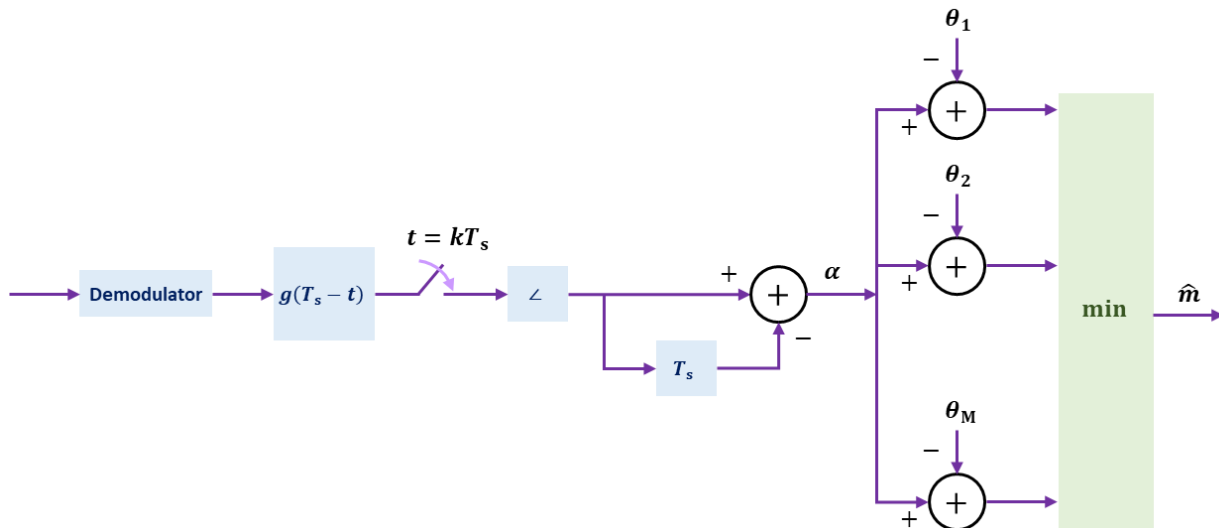


شکل 12 شمای مدولاسیون DPSK

بنابراین برای پیاده‌سازی مدولاتور DBPSK، ابتدا رشته بیت ورودی (b_n) را به صورت زیر کدگذاری می‌کنیم.

$$e_k = e_{k-1} \oplus b_k$$

سپس با توجه به رشته بیت تولید شده، به ازای هر 0، فاز 0 و هر 1، فاز π را در نظر گرفت. دمدولاتور DPSK نیز به صورت زیر می‌باشد.



شکل 13 شمای گیرنده‌ی مدولاسیون DPSK

که برای حالت خاص $m = 2$ می‌توان به صورت زیر تصمیم‌گیری نمود

$$\Re\{r_{\ell,k} r_{\ell,k-1}^*\} > 0 \rightarrow \hat{b}_k = 0$$

$$\Re\{r_{\ell,k} r_{\ell,k-1}^*\} < 0 \rightarrow \hat{b}_k = 1$$

$r_{\ell,k}$ معادل برداری سیگنال پایه دریافتی در k امین بازه‌ی سمبل (T_s) است.

شرح آزمایش

آزمایش ۱-۷: مقداردهی‌های اولیه

در این آزمایش بنا داریم تعدادی بسته‌ی داده را برای ارسال از طریق مدولاسیون FSK ناهمدوس و DBPSK آماده نماییم. از آن جا که در این آزمایش بناست این فرستنده با استفاده از رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی شود، می‌بایست با استفاده از پارامترها و ملاحظات عملی، این کار انجام شود.

پارامترهایی که در این پیاده‌سازی اهمیت دارد شامل نرخ نمونه‌برداری (f_s)، تعداد نمونه‌های هر سمبل (smpl_per_syml)، تعداد سمبل‌های ارسالی در بسته‌ی داده (pkt_size) و مدت زمان ارسال داده (stop_time) می‌باشد. در این آزمایش مقدار پارامترهای مورد نیاز به صورت جدول ۵ می‌باشد.

جدول ۶ پارامترهای آزمایش ۷

| پارامتر | f_s | smpl_per_syml | pkt_size | stop_time |
|---------|-------|--------------------------|--------------------------------------|---------------------|
| مقدار | 10MHz | FSK: 64 DBPSK: 8 | شبیه‌سازی: ۱۰۰۰۰۰ سخت‌افزار: ۱۰۰۰ | ۱۰۰ ثانیه |

دقت نظر داشته باشید که در این آزمایش نیز تنها می‌خواهیم برنامه‌هایی که در آزمایش قبل نوشته شده است را تکمیل نماییم.

۱. تولید M-File مربوط به مقدار دهی اولیه: قالب کلی پارامترهای مورد نیاز در این آزمایش، مانند آزمایش قبل است و فایل `dcl_init.m` می‌بایست پارامتری به نام `mod_det_opt` داشته باشد که نوع آشکارسازی را که می‌تواند همدوس (`coherent`) یا ناهمدوس (`noncoherent`) باشد را مشخص می‌نماید. برنامه با توجه به این گزینه نوع گیرنده و نحوه‌ی ارسال فرستنده مدیریت می‌شود. هم‌چنین پارامتر `phase_amb_opt` برای نوع اعمال ابهام فاز در نظر گرفته شود. این فایل می‌بایست از برنامه‌ی اصلی فراخوانی نمایید.

آزمایش ۲-۷: پیاده‌سازی مدولاسیون FSK ناهمدوس

۱. شبیه‌سازی کانال با فاز تصادفی: برای شبیه‌سازی کانال با فاز تصادفی می‌بایست بردار `tx_smpl_delayed` را در `exp(1i*chnl_phase_offset_amb)` ضرب شود. حاصل را در بردار `tx_smpl` قرار دهید. `chnl_phase_offset_amb` را به صورت یک بردار تصادفی بین ۰ و 2π تولید کنید. طول این بردار برابر با طول بردار `tx_smpl` می‌باشد. دقت کنید که فاز اضافه شده در طول هر سمبل ثابت باشد. در این حالت `phase_amb_opt` برابر با ۰ است. (راهنمایی: از دستور `kron` می‌توان برای این کار استفاده نمود.)

۲. شبیه‌سازی مدولاسیون FSK ناهمدوس: برای پیاده‌سازی این مدولاسیون، همانند FSK همدوس که در آزمایش قبل پیاده شد عمل نمایید. نمودار احتمال خطای بیت را برای مدولاسیون ۴FSK و ۲FSK به دست آورده و نتیجه‌ی را با دستور `berawgn` صحت‌سنجی نمایید. دقت نمایید در حالت آشکارسازی ناهمدوس باید حداقل فاصله‌ی فرکانسی $1/T_s$ باشد. آشکارساز این مدولاسیون نیز به صورت $\arg \max |r_{\ell}, S_{\ell,m}|$ می‌باشد.

۳. پیاده‌سازی سخت‌افزاری مدولاسیون FSK ناهمدوس: با فرض این که توان فرستنده‌ی ADALM-PLUTO برابر با ۰dBm و بهره‌ی گیرنده برابر با ۲۰dB باشد و از آنتن به منظور ارتباط فرستنده و گیرنده استفاده می‌شود، خطای آشکارسازی با استفاده از آشکارسازی ناهمدوس را برای مدولاسیون ۲FSK محاسبه نمایید. هم‌چنین در این حالت منظومه‌ی سیگنالی این مدولاسیون را نیز رسم نمایید. در این جا از هدر تنها برای به دست آوردن ابتدای بسته‌ی ارسالی استفاده می‌شود و هیچ گونه جبران‌سازی فازی صورت نمی‌پذیرد.

آزمایش ۳-۷: پیاده‌سازی مدولاسیون DBPSK

۱. تولید بیت: با استفاده از تابع `bit_gen` تعدادی بیت ۰ و ۱ را متناسب با طول بسته‌ی داده و با در نظر گرفتن پارامترهای مدولاسیون DBPSK تولید نموده و آن را درون ماتریس `b_tx` ذخیره نمایید. ماتریس `b_tx` یک ماتریس با اندازه‌ی $\text{pkt_size} \times 1$ می‌باشد.
۲. کد کردن بیت‌ها: با استفاده از شکل 12 رشته بیت‌های e_n را تولید نمایید و آن را درون بردار `enc_b_tx` قرار دهید.
۳. نگاشت بیت به سمبل:
 - گام ۱. کدگذاری گری بیت‌های تولیدی: با وجود این که در این جا نیازی به کدگذاری گری نیست ولی به منظور کلی بودن برنامه این مرحله نیز انجام می‌شود. با استفاده از تابع `gray_code`، ماتریس مربوط به کدگذاری گری مدولاسیون DBPSK را تولید کنید و آن را در ماتریس `b_gray` ذخیره کنید. بردار جدیدی متناظر با ماتریس `enc_b_tx` و با نام `sym_idx` تولید نمایید و نگاشتی یک‌به‌یک بین سطرهاى ماتریس `enc_b_tx` و ماتریس `b_gray` برقرار کنید. به عبارتی سطر i ام بردار `sym_idx`، شماره‌ی سطری از `b_gray` را نشان می‌دهد که برابر با سطر i ام ماتریس `enc_b_tx` باشد.
 - گام ۲. تولید سمبل‌های ارسالی: با استفاده از تابع `constellation`، تمامی سمبل‌های ارسالی مدولاسیون DBPSK را تولید کرده و سمبل‌های متناظر با هر سطر بردار `sym_idx` را تولید کنید. این سمبل‌ها را در بردار `mod_sym` ذخیره نمایید.
۴. شکل‌دهی پالس ارسالی: با استفاده از تابع `pulse_modulation` نمونه‌های ارسالی مربوط به یک بسته‌ی داده را آماده‌ی ارسال کرده و حاصل را در بردار `tx_smp1` ذخیره نمایید. تابع شکل‌دهی پالس را مثلثی در نظر بگیرید.

آزمایش ۴-۷: مدل‌سازی کانال

۱. افزودن تأخیر در کانال: برای شبیه‌سازی تأخیر در کانال به اندازه‌ی پارامتر `chnl_delay_in_smp1` به ابتدای بردار `tx_smp1` صفر اضافه نمایید. حاصل را درون بردار `tx_smp1_delayed` قرار دهید. دقت نمایید که طول بردار افزایش می‌یابد و می‌بایست در ادامه‌ی برنامه اثر این افزایش لحاظ شود. در این جا فعلاً مقدار تأخیر برابر با صفر قرار دهید.
۲. اعمال ابهام فاز کانال: مدولاسیون DPSK به تغییرات شدید فاز حساس است و بنابراین باید ابهام فاز ایجاد شده خیلی زیاد نباشد. ابهام فاز ایجاد شده برای سمبل‌ها را به صورت زیر مدل می‌کنیم

$$\text{chnl_phase_offset_amb} = 0 : \text{phase_step} : (\text{pkt_size}-1) * \text{phase_step}$$
 پارامتر `phase_step`، مقدار ابهام فاز ایجاد شده برای یک سمبل نسبت به سمبل قبلی را مشخص می‌کند. حال با استفاده از دستور `kron`، فاز تولید شده را برای کل نمونه‌های هر سمبل تکرار کنید. با ضرب کردن عبارت `exp(1j * chnl_phase_offset_amb)` در `tx_smp1_delayed`، ابهام فاز ایجاد شده را به سینگال ارسالی اضافه نمایید و حاصل را در `tx_smp1` ذخیره نمایید. در این جا `phase_step` را برابر با ۱۰ درجه قرار دهید.
۳. شبیه‌سازی کانال با نویز سفید گاوسی
 - گام ۳. تعیین واریانس نویز بر اساس نسبت سیگنال به نویز (E_b/N_0): ابتدا بر اساس خروجی تابع `constellation`، مقدار متوسط انرژی سمبل را به دست آورید (`Es_avg`). حال با استفاده از متوسط انرژی سمبل، مقدار انرژی متوسط بیت را به دست آورده و برابر متغیر `Eb` قرار دهید. مقدار E_b/N_0 را برابر با 10dB در نظر بگیرید (به عبارتی پارامترهای `snr_min` و `snr_max` برابر 10 تنظیم می‌شوند). سپس واریانس نویز را بر اساس نسبت E_b/N_0 به دست آورید و آن را درون متغیر `var_noise` قرار دهید.

گام ۴. افزودن نویز به سیگنال: با استفاده از تابع `randn` یک بردار نویز مختلط با واریانس `var_noise` و ابعاد برابر با `tx_smp1` تولید نمایید و آن را `noise_smp1` نامگذاری نمایید. سپس این بردار را با بردار `tx_smp1` جمع نموده و حاصل را `tx_smp1_noise` بنامید.

آزمایش ۵-۷: پیاده‌سازی گیرنده‌ی مدولاسیون DBPSK

- در ادامه‌ی فایل آزمایش قبل می‌خواهیم گیرنده‌ی مدولاسیون DBPSK را پیاده‌سازی نماییم.
۱. دمدولاسیون یا آشکارسازی پالس: نمونه‌های ارسال شده به همراه نویز (`tx_smp1_noise`) را به عنوان نمونه‌های دریافتی درون بردار `rx_smp1` قرار دهید. با استفاده از تابع `pulse_demodulation` با ورودی‌های نمونه‌های سیگنال دریافتی (`rx_smp1`)، نام مدولاسیون (`modulation`)، مرتبه‌ی مدولاسیون (`M`)، نرخ نمونه‌برداری (`fs`)، تعداد نمونه‌های هر سمبل (`smp1_per_symb1`)، نام تابع شکل‌دهنده‌ی پالس (`pulse_name`)، روش تشخیص قدرت سیگنال دریافتی در راستای سیگنال‌های پایه (`mode`) و شیوه‌ی آشکارسازی (`det_opt`)، سمبل‌های دریافتی (`rx_sym`) و اندیس سمبل‌های آشکارشده (`det_sym_idx`) را به دست آورید.
 ۲. محاسبه‌ی خطای سمبل: با مقایسه‌ی تعداد اختلاف‌های بردارهای `det_sym_idx` و `sym_idx` و محاسبه نسبت اختلاف این دو بردار به تعداد کل سمبل‌ها، خطای سمبل را به دست آورده و آن را در متغیر `ser` قرار دهید.
 ۳. تبدیل سمبل و بیت و تبدیل از کدگذاری گری به کدگذاری باینری: بیت‌های متناظر با سطرهای `det_sym_idx` را به دست آورید و آن را درون متغیر `det_bit_gray` قرار دهید. با وجود این که در این جا عکس تبدیل گری مطرح نیست به منظور کلی ماندن برنامه بیت‌های گری را به کدگذاری باینری تبدیل نماید و آن را درون بردار `det_bit` قرار دهید.
 ۴. محاسبه‌ی خطای بیت: با مقایسه‌ی تعداد اختلاف ماتریس‌های `det_bit` و `b_tx` و محاسبه نسبت تعداد اختلاف این دو ماتریس به تعداد کل بیت‌ها، خطای بیت را به دست آورده و آن را در متغیر `ber` قرار دهید.

آزمایش ۶-۷: خواسته‌های کلی

۱. رسم نمودار نرخ خطای بیت: با تغییر نسبت سیگنال به نویز بین 0 تا 10dB نرخ خطای بیت را برای مدولاسیون‌های DBPSK به دست آورده و در یک نمودار رسم نمایید و آن را با خروجی تابع `berawgn` مقایسه نمایید. این کار را برای شکل موج مثلی انجام دهید.
۲. اثر تغییرات فاز بر روی عملکرد: با تغییر پارامتر `phase_step`، تأثیر تغییرات فاز را بر روی عملکرد این سامانه ارزیابی نمایید.

آزمایش ۷-۷: پیاده‌سازی سخت‌افزاری مدولاسیون DBPSK

۱. ارسال نمونه‌ها با استفاده از ADALM-PLUTO: بردار `tx_smp1` مربوط به مدولاسیون DBPSK (هدر اضافه شود) را با استفاده از دستور `transmitRepeat` به مدت زمان `stop_time` ثانیه به صورت پی‌درپی در فضا ارسال نمایید. این دستور داده‌ها را از طریق USB به رادیو نرم‌افزار ارسال می‌نماید و در حافظه‌ی سخت‌افزار ذخیره می‌نماید. فرستنده داده‌ها را از حافظه‌ی رادیو نرم‌افزار قرائت کرده و مدام ارسال می‌نماید. برای این منظور ابتدا می‌بایست پس از پیکربندی رادیو نرم‌افزار یک شیء فرستنده ایجاد نموده و فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را برابر با 0dB تنظیم نمایید.
۲. دریافت سیگنال با استفاده از ADALM-PLUTO: ابتدا شیء مربوط به گیرنده را تعریف نمایید. فرکانس مرکزی آن را بر روی 2400MHz تنظیم کرده و بهره‌ی آن را به صورت Manual و برابر 20dB تنظیم نمایید. به علت اطمینان از دریافت یک بسته‌ی

کامل، تعداد نمونه‌های زمانی دریافتی را برابر با دو برابر تعداد نمونه‌های زمانی بسته در نظر بگیرید. به منظور دریافت داده، شیء گیرنده را فراخوانی نمایید و حاصل را درون متغیر `rx_smp1` قرار دهید.

۳. دمدولاسیون و مشاهده‌ی منظومه‌ی سیگنالی: مشابه قبل عمل دمدولاسیون را انجام داده و منظومه‌ی سیگنالی مربوط به `rx_sym` را رسم نمایید و مشاهدات خود را یادداشت نمایید. منظومه‌ی سیگنالی باید بتواند به صورت به‌لحظه به‌روز شود.



- [3] [1] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.








آزمایش هشتم

انتقال دیجیتال از درون کانال باند محدود AWGN






اهداف آزمایش

در طول آزمایش‌های قبل فرض بر آن بود تمرکز بر روی آشکارسازی همدوس بوده است. در صورت وجود تأخیر کانال و تغییر فاز سیگنال دریافتی، ابتدا این فاز جبران و سپس آشکارسازی صورت می‌پذیرفت. در این آزمایش با مدولاسیون‌هایی آشنا می‌شویم که با وجود جبران نکردن فاز می‌توان عمل آشکارسازی را انجام داد که به این آشکارسازی، آشکارسازی ناهمدوس گفته می‌شود.

در انتهای این آزمایش دانشجو می‌بایست:

-  با مفهوم آشکارسازی ناهمدوس آشنا شود.
-  آشکارسازی ناهمدوس مدولاسیون FSK را انجام دهد.
-  با نحوه‌ی تولید و آشکارسازی مدولاسیون DBPSK آشنا شود.
-  با شبیه‌سازی کانال با فاز تصادفی آشنا باشد.
-  این نوع آشکارسازی را با سخت‌افزار پیاده‌سازی نماید.

ابزارهای مورد نیاز

-  رایانه
-  نرم‌افزار MATLAB R2020b
-  رادیو نرم‌افزار ADALM-PLUTO
-  کابل کوکسیال
-  آنتن



تئوری

کانال باند محدود

در آزمایش‌های قبل کانال مورد استفاده دارای باند فرکانسی نامحدود فرض می‌شد. در واقع مدل کانال مورد استفاده در حوزه‌ی زمان به صورت تابع ضربه بود. در عمل کانال‌های مخابراتی دارای پاسخ زمانی غیرایده‌آل است و باعث ایجاد تداخل بین سمبلی می‌شود. تداخل بین سمبلی، شکلی از اعوجاج سیگنال است که هر سمبل با چندین سمبل دیگر تداخل پیدا می‌کند. این امر، یک اتفاق ناخواسته است که سمبل‌های قبلی به دلیل پهن شدن در حوزه‌ی زمان، هم‌چون نویز برای سمبل فعلی ظاهر می‌شوند و قابلیت اطمینان سیستم‌های مخابراتی را کاهش می‌دهند. وجود تداخل بین سمبلی در سیستم باعث ایجاد خطا در تصمیم‌گیری می‌شود. بنابراین لازم است که فیلترهای فرستنده و گیرنده را به گونه‌ای طراحی کنیم که تا حد ممکن از خطاهای ایجاد شده جلوگیری شود.

دسته‌ی مهمی از کانال‌ها غیرایده‌آل، کانال‌های باند محدود هستند که پاسخ فرکانسی آنها برای فرکانس‌های مشخصی برابر با صفر است. عبور سیگنال از این کانال‌ها منجر به حذف فرکانس‌های خارج از باند گذر کانال می‌شود. از طرف دیگر، ممکن است که فرکانس‌هایی که در محدوده‌ی باند گذر کانال هستند نیز تضعیف شوند.

کانال‌های باند محدود هم در مخابرات سیمی و هم در مخابرات بی‌سیم وجود دارند. در بسیاری از موارد، این محدود شدن باند می‌تواند عمدی باشد. به عنوان مثال اگر بنا باشد چندین کاربر به صورت همزمان از یک محیط برای انتقال داده استفاده کنند، می‌توان به عنوان نمونه باند فرکانسی کاربران را از یکدیگر ایزوله نمود. محدود بودن باند می‌تواند به دلیل مشخصات فیزیکی محیط انتقال نیز باشد. به عنوان مثال ممکن است که یک کابل دارای فرکانس قطع مشخصی باشد و اجازه‌ی ارسال سیگنال در فرکانس‌های بالاتر را ندهد. معمولاً سیستم‌های مخابراتی که دارای کانال‌های باند محدود هستند از شکل‌دهی پالس برای جلوگیری از ایجاد تداخل استفاده می‌کنند.

الگوی چشم (Eye Pattern)

خروجی فیلتر ارسالی در سامانه‌ی مخابراتی با مدولاسیون PAM، PSK یا QAM را می‌توان به صورت زیر بیان نمود.

$$v(t) = \sum_{n=-\infty}^{+\infty} a_n g_T(t - nT)$$

هم‌چنین خروجی کانال که در واقع سیگنال دریافتی در دمولاتور است نیز می‌توان به صورت زیر بیان کرد.

$$r(t) = \sum_{n=-\infty}^{+\infty} a_n h(t - nT) + n(t)$$

در رابطه‌ی فوق $h(t) = c(t) * g_T(t)$ است که در آن $c(t)$ پاسخ ضربه‌ی کانال، $g_T(t)$ پاسخ ضربه‌ی فیلتر ارسالی و $n(t)$ نیز یک تابع نمونه از یک فرآیند نویز گاوسی سفید جمع شونده می‌باشد.

برای طراحی یک فیلتر ارسالی باند محدود ابتدا طراحی در شرایطی که هیچ اعوجاج کانالی وجود ندارد صورت می‌گیرد. سپس اثر اعوجاج کانال مورد بررسی قرار می‌گیرد. از آن‌جا که $H(f) = C(f)G_T(f)$ شرط انتقال بدون اعوجاج آن است که پاسخ فرکانسی $C(f)$ کانال دارای دامنه‌ی ثابت و فازی خطی در کل پهنای باند سیگنال ارسالی باشد. به عبارتی به صورت رابطه‌ی زیر باشد.

$$C(f) = \begin{cases} C_0 e^{-j2\pi f t_0} & |f| \leq W \\ 0 & |f| > W \end{cases}$$

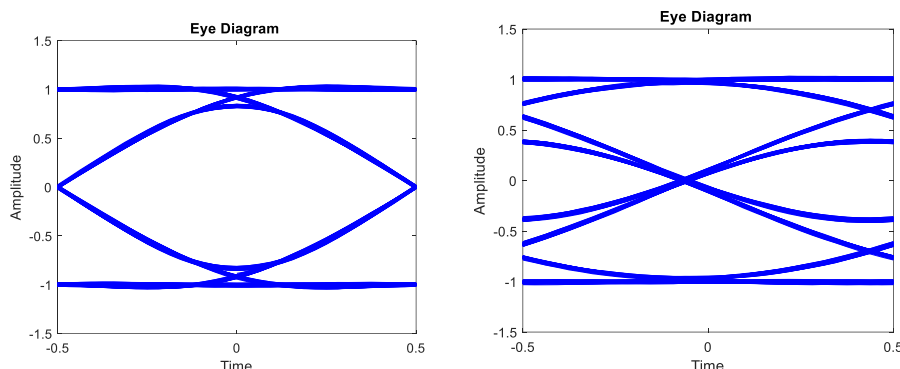
در عبارت فوق W پهنای باند در دسترس کانال، t_0 یک تأخیر دلخواه محدود و C_0 نیز یک ضریب بهره‌ی ثابت است. برای سادگی می‌توان فرض نمود $C_0 = 1$ و $t_0 = 0$ است. در شرایطی که کانال بدون اعوجاج است برای $|f| \leq W$ خواهیم داشت $H(f) = G_T(f)$ و برای $|f| > W$ صفر خواهد بود. در نتیجه فیلتر منطبق دارای پاسخ فرکانسی $H^*(f) = G_T^*(f)$ بوده و خروجی آن در زمان‌های نمونه‌برداری متناوب $t = mT$ به صورت زیر است.

$$y(mT) = x(0)a_m + \sum_{n \neq m} a_n x(mT - nT) + v(mT) \equiv y_m = x_0 a_m + \sum_{n \neq m} a_n x_{m-n} + v_m$$

در عبارت فوق $x(t) = g_T(t) * g_R(t)$ و $v(t)$ نیز خروجی فیلتر منطبق به فرآیند AWGN ورودی $n(t)$ می‌باشد.

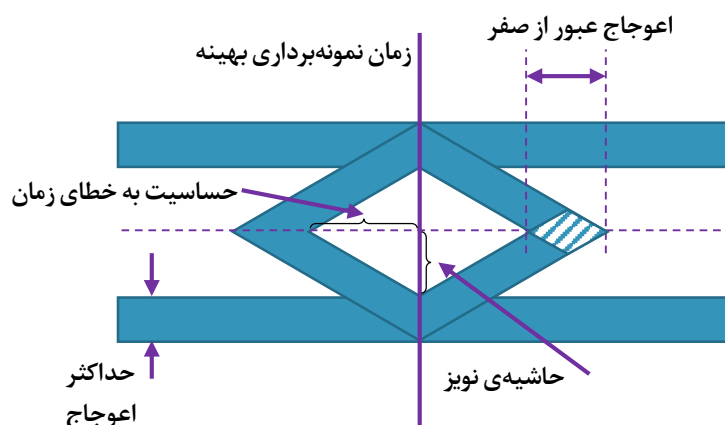


در رابطه‌ی فوق عبارت $\sum_{n \neq m} a_n x_{m-n}$ تداخل بین سمبلی (ISI) را بیان می‌کند. مقدار این تداخل بین سمبلی و نویز حاضر در سیگنال دریافتی را می‌توان بر روی یک نوسان‌نگار مشاهده نمود. برای این منظور می‌بایست سیگنال دریافتی را بر روی محور عمودی نشان داد و نرخ جاروب محور افقی را نیز برابر با $1/T$ قرار داد. نمایش حاصل بر روی نوسان‌نگار را الگوی چشم گویند که علت آن شباهتی است که با چشم انسان دارد. نمونه‌هایی از دو الگوی چشم در شکل 14 نشان داده شده است.



شکل 14 نمونه‌های از نمودار چشمی

تداخل بین سمبلی باعث بسته شدن چشم می‌شود و به عبارتی منجر به کاهش حاشیه‌ای که نویز جمع‌شونده ایجاد خطا می‌کند می‌شود. شکل 15 تأثیر تداخل بین سمبلی را بر روی کاهش باز بودن چشم نشان می‌دهد. می‌بایست توجه کرد که تداخل بین سمبلی لحظات گذر از صفر را دچار اعوجاج کرده و باعث کاهش باز بودن چشم می‌شود. در نتیجه این سامانه به خطای همگام‌سازی حساسیت بیشتری داشته و حاشیه‌ی کمتری در برابر نویز جمع‌شونده نشان می‌دهد.



شکل 15 شمای اثر تداخل بین سمبلی بر روی مقدار بازی الگوی چشم

شرح آزمایش

آزمایش ۱-۸: مقداردهی‌های اولیه

در این آزمایش بنا داریم تعدادی بسته‌ی داده را برای ارسال از طریق مدولاسیون 2PAM آماده نماییم. از آن‌جا که در این آزمایش بناست این فرستنده با استفاده از رادیو نرم‌افزار ADALM-PLUTO پیاده‌سازی شود، می‌بایست با استفاده از پارامترها و ملاحظات عملی، این کار انجام شود.

پارامترهایی که در این پیاده‌سازی اهمیت دارد شامل نرخ نمونه‌برداری (f_s)، تعداد نمونه‌های هر سمبل (smp1_per_sybl)، تعداد سمبل‌های ارسالی در بسته‌ی داده (pkt_size) و مدت زمان ارسال داده (stop_time) می‌باشد. در این آزمایش مقدار پارامترهای مورد نیاز به صورت جدول 5 می‌باشد.

جدول 7 پارامترهای آزمایش ۷

| پارامتر | f_s | smp1_per_sybl | pkt_size | stop_time |
|---------|-------|--------------------------|--------------------------------------|---------------------|
| مقدار | 10MHz | 32 | شبیه‌سازی: ۱۰۰۰۰۰ سخت‌افزار: ۱۰۰۰ | ۱۰۰ ثانیه |

دقت نظر داشته باشید که در این آزمایش نیز تنها می‌خواهیم برنامه‌هایی که در آزمایش قبل نوشته شده است را تکمیل نماییم.

آزمایش ۲-۸: پیاده‌سازی مدولاسیون 2PAM

۱. شبیه‌سازی مدولاسیون 2PAM: برای پیاده‌سازی این مدولاسیون، همانند آزمایش ۴ یا ۵ عمل نمایید. احتمال خطای بیت را برای نسبت $\frac{E_b}{N_0}$ برابر با 10dB به دست آورید. این عمل را برای دو شکل پالس مستطیلی و Root Raised Cosine ($\beta = 0.9$) و $\text{span_in_sybl} = 6$ انجام دهید. (حالت عملکردی دمدولاتور را بر روی فیلتر منطبق تنظیم نمایید).

۲. رسم نمودار چشمی: با استفاده از دستور **eyediagram** نرم‌افزار MATLAB نمودار چشمی مدولاسیون 2PAM را برای دو شکل پالس گفته شده و در دو نسبت E_b/N_0 برابر با 10dB و 40dB رسم نمایید. پارامترهای بر روی شکل 15 را به دست آورید.

آزمایش ۳-۸: مدل‌سازی کانال باندمباریک

۱. تولید یک فیلتر FIR: با استفاده از دستور **fir1** نرم‌افزار MATLAB یک فیلتر با پهنای باند 300kHz و با تعداد ۱۰۰ تپ تولید نمایید. پاسخ فرکانسی و پهنای باند نویز فیلتر را به دست آورید و با پهنای باند سیگنال ارسالی مقایسه نمایید.

۲. کانال باند محدود: پس از اعمال تأخیر و اعمال ابهام فاز، سیگنال ارسالی حاصل را با استفاده از عمل کانولوشن از فیلتر FIR مرحله‌ی قبل عبور دهید. مجدد نمودار چشمی را در دو نسبت E_b/N_0 برابر با 10dB و 40dB رسم نمایید. احتمال خطای بیت را نیز به دست آورید. این عمل برای هر دو شکل پالس گفته شده انجام شود.

آزمایش ۴-۸: پیاده‌سازی کانال باندمحدود

۱. کار با ADALM-PLUTO: موارد گفته شده را با استفاده از رادیو نرم‌افزار ADALM-PLUTO نیز انجام دهید.

آزمایش ۵-۸: خواسته‌های کلی

۱. اثر طول شکل پالس root raised cosine: با تغییر پارامتر span_in_sybl تأثیر آن را بر روی نمودار چشمی و خطای بیت مدولاسیون 2PAM با شکل پالس root raised cosine را مشاهده نمایید.



- [1] R. W. Stewart, K. Barlee, L. Crockett, and D. Atkinson, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. 2015.

