University of Utah
Summer Internship/Senior Project 2021/2022
Scott Talbot Ph.D. – May 24, 2021

Part 1

Presentation Outline

- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

Internship/Senior Project Breakdown

- The Zeta Associates sponsored project consists of two pieces
  1. Build an OFDM modem
  2. Use the OFDM modem to run tests on POWDER
- The POWDER portion will likely not happen until Fall

- OFDM – Orthogonal Frequency Division Multiplexing (Quite a mouthful. More on this later.)
- Modem – stands for modulator/demodulator, which is basically a communication device that can transmit and receive data.

Project Tools

- Hardware
  - System76 Galago Pro laptop running Pop!_OS/Ubuntu Linux
  - USRP B205mini

- Software
  - Linux
  - Python
  - ~~GNU Radio~~
  - Other packages as the need arises

Important!!!

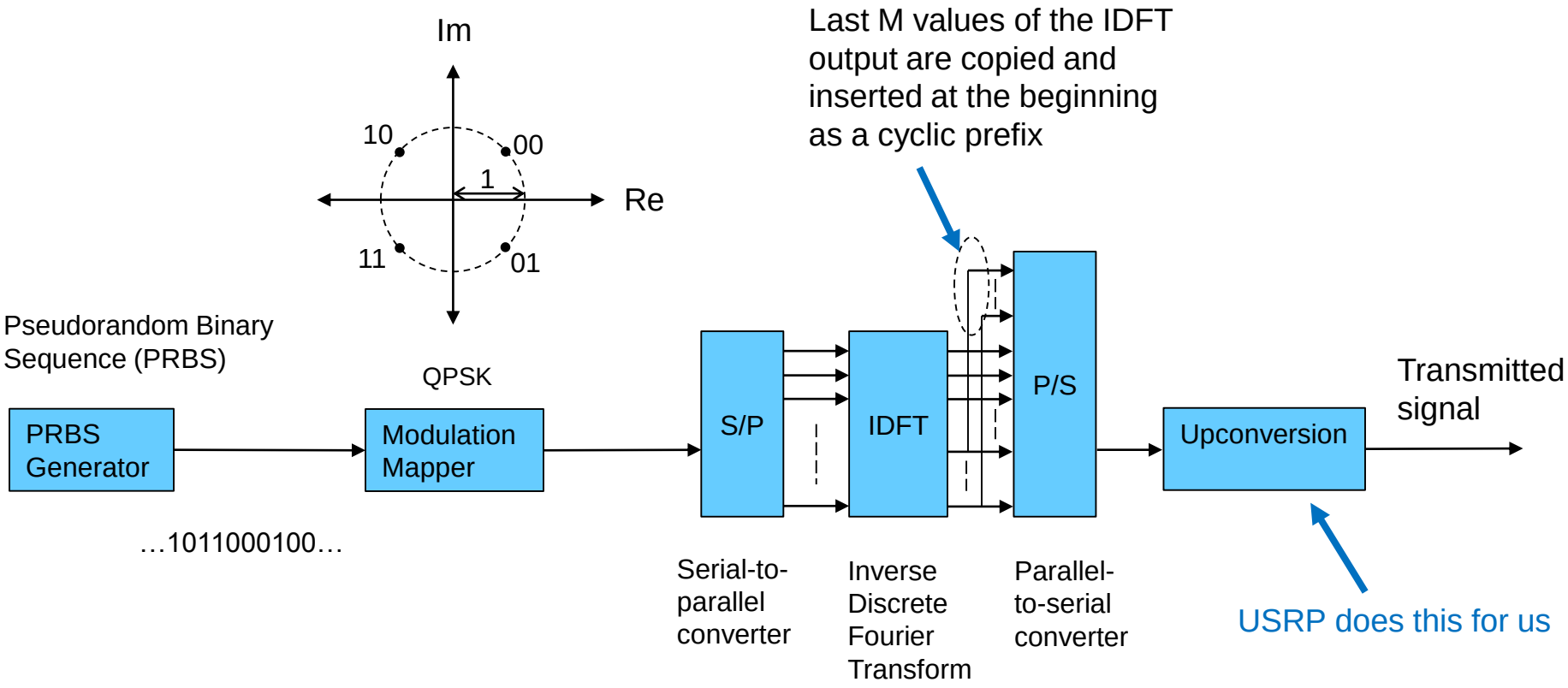
- When you get the USRP
  - It is for your convenience so that you do not have to wait for resources on POWDER
  - Don't just start transmitting stuff – In general, it's illegal
  - Don't receive certain signals – It might be illegal
    - FM/TV broadcasts are safe
  - Take care with the RF cables
    - They don't like to be bent at sharp angles

Computing Resources

- Linux:
  - You'll want to learn the command line interface (CLI)
  - https://ryanstutorials.net/linuxtutorial/
- Python:
  - https://automatetheboringstuff.com/
- ~~GNU Radio~~
  - ~~https://wiki.gnuradio.org/index.php/Tutorials~~
  - After a deep-dive into GNU Radio, I decided that we didn't need yet another framework to learn. We will control the B205-mini directly from python.
  - We may use it to build a simple spectrum analyzer
- You can always ask me

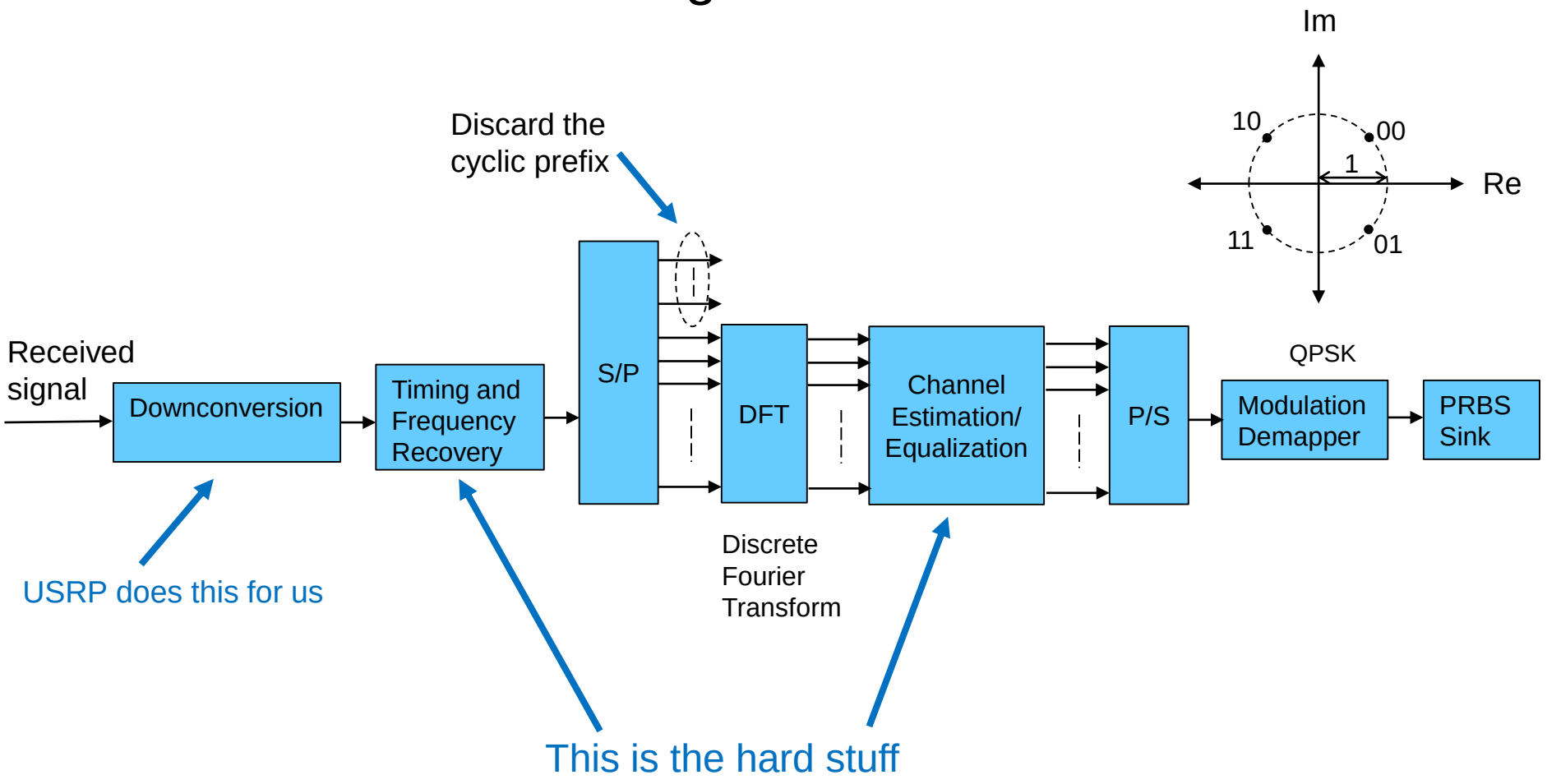
OFDM Modem - Modulator

- OFDM Modem Design



OFDM Modem - Demodulator

- OFDM Modem Design



Modem Details

- We'll go over the details as we progress
- Probable order of development
  1. PRBS generator and PRBS sink
  2. Modulation Mapper/Modulation Demapper
  3. IDFT/DFT
  4. Cyclic prefix insertion/cyclic prefix removal
  5. Timing recovery
  6. Frequency recovery
  7. Channel estimation/equalization

Presentation Outline

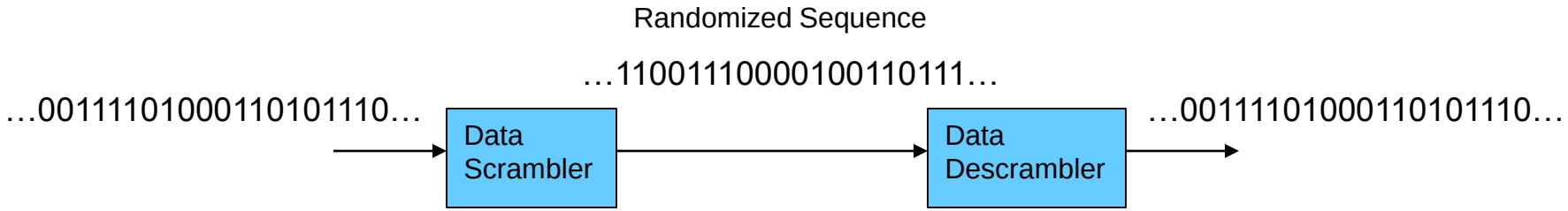
- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

PRBS

- Pseudo-random bit stream (PRBS)
  - It's hard to get machines to generate truly random numbers (or bit sequences)
  - Eventually, the numbers (or bits) repeat
- Why are we transmitting a random bit sequence?
  - It's just for testing purposes
  - We can always swap out the PRBS with a "real" data payload later

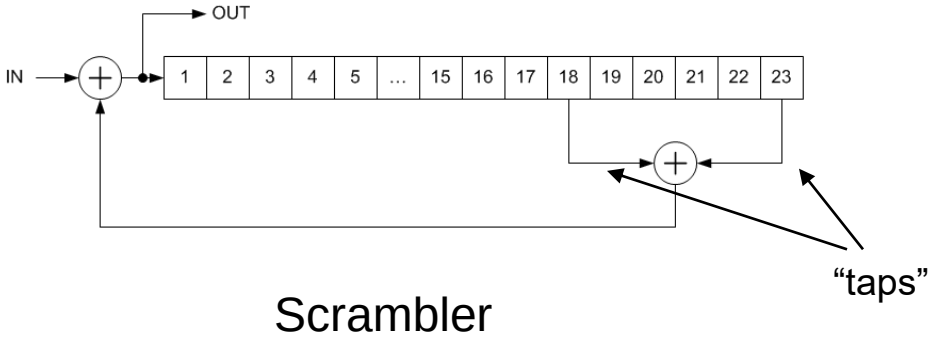
Data Scramblers

- What's a data scrambler?
  - Takes an input bit stream and randomizes it
  - The many purposes of data scramblers are beyond the scope of this presentation and/or the project
  - We are going to use data scramblers descramblers to generate our PRBS

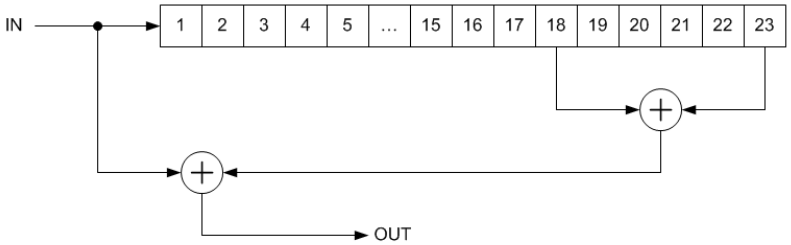


Multiplicative Data Scramblers

- Multiplicative data scramblers are interesting because the descramblers are self-synchronizing
  - The descrambler doesn't have to do anything other than take in the input and it will properly descramble the data
- The general structure is as shown here:



Scrambler

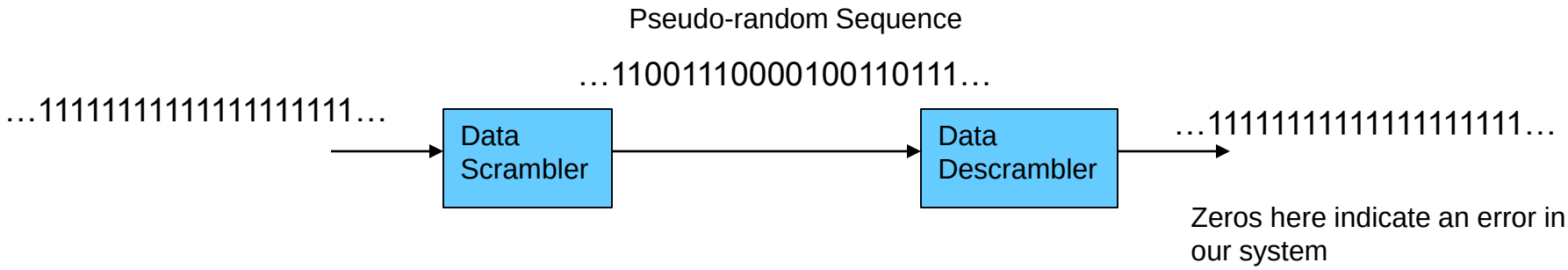


Descrambler

- See https://en.wikipedia.org/wiki/Scrambler

Scramblers/PRBS

- If we feed all ones into a scrambler with properly chosen taps, we will get a PRBS sequence that repeats every $2^N - 1$ bits, where N is the length of the scrambler shift register
- Convenient for testing a communications system, because we transmit “random” data and get all ones at the output
  - Output bits that are zeros indicate system errors



Presentation Outline

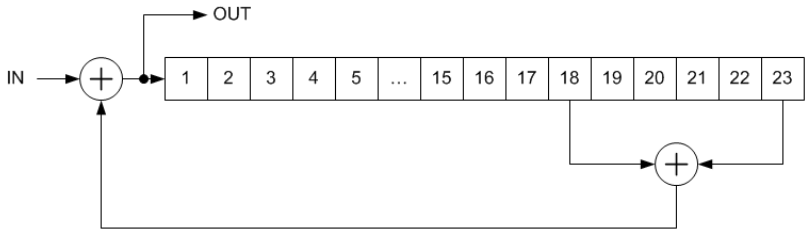
- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

Task #1

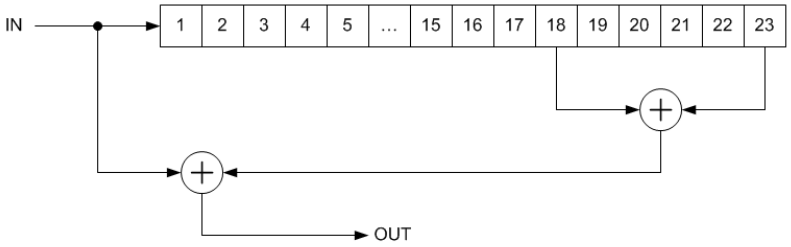
- Write a python implementation of a multiplicative scrambler and a multiplicative descrambler.
- Test the scrambler/descrambler by inserting all ones into the scrambler and seeing if all ones come out of the descrambler.
- Depending on the length of the shift register, there may be errors (some zeros) at the initialization of the descrambler. This is fine.

Task #1

- For now, implementing the below diagrams is fine
- We might make it more generic or choose another shift register length with different taps later



Scrambler



Descrambler

Presentation Outline

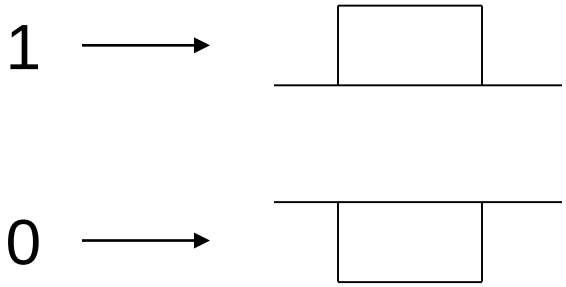
- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

Digital Communications

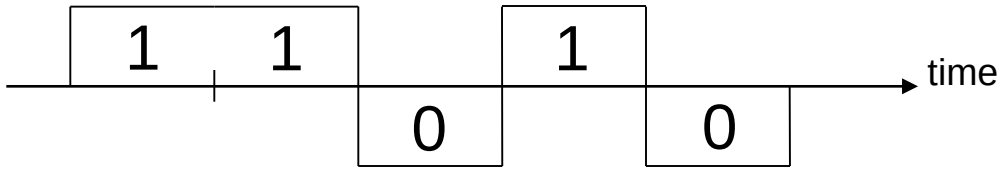
- What is digital communications?
  - In general, digital communications is a wired or wireless communications method that uses an analog signal of a finite duration to convey a finite number of data bits.
  - A series of these finite duration analog signals allows a user to send a message of bits of arbitrary length.

An Example

- Let the binary digits 1 and 0 be mapped to the analog signal/pulses shown below.



- To send the sequence 11010, would look like the signal below.

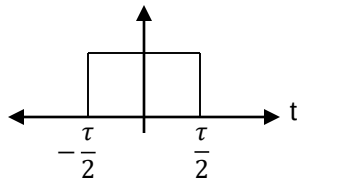


Rect Pulses

- Is that it? Just use rectangular pulses to transmit bits of information?

No!

- Won't focus on this for now… but…
- Remember Fourier Transforms?



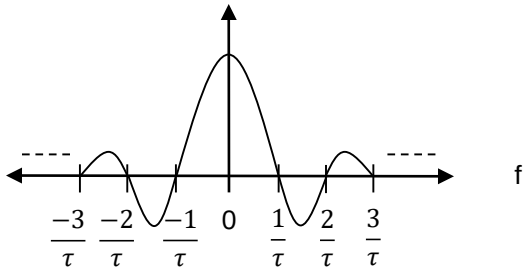
$$\Pi(t) = \begin{cases} 1, & |t| < \tau/2 \\ 0, & \text{otherwise} \end{cases}$$

Fourier Transform

$\mathcal{F}$

$\mathcal{F}^{-1}$

Inverse
Fourier Transform



$$\tau \times \text{sinc}(f\tau) = \frac{\sin(\pi f \tau)}{\pi f \tau}$$

Continues in both directions, slowly decaying, forever

Undesirable

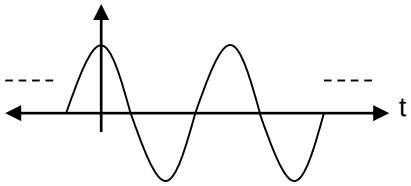
Rect Pulses Continued

- As I said, we won't focus on this for now, but…
  - See section 3.2 "Pulse-Shape Designs for Band-Limited Communications" in *Signal Processing Techniques for Software Radios, 2nd Edition* by Prof. Farhang.

Sinusoids

- What types of signals should we use for digital communications?
- Let's take a closer look at sinusoids.

- Do you remember this from your signal processing class?



$\cos(2\pi f_0 t)$

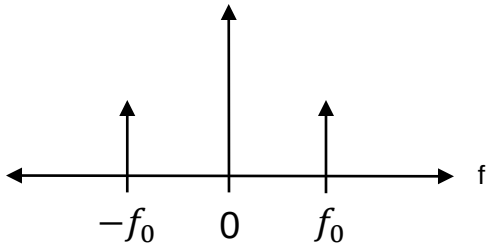
Continues forever in both directions

Fourier Transform

$\mathcal{F}$

$\mathcal{F}^{-1}$

Inverse Fourier Transform



$\frac{1}{2}[\delta(f - f_0) + \delta(f + f_0)]$

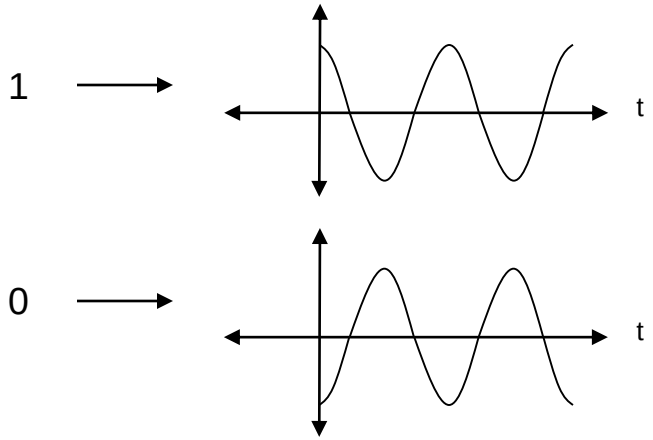
Conveying Information with Sinusoids

- How do we convey information (bits) using sinusoids?
- There are a few basic ways to do this:
  - Input bits determine sinusoid amplitude
  - Input bits determine sinusoid phase
  - Input bits determine sinusoid frequency

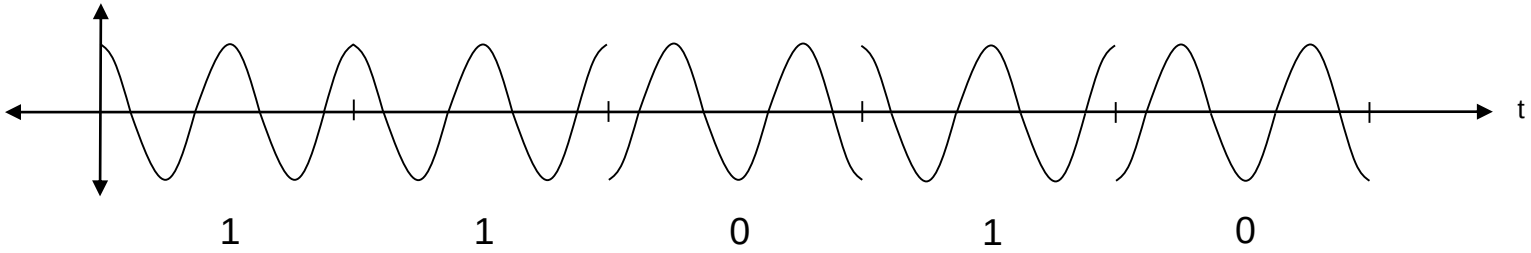
Pulse-Amplitude Modulation

- In pulse-amplitude modulation (PAM) we adjust the amplitude of the sinusoid based on bit input
- Example: Let's convey one bit of information per sinusoid segment (2PAM).

Bit Input	Sinusoid Amplitude
1	1
0	-1



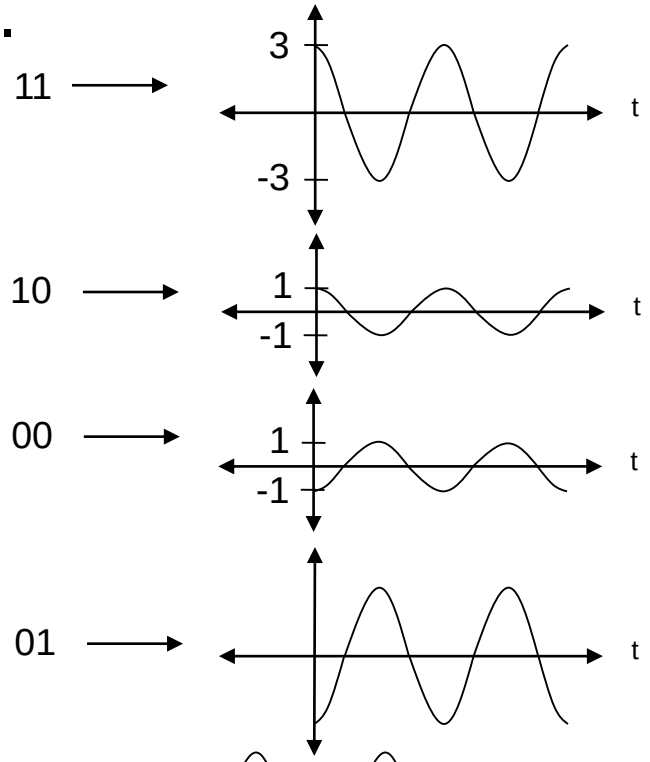
Transmitted 11010 sequence



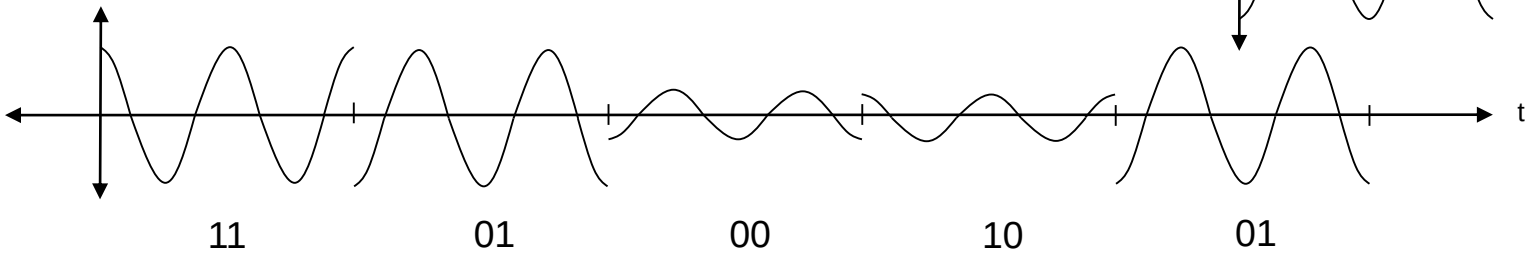
4PAM

- Example: Let's convey two bits of information per sinusoid segment (4PAM).

Bit Input	Sinusoid Amplitude
11	3
10	1
00	-1
01	-3

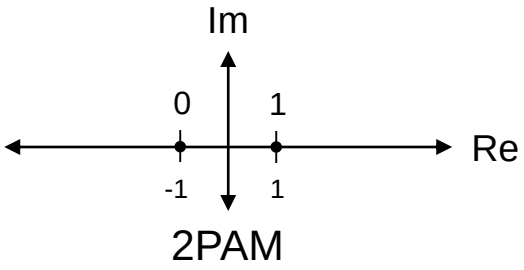


Transmitted 1101001001 sequence

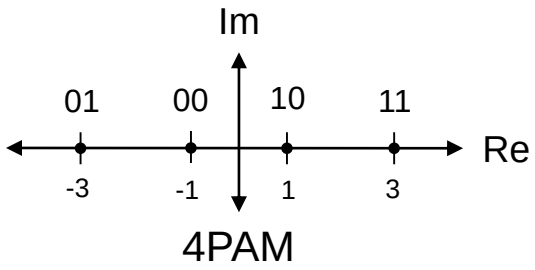


Constellation Maps

- Each sinusoid segment is multiplied by a “symbol” the represents the underlying bits
- We can take the set of symbols for a modulation type and display those as dots on a plot.
- The set of dots is called a constellation map



2PAM



4PAM

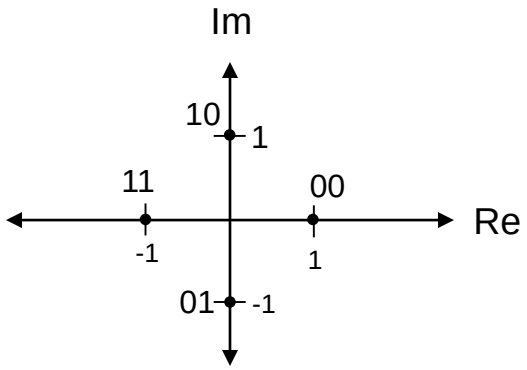
Re - Real

Im - Imaginary

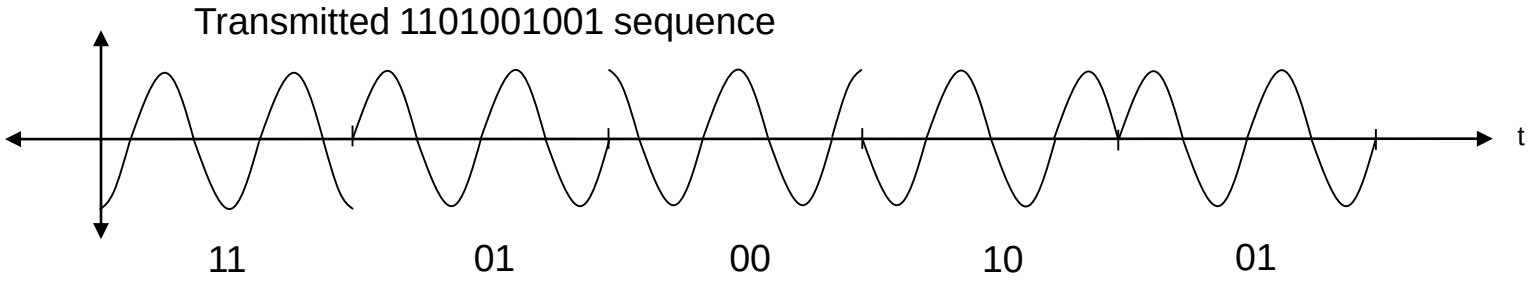
Phase-Shift Keying

- With phase-shift keying (PSK), we adjust the phase of the sinusoid based on the bit input
- Example: Let's map two bits of information on four phases

Bit Input	Sinusoid Phase (deg)
11	180
10	90
00	0
01	270

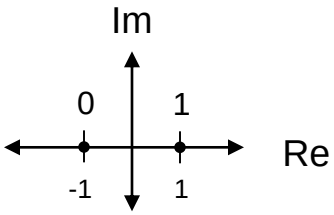


QPSK (4PSK)

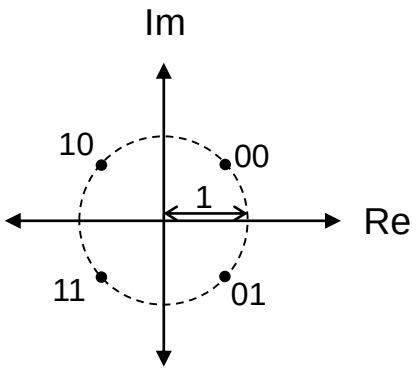


Other Flavors of PSK

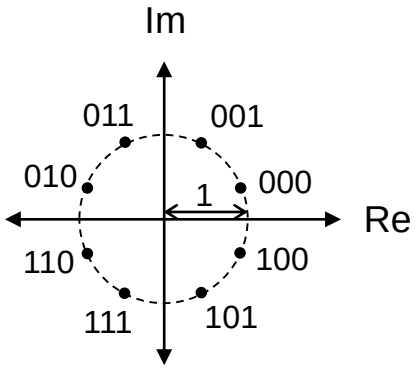
- The symbols of a PSK constellation are, in general, spaced evenly on the unit circle in the complex plane
- Note that BPSK (2PSK) is the same as 2PAM
- QPSK symbols more commonly take the phases shown on this slide as opposed to the previous slide



BPSK



QPSK (4PSK)



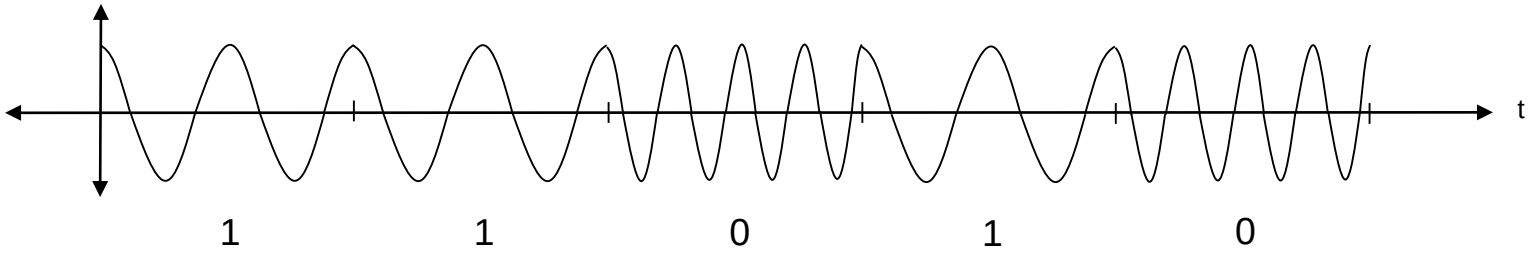
8PSK

Frequency Shift Keying

- With frequency-shift keying (FSK), we adjust the frequency of the sinusoid based on the bit input

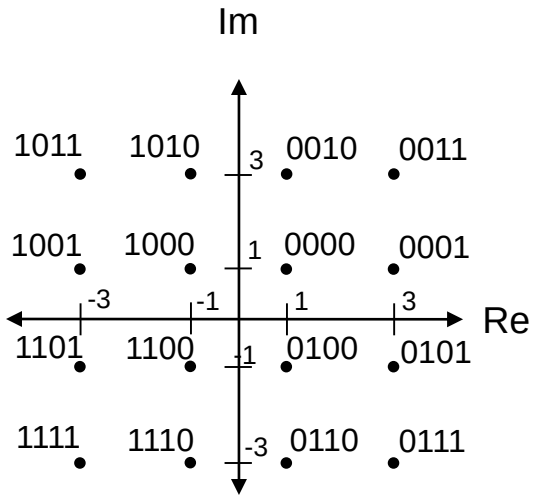
Bit Input	Sinusoid Frequency
1	f_1
0	f_0

Transmitted 11010 sequence



Quadrature Amplitude Modulation

- It is possible to adjust various parameters (amplitude, phase, frequency) simultaneously
- Quadrature Amplitude Modulation (QAM) adjust phase and amplitude depending on input bits
- It's like PAM in 2D, real and imaginary



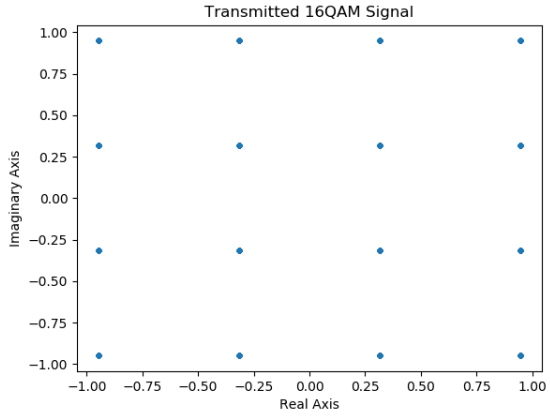
16QAM

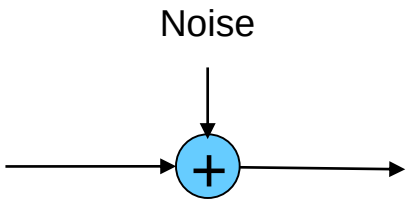
My Experience

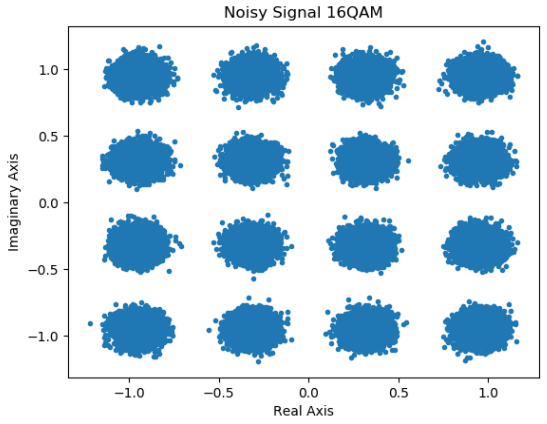
- What have I seen in “real” digital communication systems?
  - BPSK, QPSK, 8PSK
  - BFSK, 4FSK, and higher
  - 8QAM, 16QAM, 64QAM, 256QAM, 1024QAM
  - And a bunch of other more complicated varieties

Noise

- This seems fairly easy so far. What problems do we encounter in a real system?
  - There are a lot of problems, but we will focus on noise for now
- Let's say we are sending a "long" message of composed of many, many 16QAM data symbols
- We will see noise on the symbols at the receiver

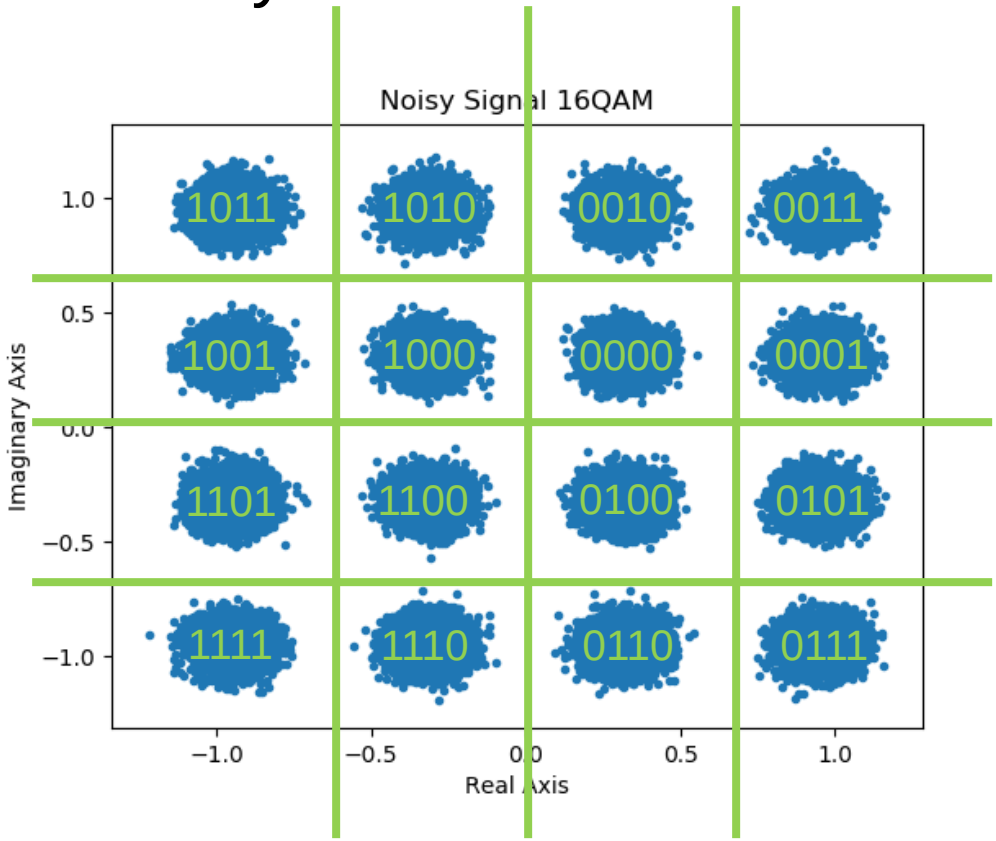






Noise

- You can convert symbols back to bits by defining decision regions and outputting the appropriate bits for each symbol



Power and Noise

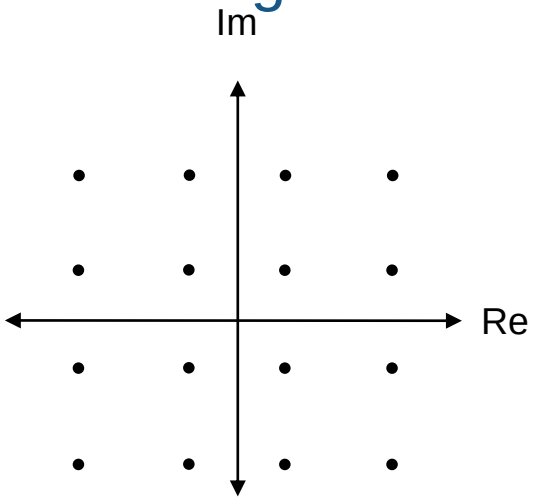
- If you can transmit 4, 6, or even 10 bits at a time, why would you every transmit just one lousy bit at a time?

Power limitations and noise

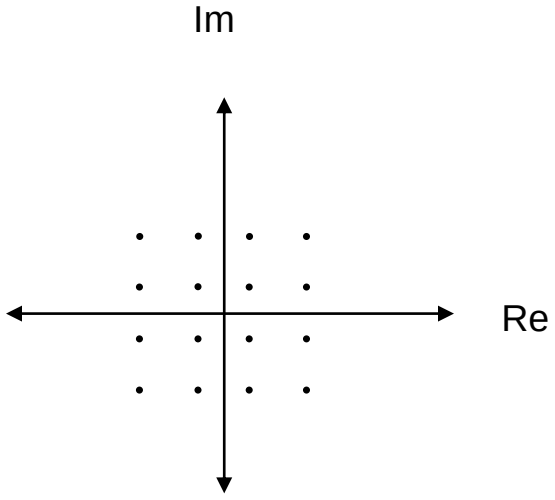
- Communication systems are power-limited
  - Finite power source (such as a battery)
  - Regulatory requirements (FCC)
  - Etc.

Power and Signal Constellations

- How does transmit power affect the signal constellation?
  - More signal power moves the constellation points farther apart
  - Less signal power moves the constellation points closer together



16QAM – Higher Power



16QAM – Less Power

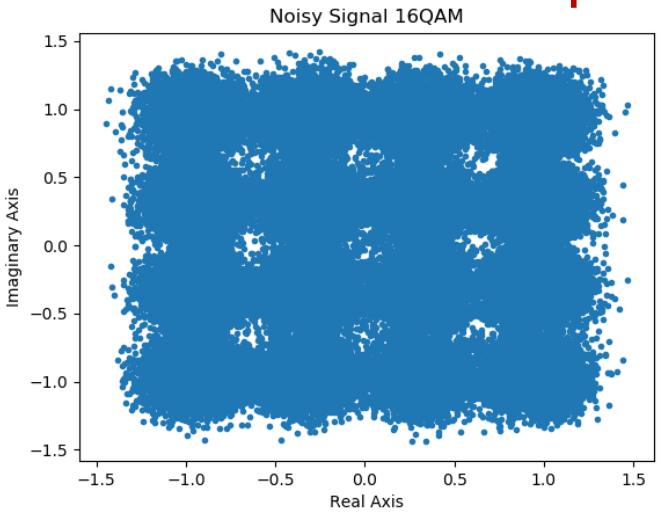
Noise

- In general, noise in a system is relatively constant.
- The ratio between signal power to noise power is called the signal to noise ratio (SNR)
- Higher SNR is better

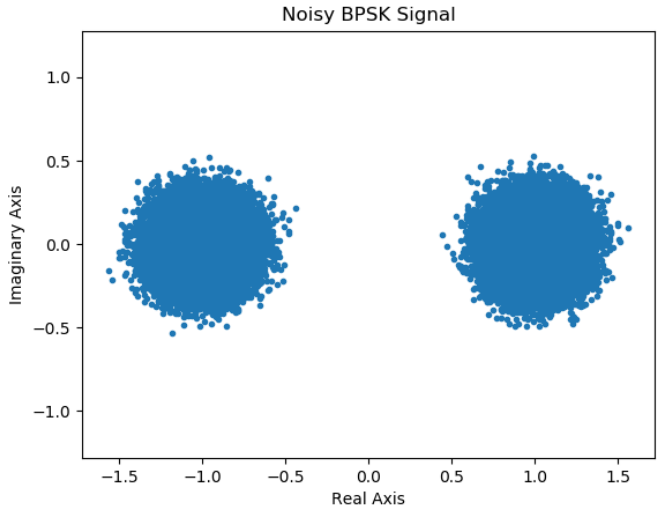
Noise

- What if the noise is too big for our constellation?
  - Symbol to bit conversion mistakes are made
- Many things you can do to overcome this, but for the same transmit power, you can just change the constellation and transmit at a lower data rate

Mistakes in bit output



No mistakes in bit output



Same transmit and noise power for 16QAM and BPSK

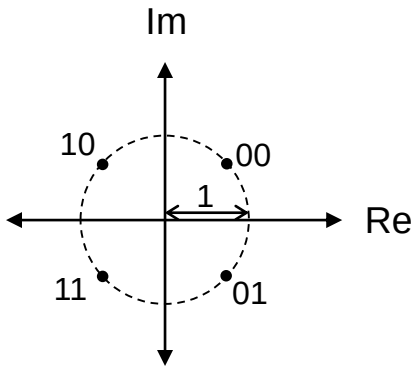
Presentation Outline

- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

Task #2

- Implement a modulation mapper and demapper for a QPSK constellation in python
  - Arrange the input bits into groups of two bits per group
  - For each group of two input bits there will be one complex number as output, i.e., if there were 100 input bits, there should be 50 complex output numbers

Bit Input	Complex Output
00	$e^{j\pi/4}$
10	$e^{j3\pi/4}$
11	$e^{j5\pi/4}$
01	$e^{j7\pi/4}$

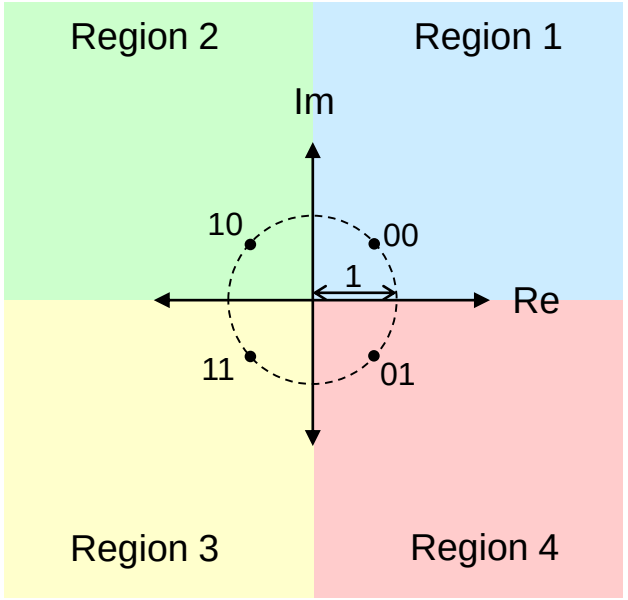


QPSK (4PSK)

Task #2

- For the demapper, we map numbers in decision regions to bits
  - For QPSK there are four decision regions
  - If an input number lands on a boundary, it can go into either region that it touches

Complex Input	Bit Output
Region 1	00
Region 2	10
Region 3	11
Region 4	01



Presentation Outline

- Project Introduction
  - Hardware
  - Software
  - Modem Design
- PRBS
- Task #1
- Introduction to Digital Communications
  - Transmission of bits
  - Modulation methods
  - Signal Power, Noise Power, SNR
- Task #2
- To be continued…

Part 2

Presentation Outline

- Digital Signal Processing
  - Sampling/analog-to-digital conversion
  - Aliasing
  - Frequency constraints
  - Digital-to-analog conversion
- Frequency Analysis
  - Fourier Transforms/Inverse Fourier Transforms
  - Discrete Fourier Transforms/Inverse Discrete Fourier Transforms
- Orthogonal Frequency Division Multiplexing (OFDM) Modulation
  - IDFT Modulation
  - Orthogonality
  - DFT Demodulation
  - Cyclic Prefix
- Task #3
- Task #4

Digital Signal Processing

- In general, signals in the “real” world are analog signals.
- However, computers like to work on digital signals.

- What is a digital signal? How do we convert to/from digital signals?

Sampling

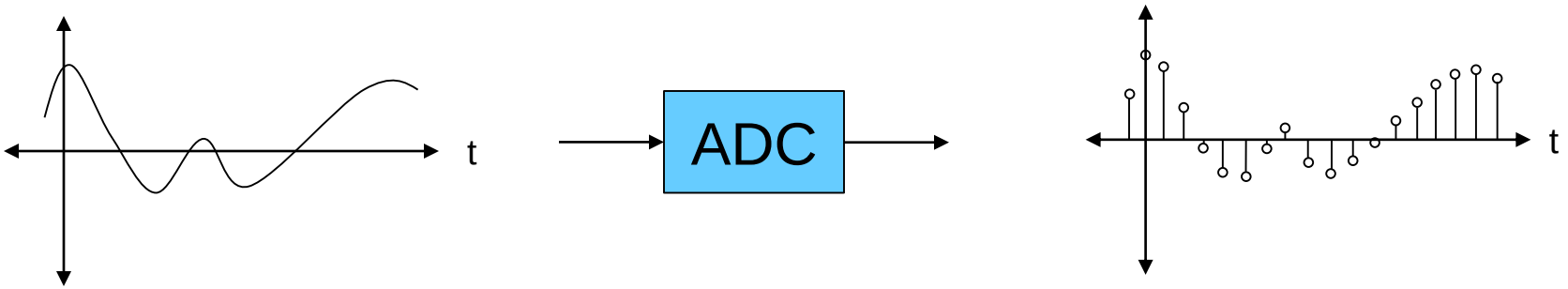
- Let's say that we have an analog signal $x_a(t)$
- We convert $x_a(t)$ to a discrete-time signal as follows:

$$x[n] = x_a(nT_s)$$

  - n is an integer
  - T_s is called the sample period. It is the time spacing between data points (samples) in the discrete signal

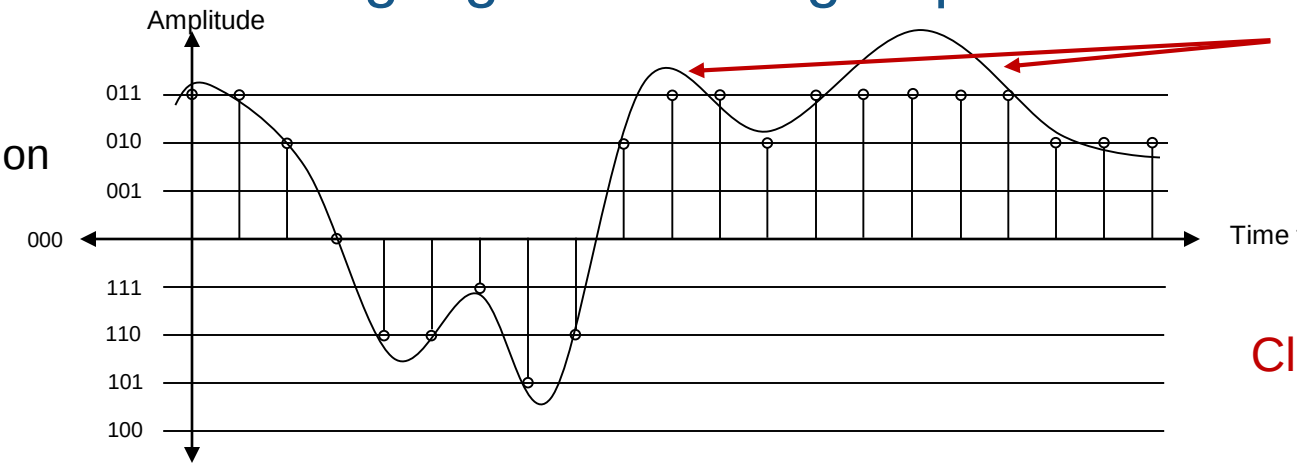
Analog to Digital Conversion

- The process of converting an analog signal to a digital signal is performed by an analog to digital converter (ADC) or A/D and is called sampling.
  - The analog signal is fed into the ADC along with a clock. The clock runs at the sample rate $F_s = 1/T_s$.
  - At every tick of the clock, the ADC produces a "sample" of the analog signal
  - Samples of the signal are taken with a constant (or uniform) spacing between samples.



Quantization

- The input to the ADC is an electrical signal with a varying voltage level.
- The output of the ADC, or samples, are represented by a finite number of bits
- A finite number of bits can only represent a finite number of values
  - The analog signal values get quantized



3-bit quantization

If the input voltage exceeds the range of the ADC, the signal gets "clipped"

Clipping is bad

ADC Limits

- Are there limits to the sampling process?
- Anything that we should think about?

Yes!

- The bandwidth of the signal must be limited

Continuous-Time Sinusoids

- Consider this continuous-time sinusoid

$$x_a(t) = \cos(2\pi F t + \theta)$$

  - This signal is periodic with period $T = {}^{1}/_{F}$
  - Continuous-time sinusoids with distinct frequencies are distinct
    - The frequency range of a continuous time sinusoid is $-\infty < F < \infty$
  - Increasing frequency F increases the signal oscillations

Discrete-Time Sinusoids

- Let's sample the continuous-time sinusoid with period T_s

$$x[n] = x_a(nT_s) = \cos(2\pi F n T_s + \theta)$$
$$= \cos\left(2\pi \frac{F}{F_s} n + \theta\right)$$
$$= \cos(2\pi f n + \theta)$$

- Where $f = F/F_s$ is the discrete-time frequency in cycles per sample

Discrete-Time Sinusoid Results

$$x_a[n] = \cos(2\pi f n + \theta)$$

- Discrete-time sinusoid properties:
  - This signal is only periodic if the frequency f is rational (no proof given)
  - Discrete-time sinusoids separated in frequency f by integer values are identical
  - The highest oscillation in a discrete-time sinusoid is when $f = \frac{1}{2}$ or when $f = -\frac{1}{2}$

Different Frequencies, Identical Signals

- Discrete-time sinusoids separated in frequency by integers values are identical
  - Let the frequency f be in the range $-\frac{1}{2} < f < \frac{1}{2}$
  - Let's consider a sinusoid with a new frequency of $f_n = f + k$ where k is any integer

$$\begin{aligned}\cos(2\pi f_n n + \theta) &= \cos[2\pi(f + k)n + \theta] \\ &= \cos(2\pi f n + 2\pi k n + \theta) \\ &= \cos(2\pi f n + \theta)\end{aligned}$$

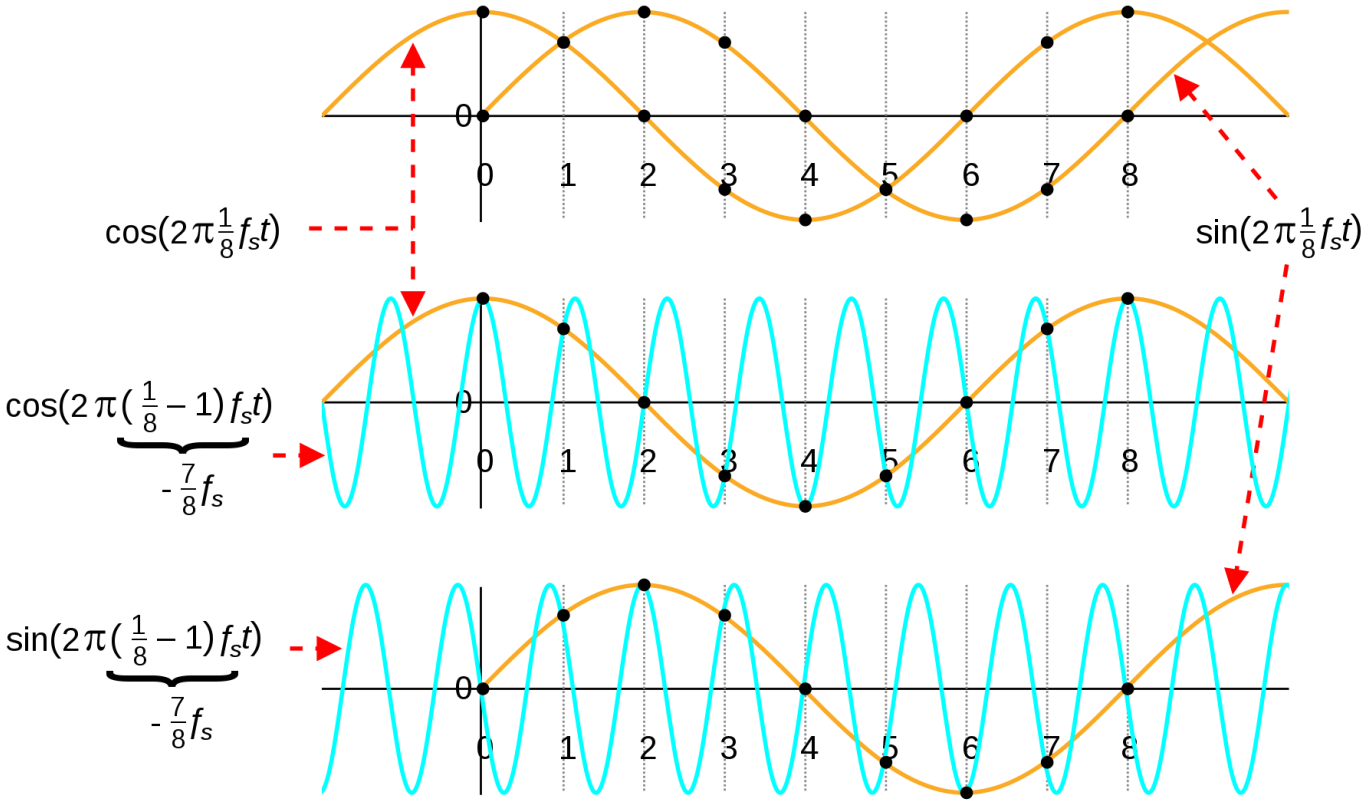
Aliasing

- Aliasing in digital signal processing is when one signal looks like another.
- The sinusoids of the previous slide are a perfect example

$$\begin{aligned}\cos(2\pi f_n n + \theta) &= \cos[2\pi(f + k)n + \theta] \\ &= \cos(2\pi f n + 2\pi k n + \theta) \\ &= \cos(2\pi f n + \theta)\end{aligned}$$

- The sinusoid of frequency f_n cannot be distinguished from that of frequency f when sampled

Visual Example of Aliasing



Practical Implications of Aliasing

- Need to limit the frequency content of an input signal prior to sampling, otherwise the aliasing causes “problems” with the signal we are interested in

- What is the allowable frequency range of the input signal?

Input Signal Frequency Range

- From slide 52 we saw that $f = \frac{F}{F_s}$ where f is the discrete-time signal frequency, F is the continuous-time signal frequency, and F_s is the sampling rate in Hz

$$-\frac{1}{2} \leq f \leq \frac{1}{2} \quad \rightarrow \quad -\frac{1}{2} \leq \frac{F}{F_s} \leq \frac{1}{2} \quad \rightarrow \quad -\frac{F_s}{2} \leq F \leq \frac{F_s}{2}$$

- The allowable frequency range of the input is $-\frac{F_s}{2} \leq F \leq \frac{F_s}{2}$
- This is usually controlled for by filtering the ADC input first with an anti-aliasing filter that eliminates frequencies outside this range

Sampling Theorem

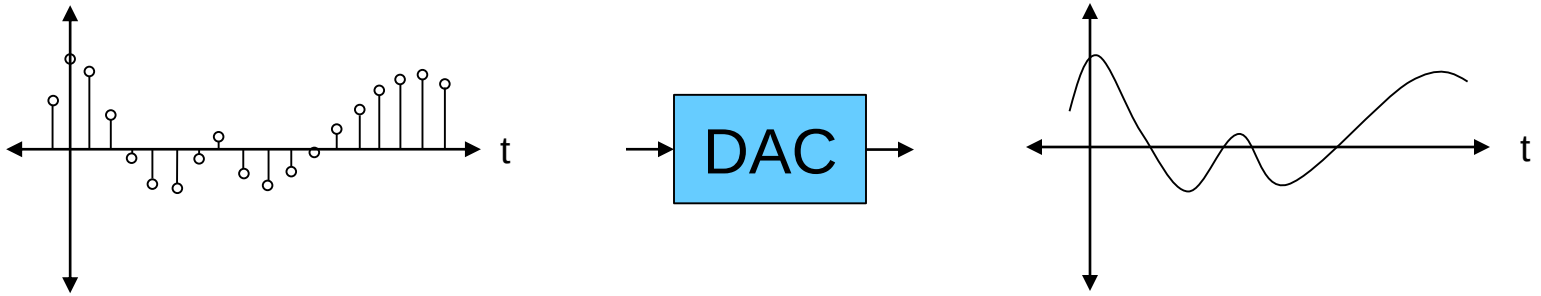
- If the highest frequency contained in an analog signal $x_a(t)$ is $F_{\max} = B$ then $x_a(t)$ can be exactly recovered from $x[n]$ as

$$x_a(t) = \sum_{n=-\infty}^{\infty} x[n] \frac{\sin(2\pi Bt)}{2\pi Bt}$$

- No proof given here

Digital to Analog Conversion

- Converting a digital signal to an analog signal is performed by a digital to analog converter (DAC) or D/A.
  - This process is the reverse of the A/D conversion, but it requires a low-pass filter (LPF)



Presentation Outline

- Digital Signal Processing
  - Sampling/analog-to-digital conversion
  - Aliasing
  - Frequency constraints
  - Digital-to-analog conversion
- Frequency Analysis
  - Fourier Transforms/Inverse Fourier Transforms
  - Discrete Fourier Transforms/Inverse Discrete Fourier Transforms
- Orthogonal Frequency Division Multiplexing (OFDM) Modulation
  - IDFT Modulation
  - Orthogonality
  - DFT Demodulation
  - Cyclic Prefix
- Task #3
- Task #4

Frequency Analysis

- We can analyze the frequency content of a continuous-time time-domain signal through the use of the Fourier Transform

Fourier Transform $$X(F) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi Ft} dt$$

Inverse Fourier Transform $$x(t) = \int_{-\infty}^{\infty} X(F) e^{j2\pi Ft} dF$$

Fourier Transform for Discrete-Time Signals

- There is a Fourier transform for discrete-time signals as well

Fourier Transform $$X(f) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi fn}$$

Inverse Fourier Transform $$x[n] = \int_{-\frac{1}{2}}^{\frac{1}{2}} X(f)e^{j2\pi fn}df$$

Discrete Fourier Transform (DFT)

- We are going to focus on a Fourier transform that works with N samples of a discrete-time time-domain signal and produces N samples of a discrete-frequency frequency-domain signal

Discrete Fourier Transform (DFT)

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N}$$

for $0 \leq k \leq N-1$

Inverse Discrete Fourier Transform (IDFT)

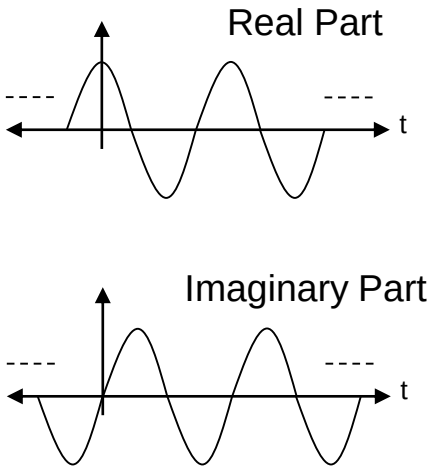
$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi nk/N}$$

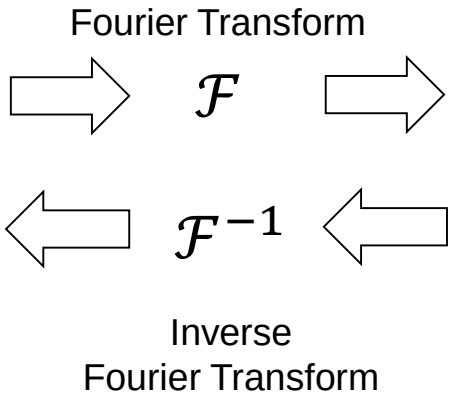
for $0 \leq n \leq N-1$

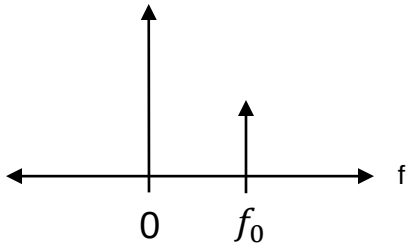
Here k is the frequency-domain sample index

Complex Sinusoids

- We looked at real sinusoids briefly with respect to different methods of digital communications
- We now consider complex sinusoids







$\delta(f - f_0)$

$e^{j2\pi f_0 t} = \cos(2\pi f_0 t) + j\sin(2\pi f_0 t)$

Discrete-Time Complex Sinusoids

- We can sample a continuous-time complex sinusoid to produce

$$x_a(t) = e^{j2\pi f_0 t}$$

$$x[n] = x_a(nT_s) = e^{j2\pi f_0 n T_s}$$

$$x[n] = e^{j2\pi \frac{f_0}{F_s} n}$$

- Let's consider the periodic discrete time sinusoids where $\frac{f_0}{F_s} = \frac{k}{N}$ where k is an integer.

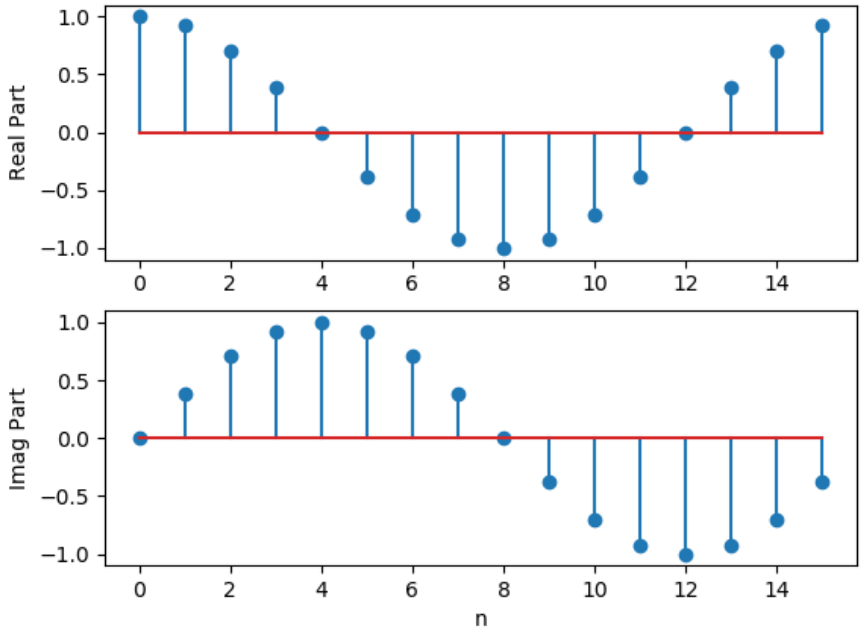
$$x[n] = e^{j2\pi kn/N}$$

Periodic Discrete-Time Complex Sinusoids

- The signal $x[n] = e^{j2\pi kn/N}$ where $0 \leq k \leq N-1$ represents all of the complex sinusoids that have an integer number of cycles in N consecutive samples.

$e^{j2\pi nk/N}$

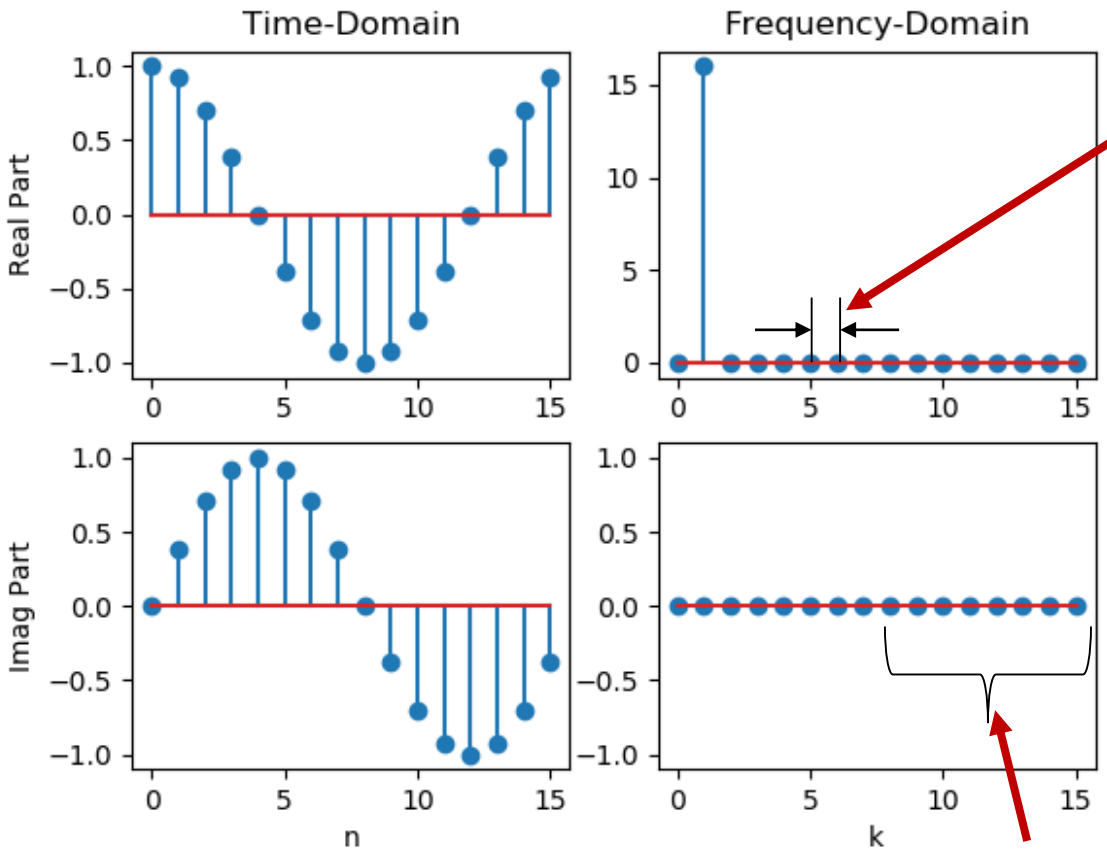
For $N = 16$ and $k = 1$



DFT of Complex Sinsoids

$e^{j2\pi nk/N}$

For $N = 16$ and $k = 1$



The spacing between frequency-domain samples is $\frac{1}{NT_S} = \frac{F_S}{N}$

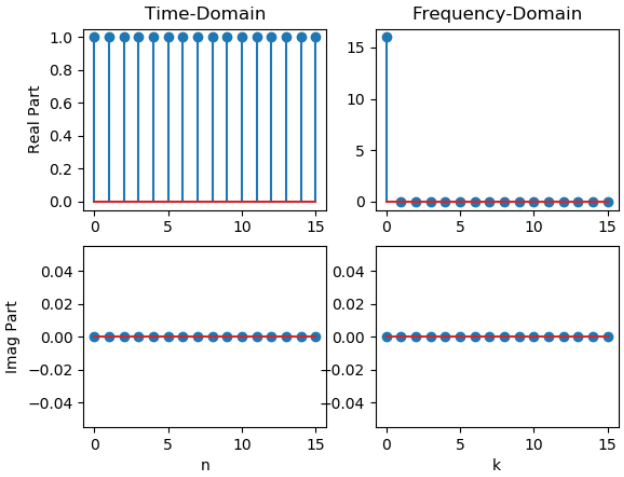
The frequency "bins" from 0 to $N-1$ represent the non-sampled frequencies of 0 to $F_S - \frac{F_S}{N}$.

We will briefly see that the "bins" from $\frac{N}{2}$ to $N-1$ represent the negative frequency range $-\frac{F_S}{2}$ to 0.

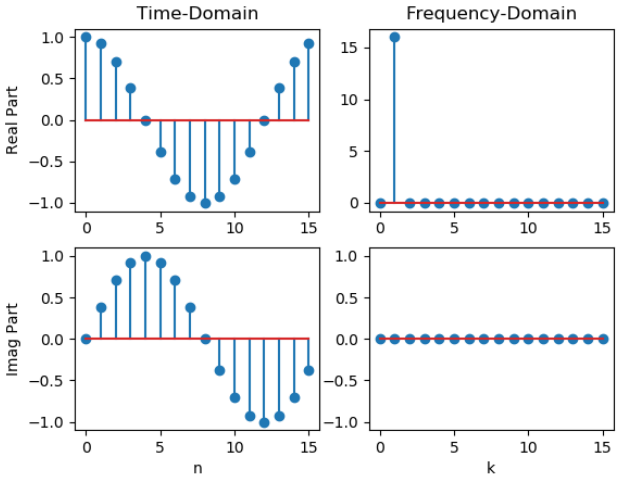
DFT of Complex Sinusoids

- DFT/IDFT complex sinusoid pairs

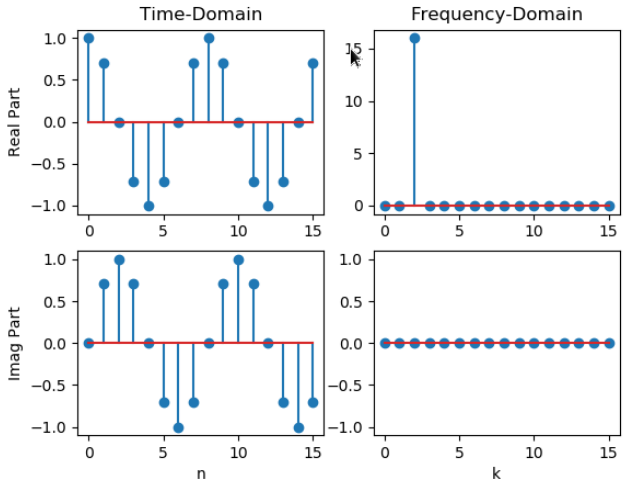
$k = 0$



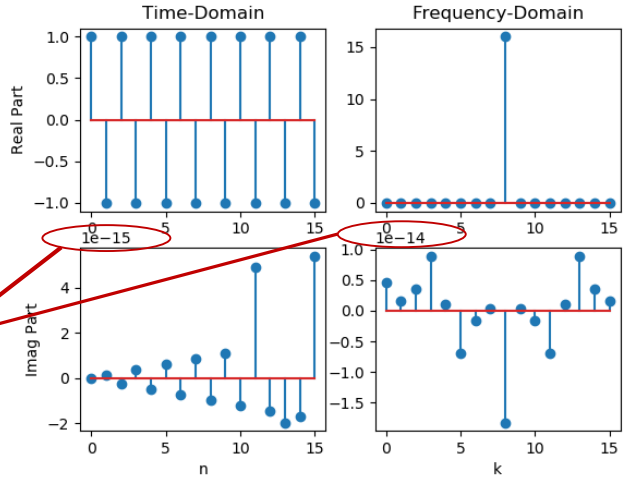
$k = 1$



$k = 2$



$k = 8$

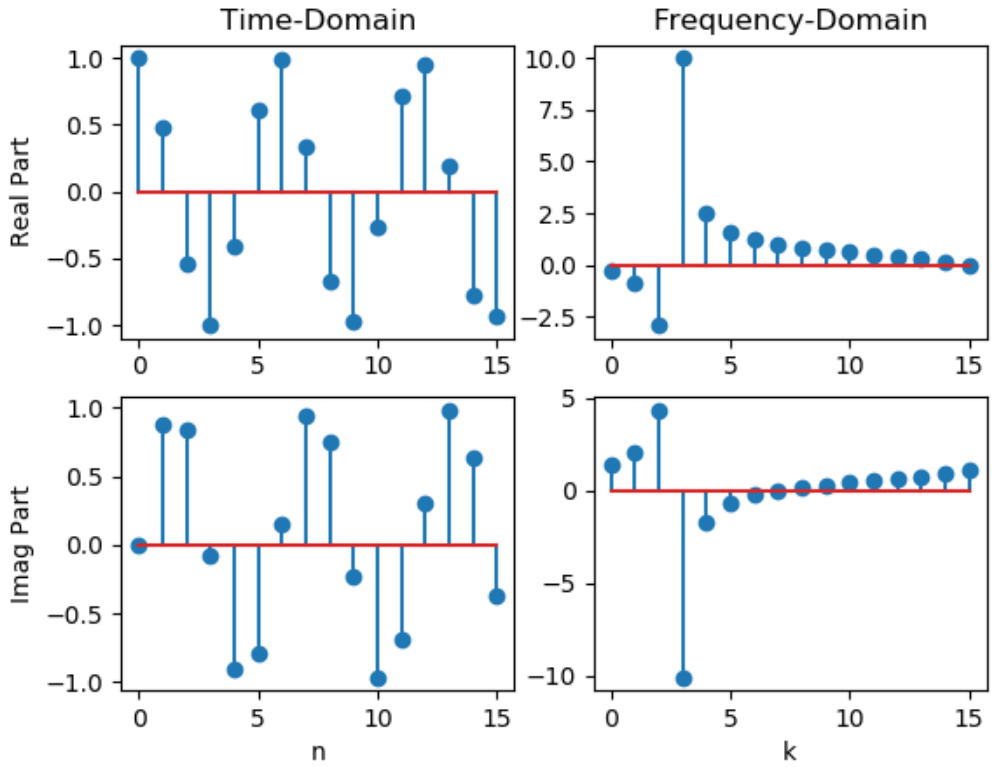


Notice the y-axis scale here

DFT of Complex Sinusoids

- Notice the difference when k is not an integer, i.e., there is a non-integer number of cycles in N samples

$k = 2.732$

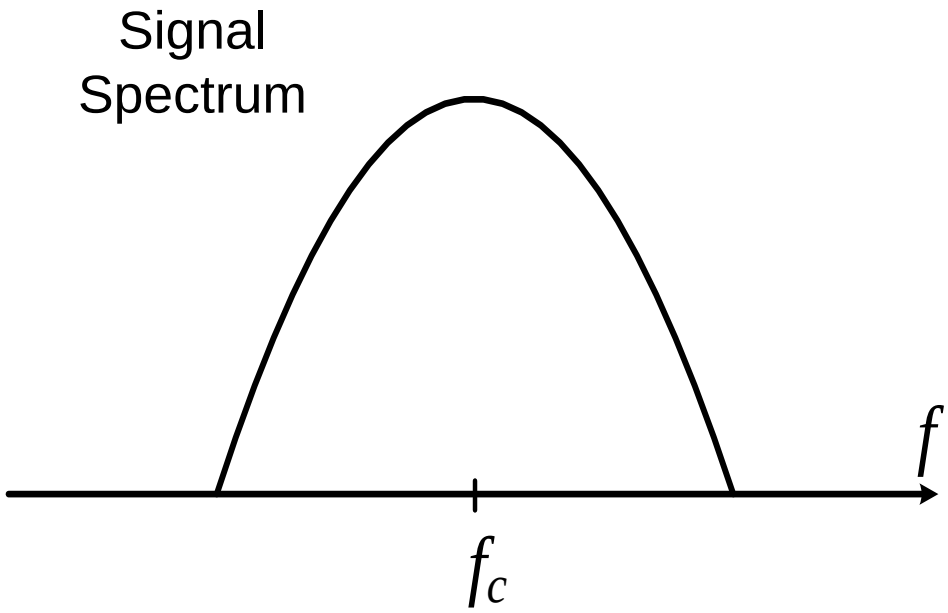


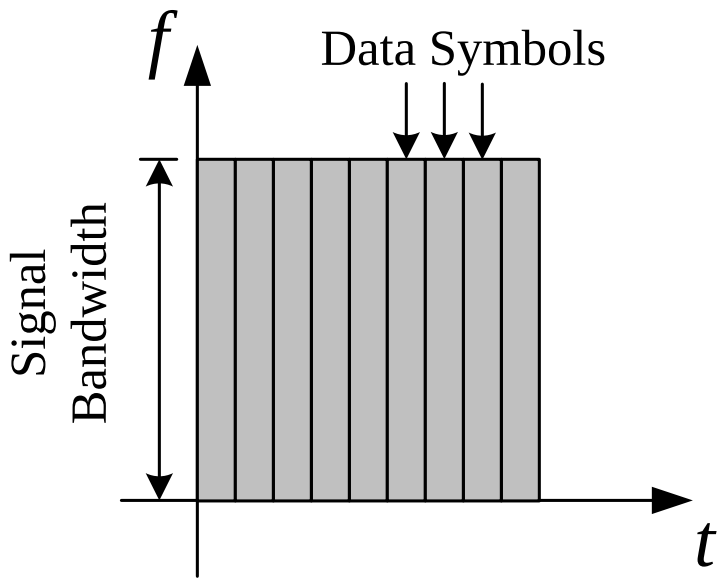
Presentation Outline

- Digital Signal Processing
  - Sampling/analog-to-digital conversion
  - Aliasing
  - Frequency constraints
  - Digital-to-analog conversion
- Frequency Analysis
  - Fourier Transforms/Inverse Fourier Transforms
  - Discrete Fourier Transforms/Inverse Discrete Fourier Transforms
- Orthogonal Frequency Division Multiplexing (OFDM) Modulation
  - IDFT Modulation
  - Orthogonality
  - DFT Demodulation
  - Cyclic Prefix
- Task #3
- Task #4

Single Carrier Communication

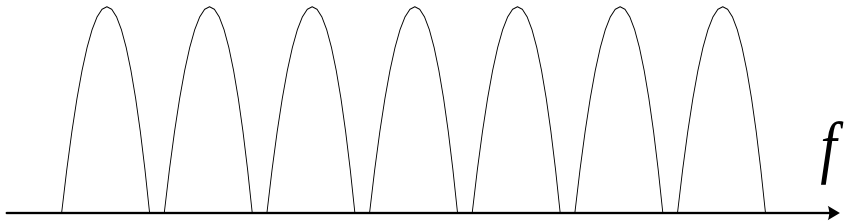
- Transmission of successive data symbols over the entire channel bandwidth

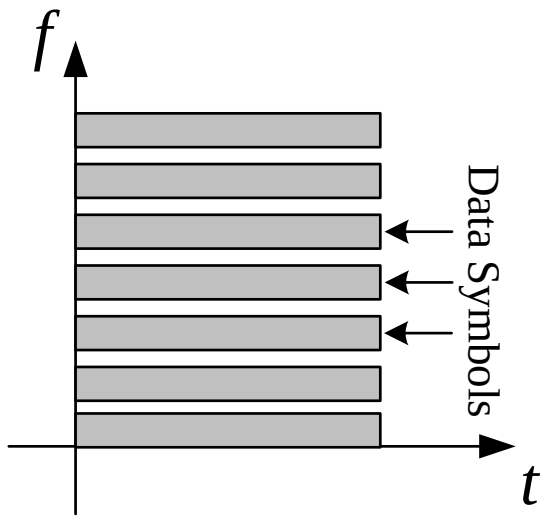




Frequency-Division Multiplexing (FDM)

- Split the signal bandwidth into parallel data streams
  - Different data symbols transmitted in the different subchannels
- Some wasted frequency spectrum due to channel spacing





IDFT Modulation

- Let's take another look at the IDFT equation

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j2\pi nk/N} \qquad \text{for } 0 \le n \le N-1$$

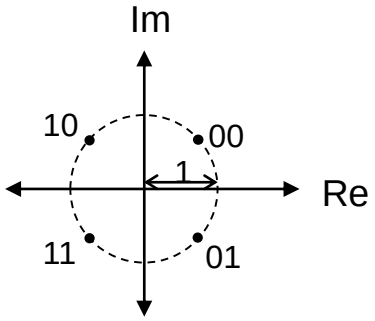
- The IDFT is essentially a sum of complex sinusoids where we have let $f_0/F_s = k/N$
- The sinusoids are "modulated" by the complex values (or symbols) $X[k]$.

QPSK Modulation of Complex Sinusoid

- Let's multiply a complex sinusoid by a QPSK symbol A_k.

$$x[n] = A_k e^{j2\pi \frac{f_0}{F_s} n}$$

Bit Input	Complex Output
00	$e^{j\pi/4}$
10	$e^{j3\pi/4}$
11	$e^{j5\pi/4}$
01	$e^{j7\pi/4}$



QPSK (4PSK)

- Let $A_k = e^{j\pi/4}$

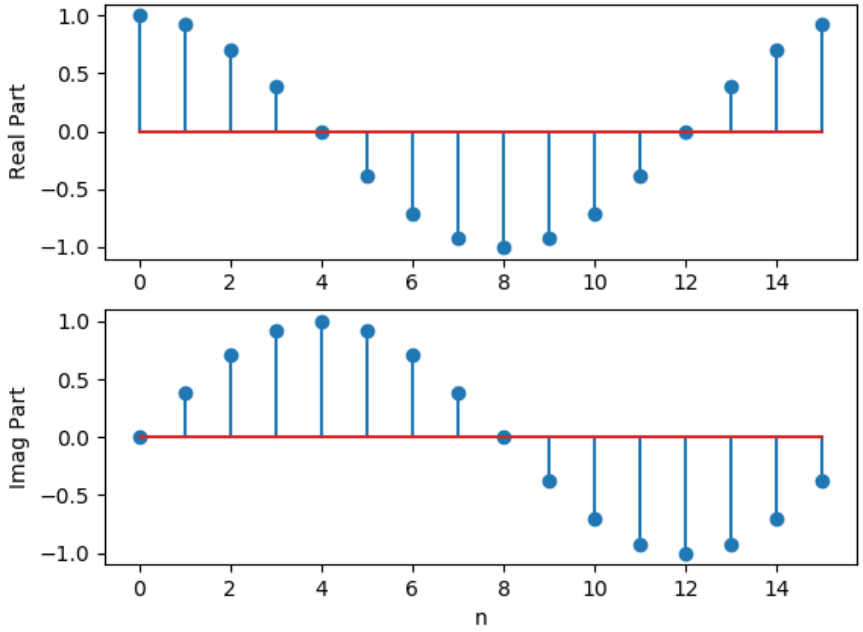
$$x[n] = e^{j\pi/4} e^{j2\pi \frac{f_0}{F_s} n} = e^{j\left(2\pi \frac{f_0}{F_s} n + \frac{\pi}{4}\right)}$$

- We see that this multiplication adjusts the phase, just like we want.

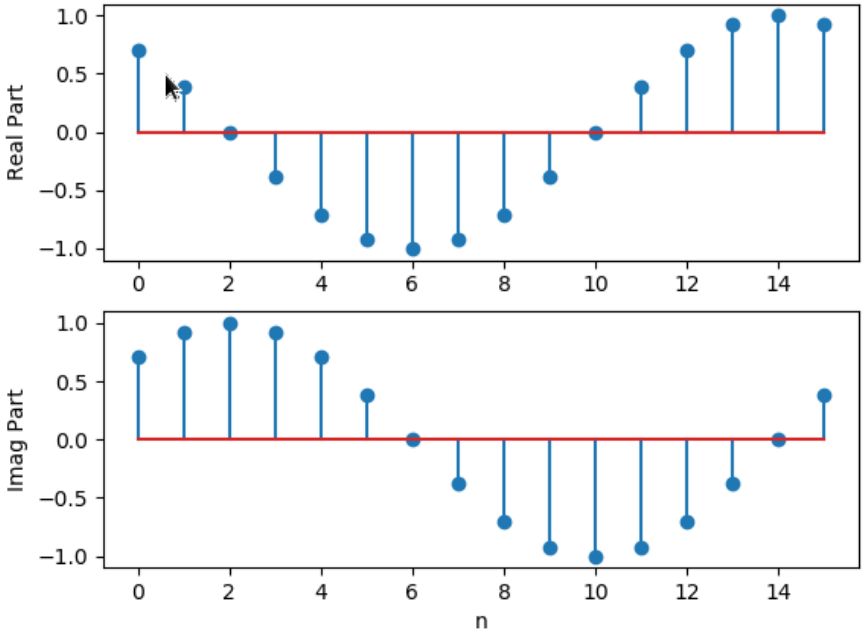
Phase-Shifted Complex Sinusoid

- Shifted by $\pi/4$

$e^{j2\pi nk/N}$ For $N = 16$ and $k = 1$

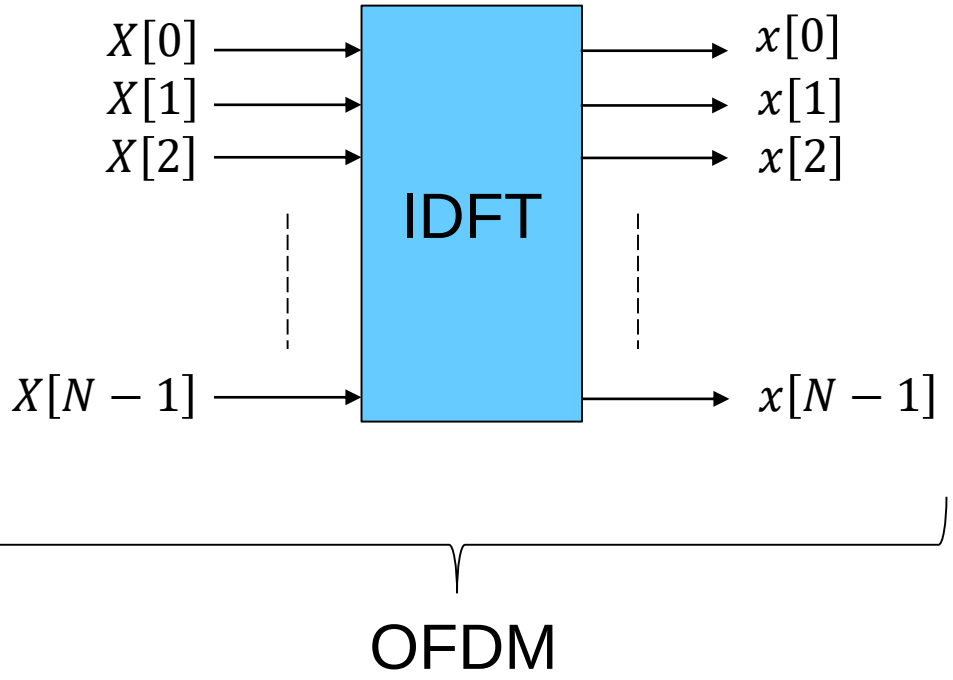


$e^{j\pi/4}e^{j2\pi nk/N}$ For $N = 16$ and $k = 1$



Orthogonal FDM (OFDM)

- It turns out that the IDFT modulation is orthogonal frequency division multiplexing (OFDM)



- Orthogonal how?

Time-Domain Orthogonality

- Two complex functions/signals $f_k(t)$ and $f_l(t)$ are orthogonal over a certain amount of time T if

$$\int_{\tau}^{\tau+T} f_k(t) f_l^*(t) dt = 0$$

where $*$ denotes complex conjugation.

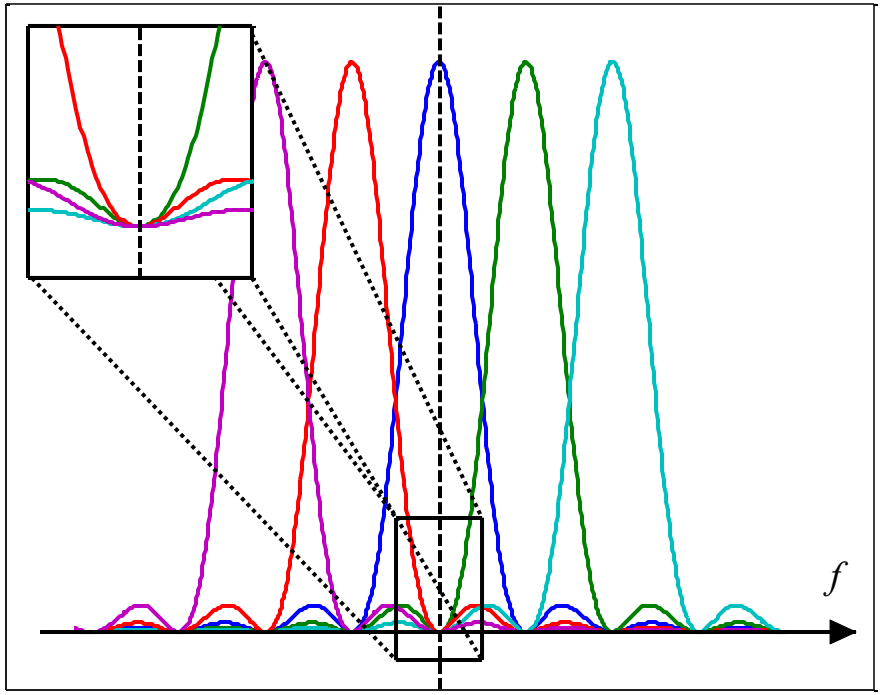
- For discrete-time signals the integral is replaced with a summation.
- The complex sinusoids in the IDFT equation are orthogonal over N samples

$$\sum_{n=0}^{N-1} e^{j2\pi kn/N} e^{-j2\pi ln/N} = \begin{cases} N & \text{for } k = l \\ 0 & \text{otherwise} \end{cases}$$

Frequency-Domain Orthogonality

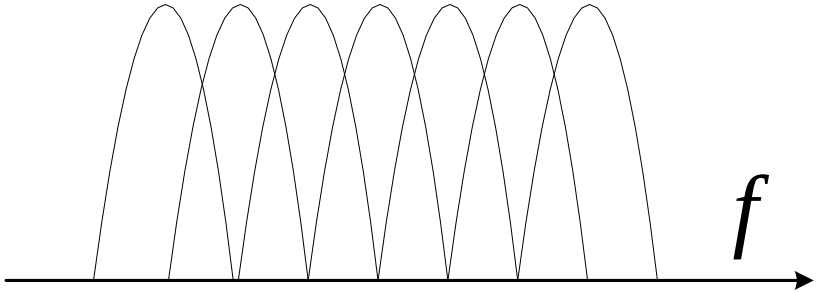
- With no proof given, we will just state the following:
- For each subcarrier peak, all other subcarriers have a null

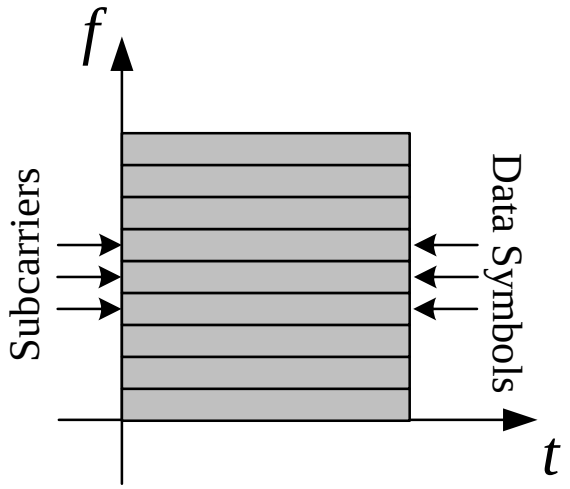
- Contrast this with FDM where there are spectral gaps between subcarriers



Orthogonal FDM

- Data transmitted simultaneously in parallel over multiple overlapping, non-interfering subchannels or subcarriers



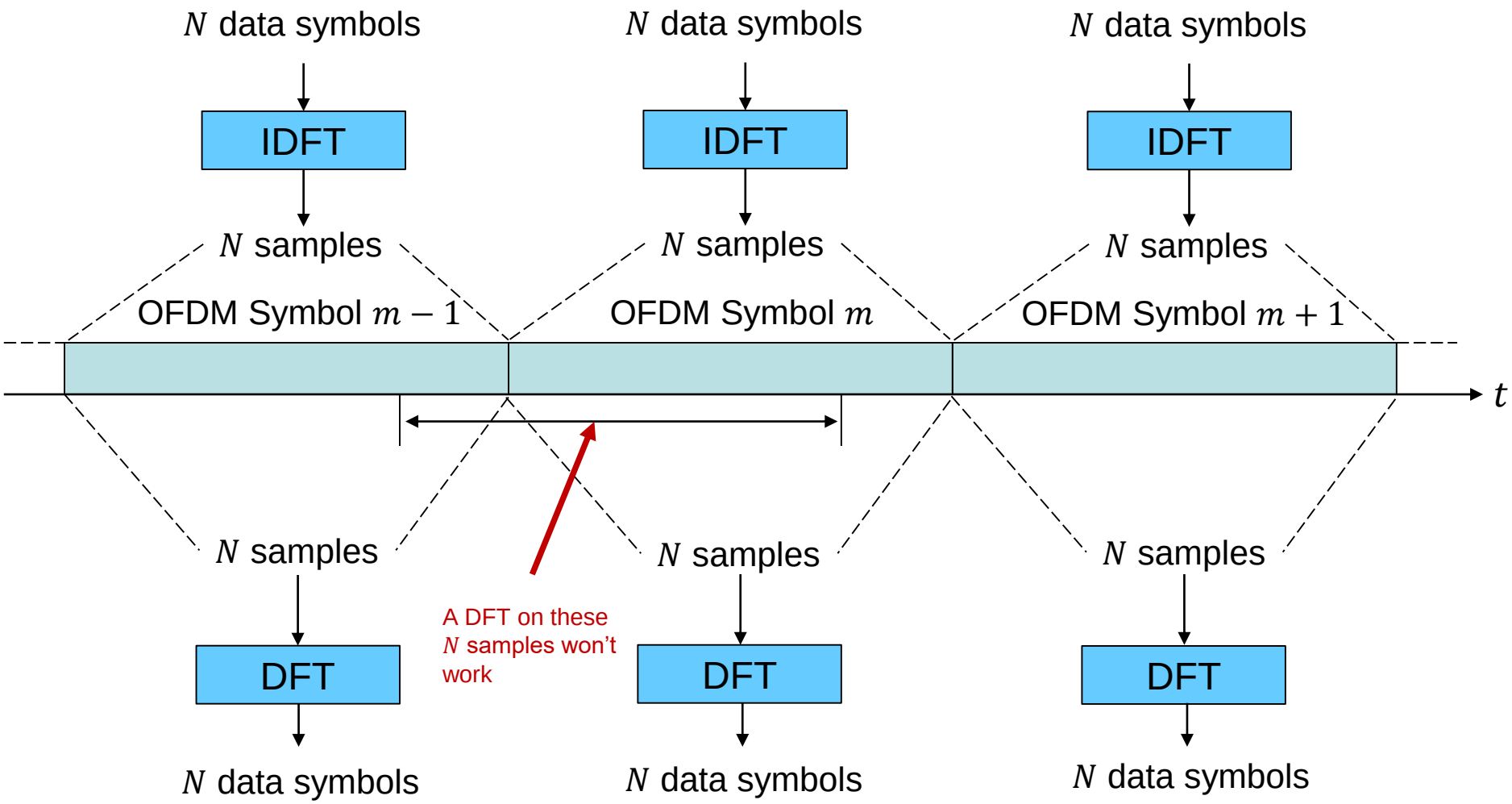


DFT Demodulation

- To get the original symbols back, you just perform a DFT on the received samples.
  - Need to make sure the DFT is aligned with the data properly

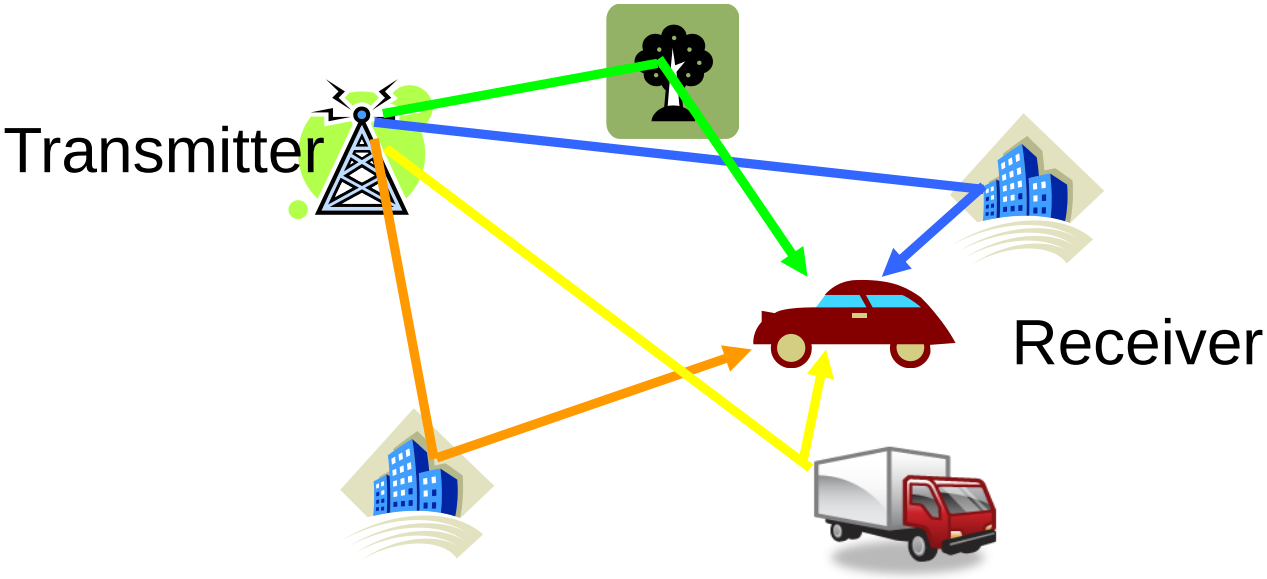
$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N} \qquad \text{for } 0 \leq k \leq N-1$$

OFDM Modem

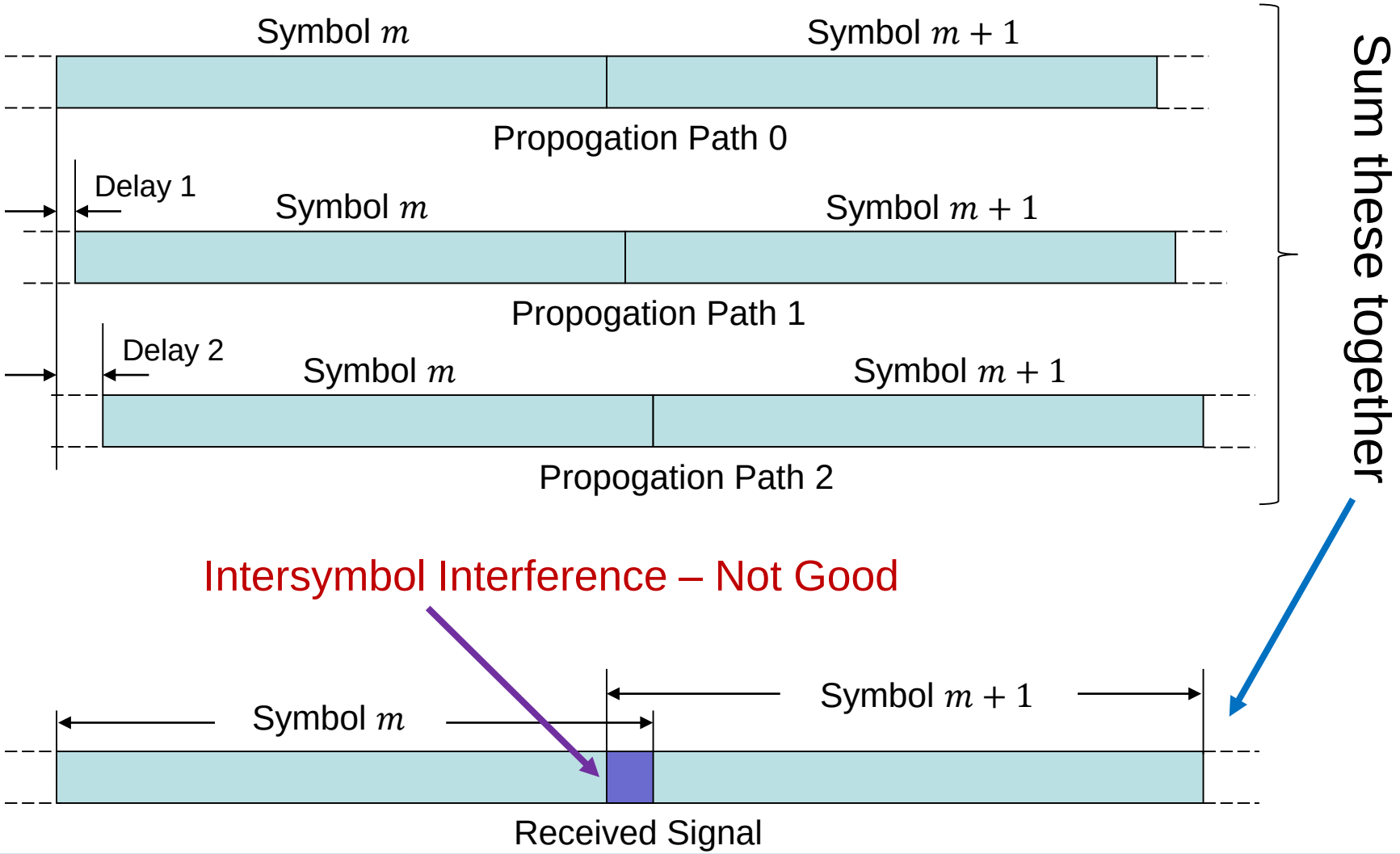

N data symbols
N data symbols
N data symbols
IDFT
IDFT
IDFT
N samples
N samples
N samples
OFDM Symbol $m-1$
OFDM Symbol m
OFDM Symbol $m+1$
t
N samples
N samples
N samples
A DFT on these N samples won't work
DFT
DFT
DFT
N data symbols
N data symbols
N data symbols

Multipath Channel

- The transmitted signal propagates through multiple paths from the transmitter to the receiver
- The received signal is the superposition of multiple (delayed) copies of the transmitted signal

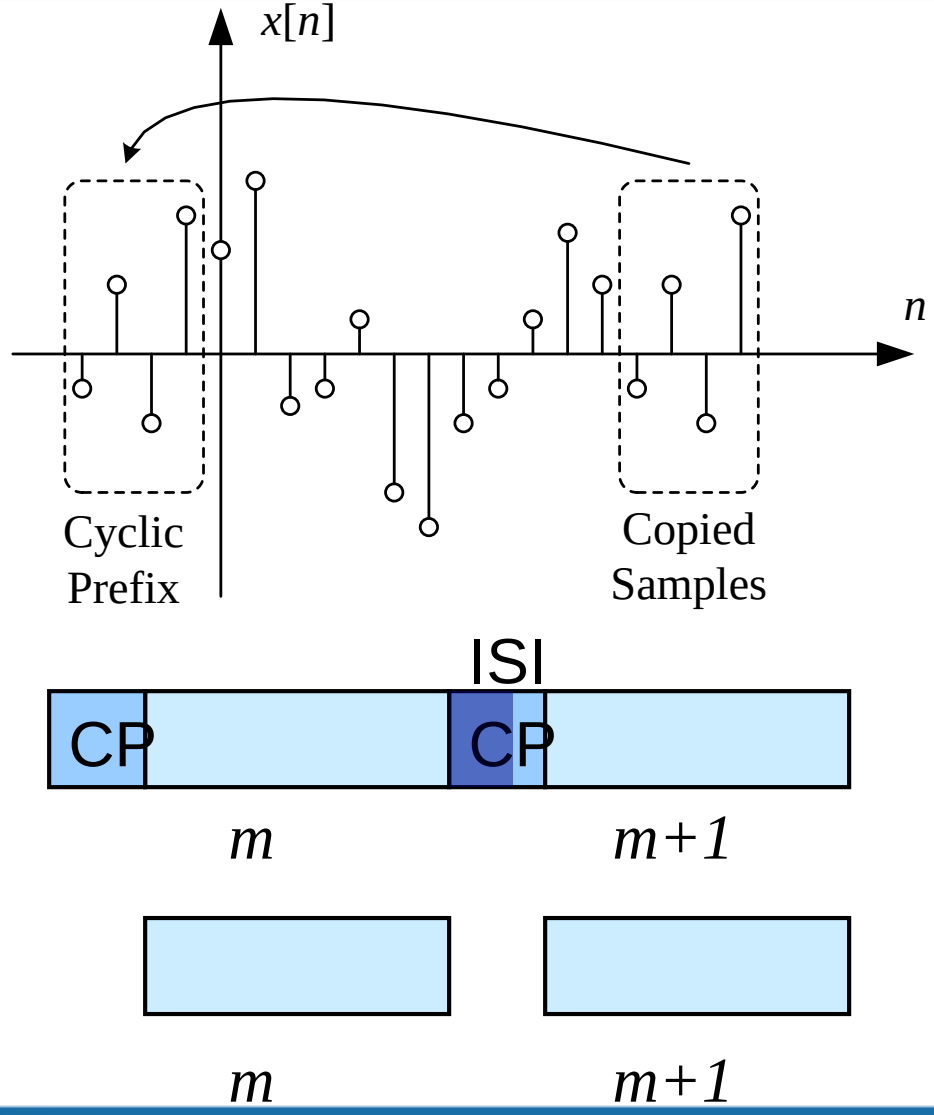


Signal Experiencing Multipath



Cyclic Prefix

- Reduce/eliminate intersymbol interference (ISI)
  - Accomplished through the use of a cyclic prefix
- A cyclic prefix is a cyclic extension of an OFDM symbol.
- Cyclic prefix removed at the receiver.
  - ISI eliminated if CP length exceeds channel length
  - Take DFT of remaining samples



Presentation Outline

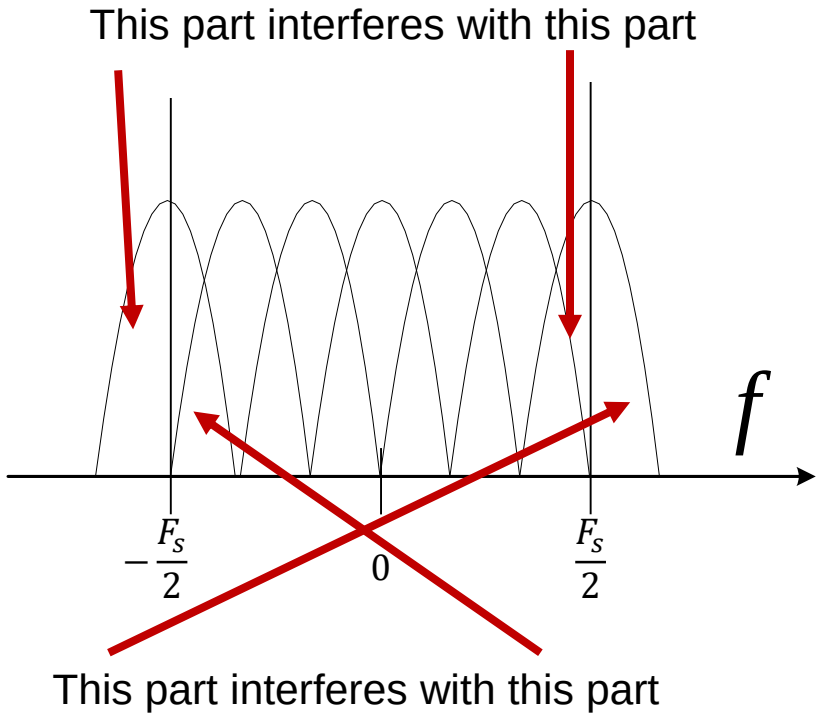
- Digital Signal Processing
  - Sampling/analog-to-digital conversion
  - Aliasing
  - Frequency constraints
  - Digital-to-analog conversion
- Frequency Analysis
  - Fourier Transforms/Inverse Fourier Transforms
  - Discrete Fourier Transforms/Inverse Discrete Fourier Transforms
- Orthogonal Frequency Division Multiplexing (OFDM) Modulation
  - IDFT Modulation
  - Orthogonality
  - DFT Demodulation
  - Cyclic Prefix
- Task #3
- Task #4

Task #3

- Build an IDFT modulator and a DFT demodulator
- Use FFT/IFFT for DFT/IDFT computation
  - FFT/IFFT stand for Fast Fourier Transform/Inverse Fast Fourier Transform
  - FFT/IFFT are fast routines for computing DFT/IDFT
- We will use 256-point FFTs/IFFTs
  - This gives us 256 subcarriers
- We will use 240 subcarriers – 16 subcarriers are unused

Task #3

- Why not use all 256 subcarriers?
  - Because of DSP aliasing



- Some subcarriers on the edge are unused to reduce the self-interference

Task #3

- We will use subcarriers -120 through 119
- The first (or earliest) QPSK symbol goes to subcarrier -120, the last QPSK symbol in goes to subcarrier 119.
  - We are going to try using subcarrier 0 as is done in 5G cellular, but we may need to change this because of the B205-mini
- But doesn't the IDFT equation have subcarriers 0 through 255 for a 256-point IDFT?
  - Yes, but subcarrier -1 is the same as subcarrier 255, -2 is the same as subcarrier 254, etc.

$$e^{j2\pi nk/N} \longrightarrow \text{Let k = N-1} \longrightarrow e^{j2\pi n(N-1)/N} = e^{j2\pi n}e^{-j2\pi n/N}$$
$$= e^{-j2\pi n/N}$$

Task #3

- We can change the IDFT equation to an equivalent form to help think about the negative frequencies

$$x[n] = \frac{1}{N} \sum_{k=-N/2}^{N/2-1} X[k] e^{j2\pi nk/N} \qquad \text{for } 0 \le n \le N-1$$

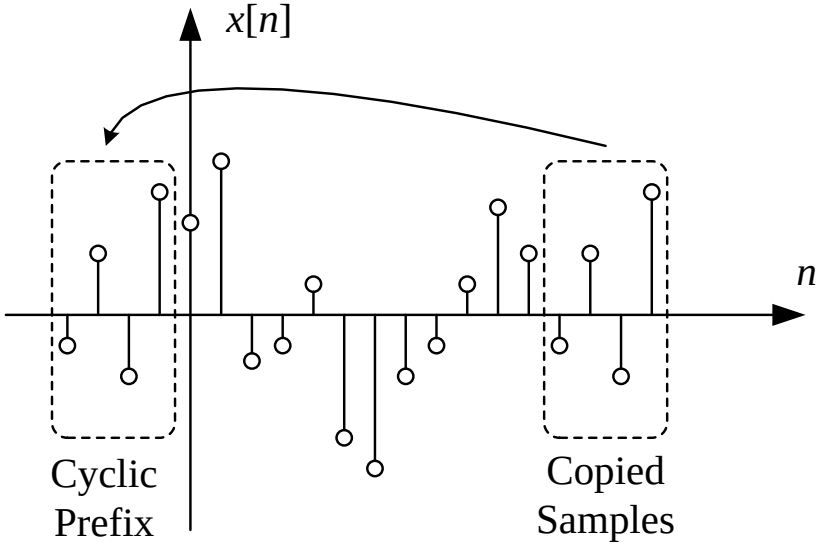
- The subcarriers where $N/2 \le k \le N-1$ are mathematically equivalent to $-N/2 \le k \le -1$.
- Our mathematical illustration does not change how the FFT/IFFT function, we just need to remember the above bullet and place the data appropriately.

Presentation Outline

- Digital Signal Processing
  - Sampling/analog-to-digital conversion
  - Aliasing
  - Frequency constraints
  - Digital-to-analog conversion
- Frequency Analysis
  - Fourier Transforms/Inverse Fourier Transforms
  - Discrete Fourier Transforms/Inverse Discrete Fourier Transforms
- Orthogonal Frequency Division Multiplexing (OFDM) Modulation
  - IDFT Modulation
  - Orthogonality
  - DFT Demodulation
  - Cyclic Prefix
- Task #3
- Task #4

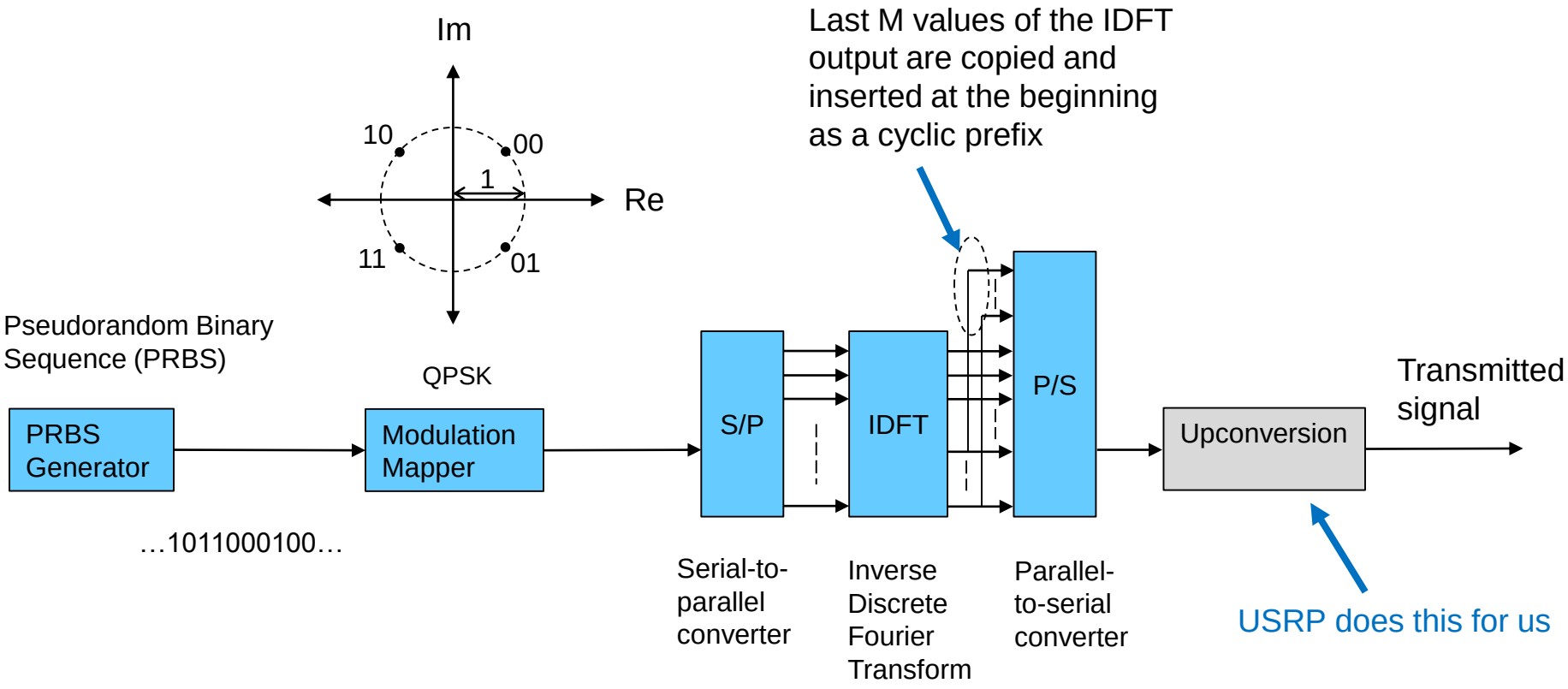
Task #4

- Add/remove a cyclic prefix at the transmitter/receiver, respectively
- The cyclic prefix will be 18 samples long
- The 18 samples come from the last 18 samples computed by the IDFT at the transmitter like is shown in this diagram



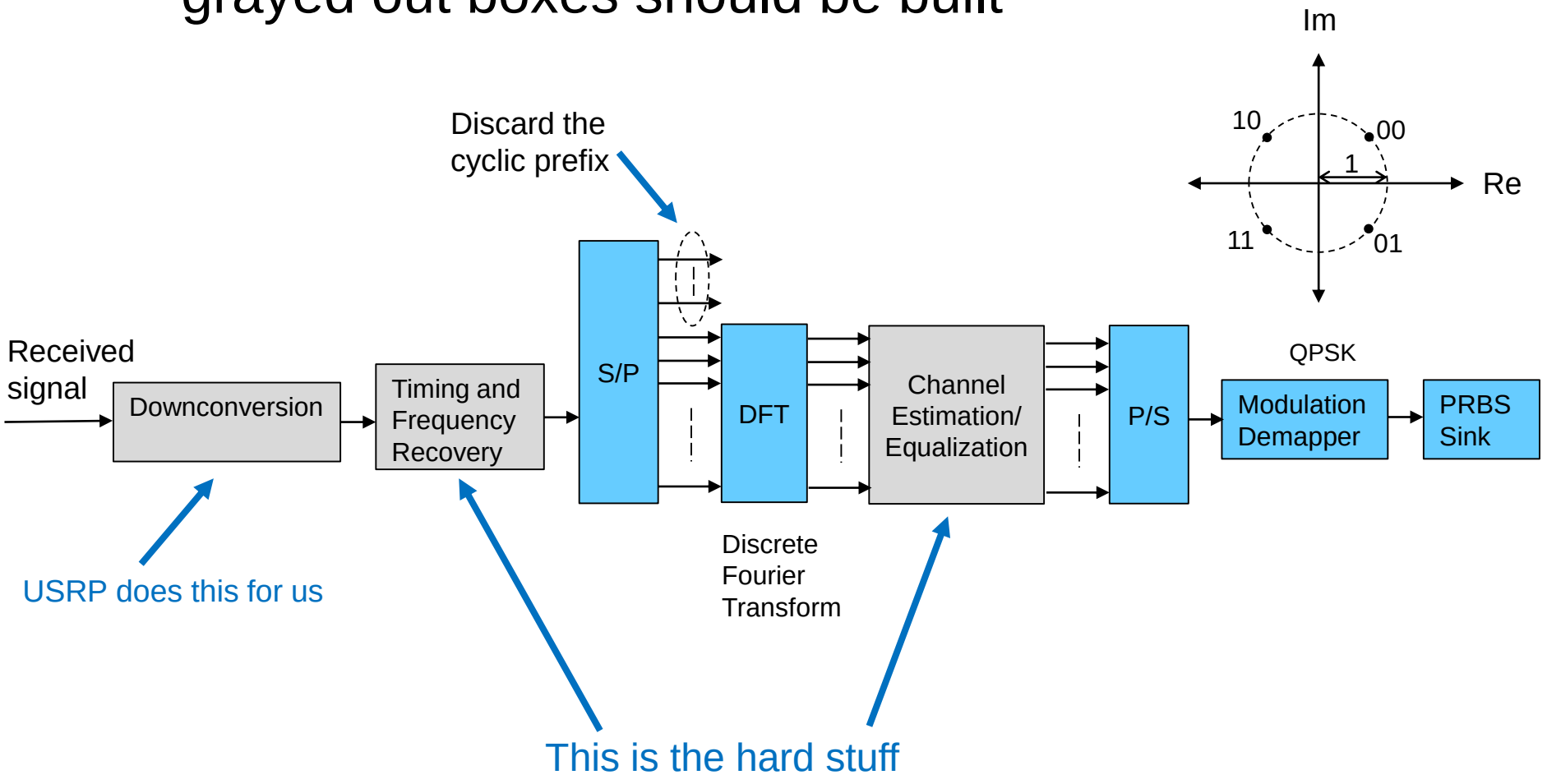
Task #4

- At the completion of task #4, all of the non-grayed out boxes should be built



Task #4

- At the completion of task #4, all of the non-grayed out boxes should be built



Remaining Tasks

- The remaining tasks (5-7) will be covered later and/or elsewhere.