

# 6.S965

# Digital Systems Laboratory II

## Lecture 9:

# Administrative

- Week 4 is due tonight (sorry typo again)
- Week 5 is out, due Friday

# Rising/Falling Edge Debate

- I think we've settled on driving and measuring on the falling edge of our AXI CLK signal.
- The rationale:
  - The modules we care about all sample and respond on the rising edge.
  - If we have our monitors sample on the rising edge (and then wait for `ReadOnly()`), we won't necessarily have seen the conditions that the device will have seen
  - Waiting for the falling edge (and then for `ReadOnly()`) allows us to see what will be experienced by the upcoming rising edge, including any changes we might have made with our own drivers (Since we are awaiting falling edge on those too)
- So we're not really doing anything on the falling edge...we're using it as a spot from which to observe the rising edge only (it is a way to get a pre-rising edge position)

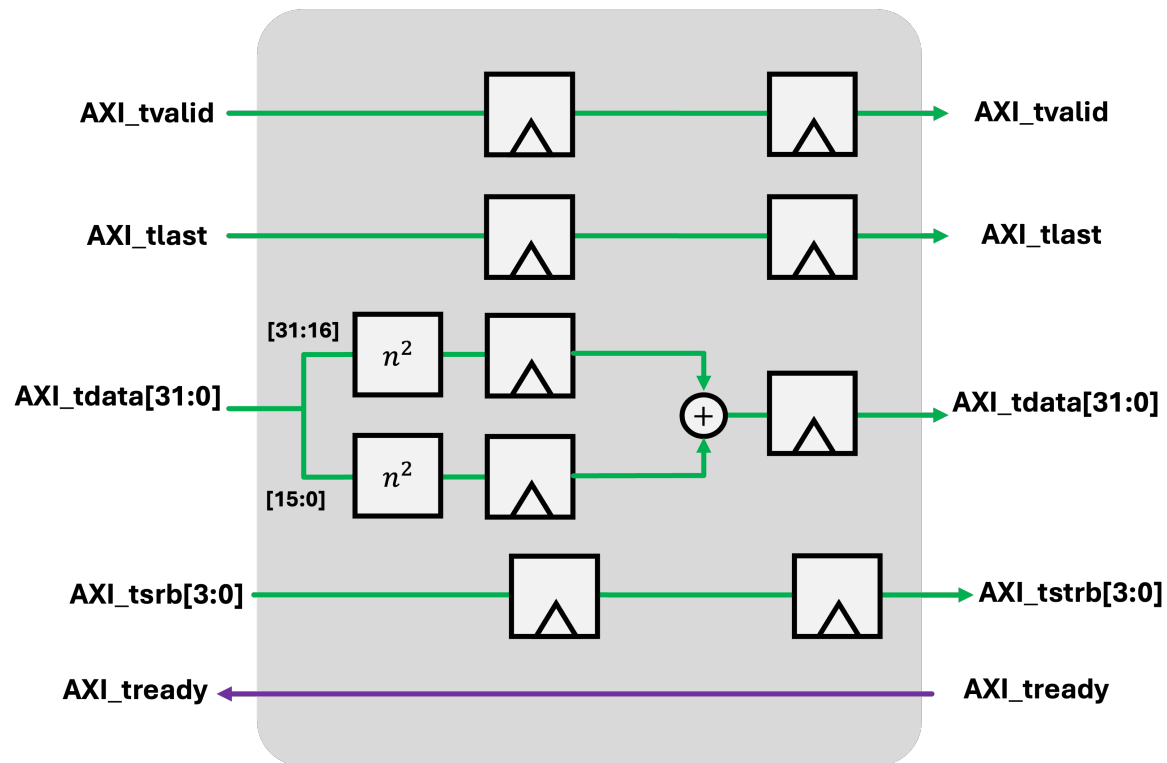
# TREADY

# Week 4 Split-Square-Sum

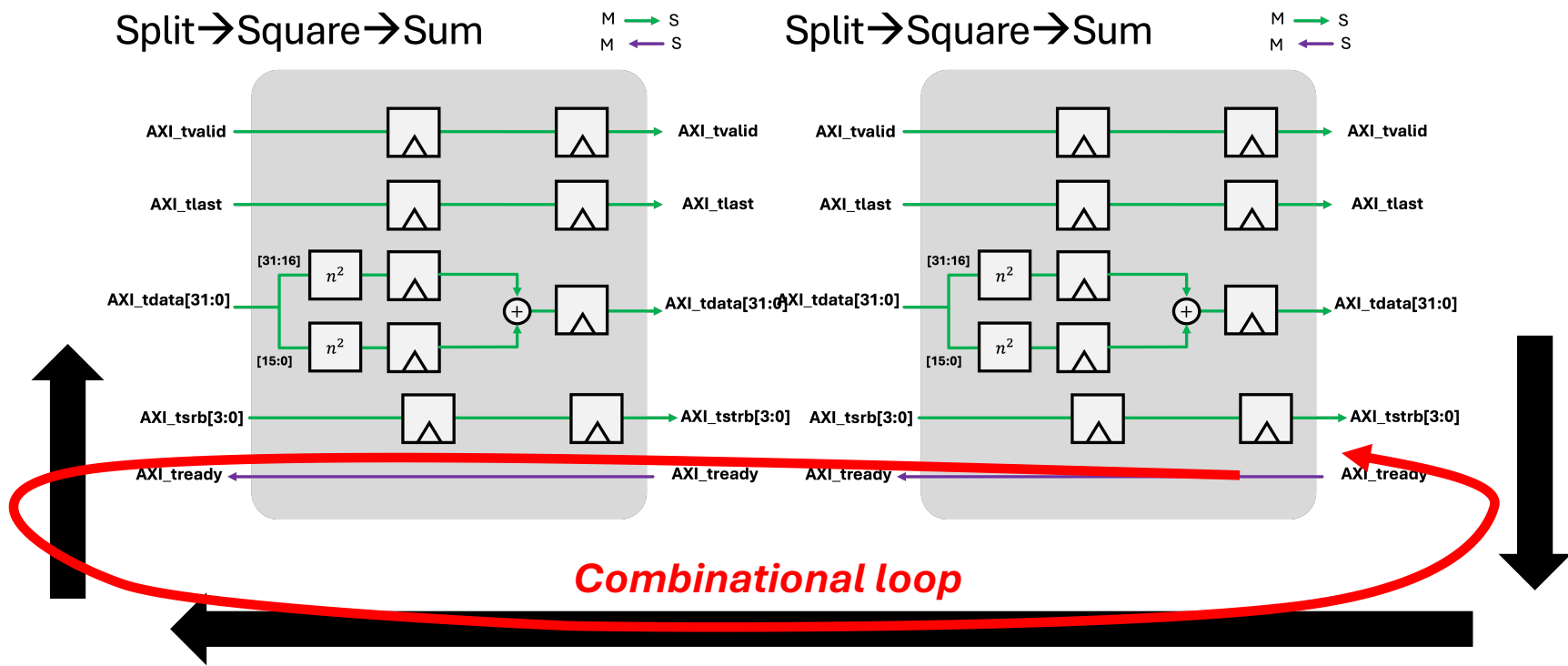
- Any Problems?

Split → Square → Sum

M → S  
M ← S



# Add into a feedback path or something...

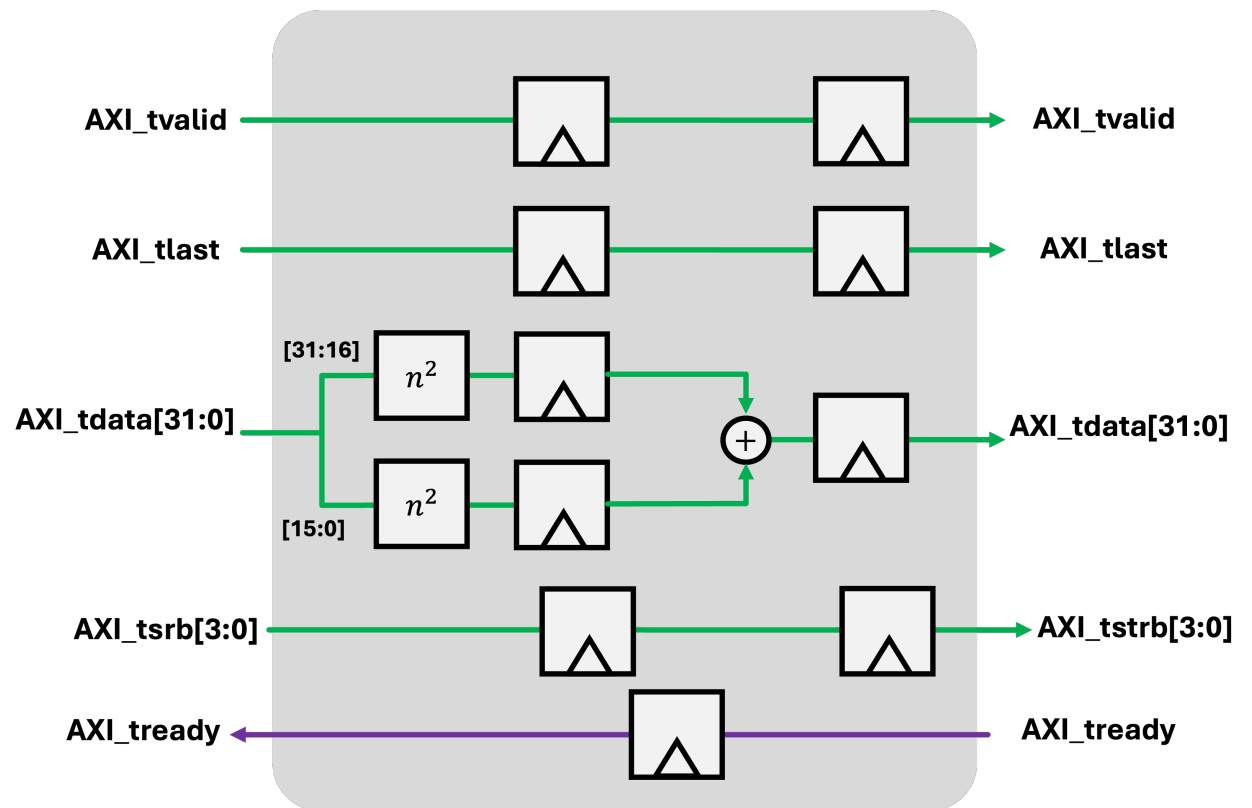


# Actually we should do

- Add a register on the TREADY pipe
- Any problems with this?

Split → Square → Sum

M → S  
M ← S



# Delaying TREADY

- Delaying the ability to convey a halt (via TREADY) to any upstream device means that there's a delay in stopping that data.
- It has to go somewhere/get absorbed somewhere
- Need a buffer/some sort of very short-form fifo
- You'll hear these called “skid buffers” or “Carlioni Buffers”

<https://ptolemy.berkeley.edu/projects/embedded/research/hsc/class.F02/ee249/lectures/lipClass.pdf>



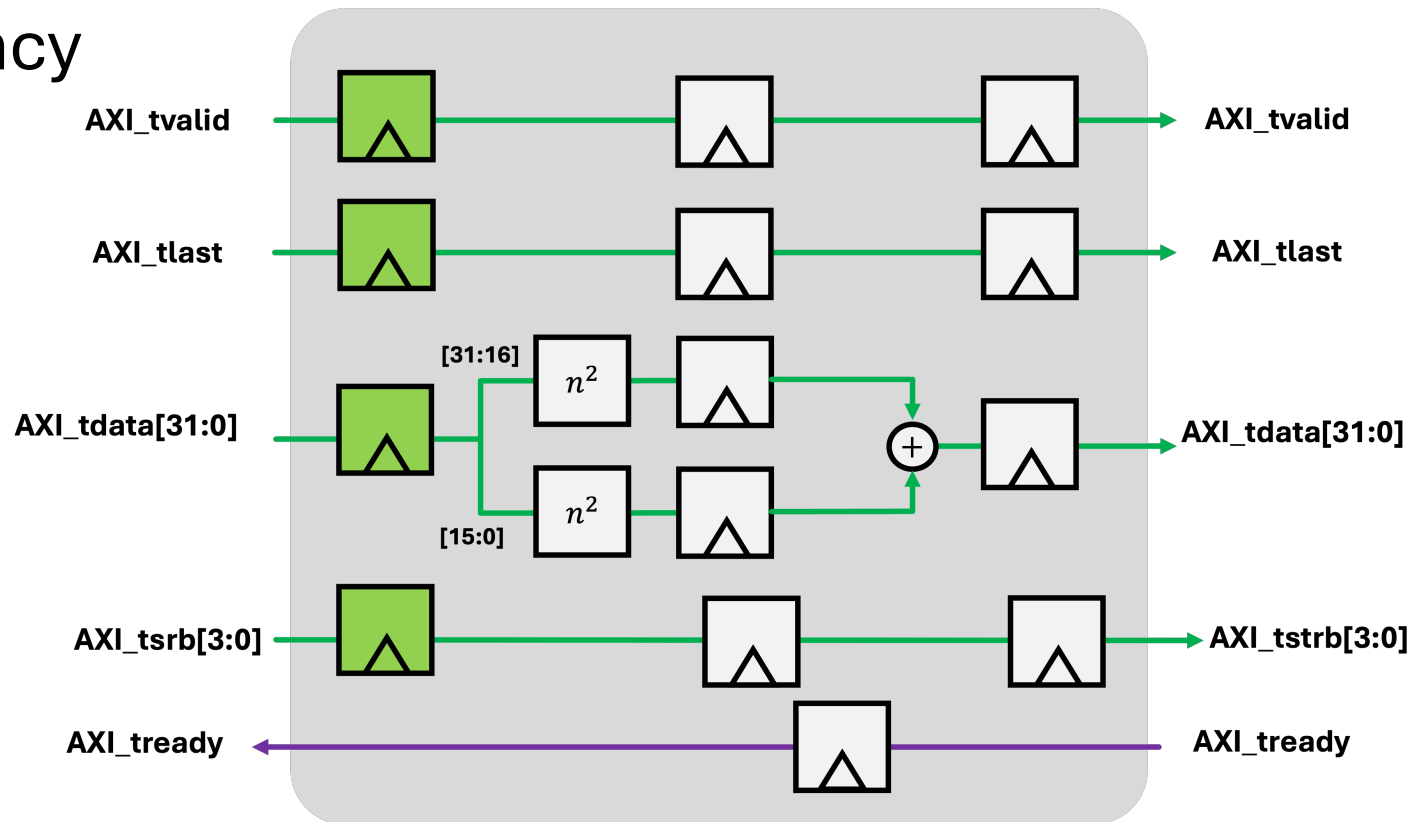
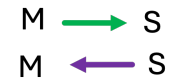
# What is a Skid Buffer?

- A device that “eats”/temporarily holds data in the event of the data pipeline having to suddenly slam on the brakes.
- Therefore the system “skids” to a halt.

# Just add more buffers?

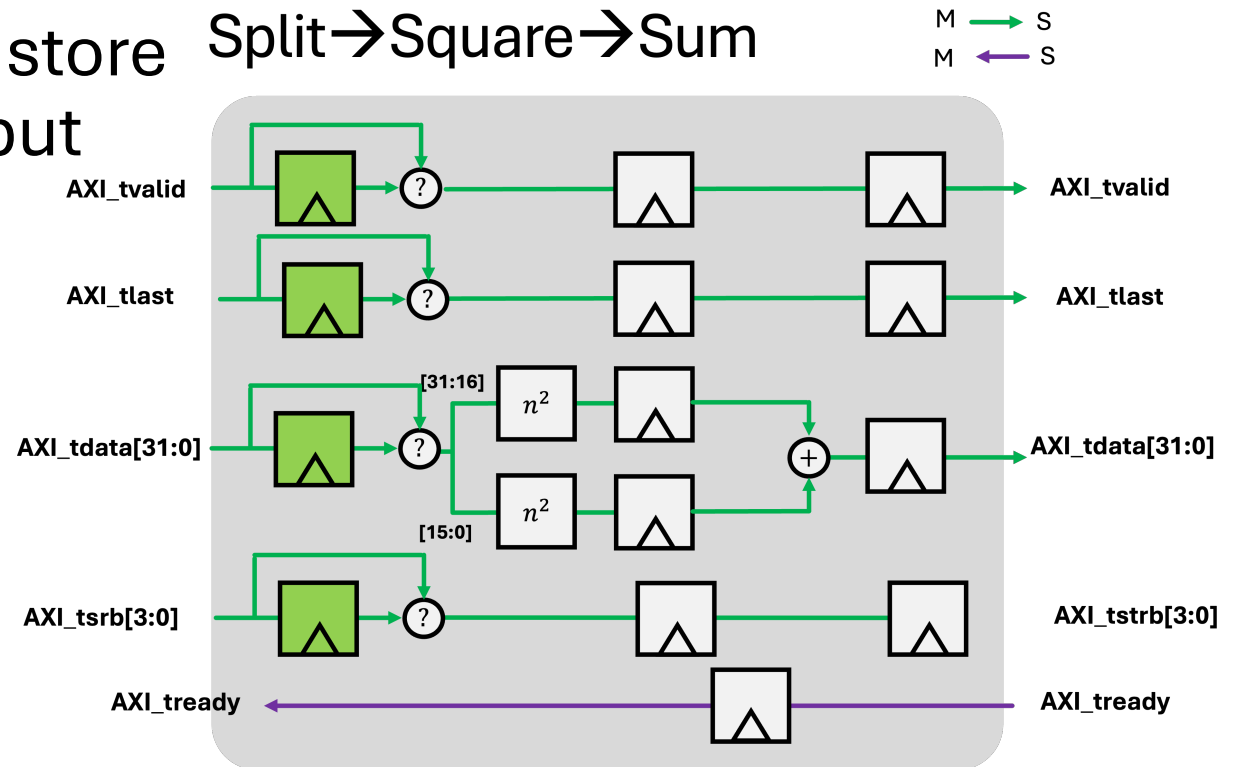
- No this just adds latency

Split → Square → Sum



# More complicated than that

- Need something that will selectively let data through or store it based on output





# Skid Buffers/FIFOs

- Looking back, we probably should have mandated you put those in, but I had kinda hoped to keep things simple
- We'll add one in next week/final week.
- Further reading:
  - <https://zipcpu.com/blog/2019/05/22/skidbuffer.html>
  - [https://fpgacpu.ca/fpga/Pipeline\\_Skid\\_Buffer.html](https://fpgacpu.ca/fpga/Pipeline_Skid_Buffer.html)

# More Modulation Stuff

# AM Modulation (The original modulation)

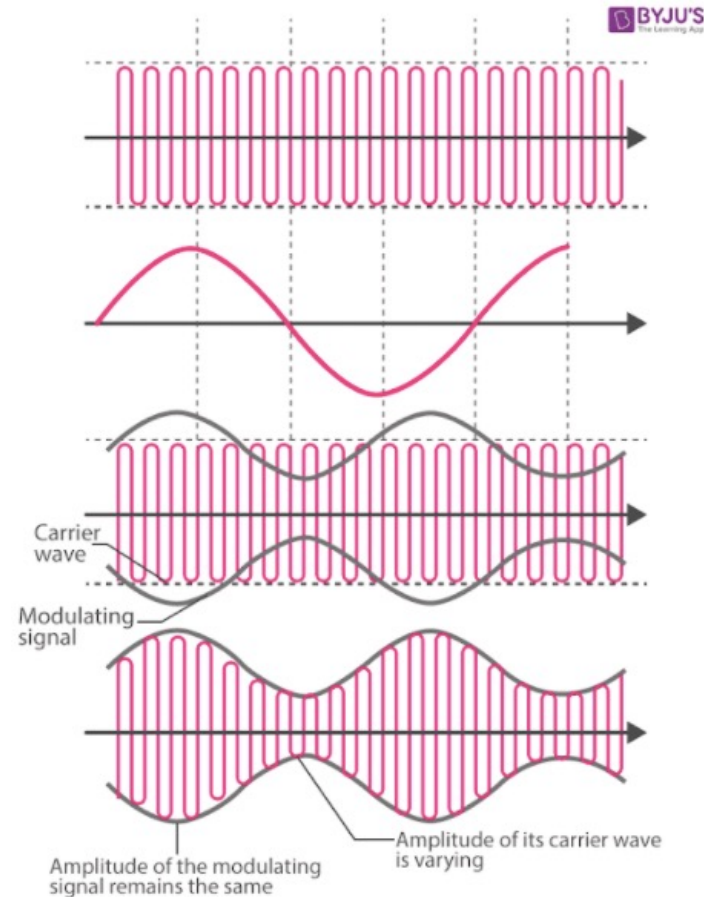
- This was really the first way that people did modulation of any sort of scale

$$v(t) = A(t) \cos(2\pi f(t)t + \phi(t))$$

- Fix  $f$
- Fix  $\phi$
- Vary  $A$  over time to convey information

# Amplitude Modulation

- Keep frequency the same and then modulate then modulate the amplitude of your carrier wave...
- Usually something as simple as a low-pass filter and some non-linearity can get the info out



<https://byjus.com/jee/amplitude-modulation/>

© Byjus.com



# AM Modulation (The original modulation)

- Let's just assume  $\phi$  is 0 for sake of study.

$$v(t) = A(t) \cos(2\pi f t + \phi)$$

- Vary  $A$  over time to convey information
- Any time varying signal  $A(t)$  is itself comprised of sinusoids...

# AM Modulation

- So when we study how AM works

$$v(t) = A(t) \cos(2\pi f t)$$

- Is really:

$$v(t) = A \cos(2\pi f_m t) \cos(2\pi f_c t)$$

- So when we study this, we're really asking how/what is created from this situation

# Trig identities to rescue again

- It can be shown that...

## Ptolemy's theorem [ edit ]

Main article: [Ptolemy's theorem](#)

See also: [History of trigonometry § Classical antiquity](#)

Ptolemy's theorem is important in the history of trigonometric identities, as it is how results equivalent to the sum and difference formulas for sine and cosine were first proved. It states that in a cyclic quadrilateral  $ABCD$ , as shown in the accompanying figure, the sum of the products of the lengths of opposite sides is equal to the product of the lengths of the diagonals. In the special cases of one of the diagonals or sides being a diameter of the circle, this theorem gives rise directly to the angle sum and difference trigonometric identities.<sup>[17]</sup> The relationship follows most easily when the circle is constructed to have a diameter of length one, as shown here.

By [Thales's theorem](#),  $\angle DAB$  and  $\angle DCB$  are both right angles. The right-angled triangles  $DAB$  and  $DCB$  both share the hypotenuse  $\overline{BD}$  of length 1. Thus, the side  $AB = \sin \alpha$ ,  $AD = \cos \alpha$ ,  $BC = \sin \beta$  and  $CD = \cos \beta$ .

By the [inscribed angle](#) theorem, the central angle subtended by the chord  $\overline{AC}$  at the circle's center is twice the angle  $\angle ADC$ , i.e.  $2(\alpha + \beta)$ . Therefore, the symmetrical pair of red triangles each has the angle  $\alpha + \beta$  at the center. Each of these triangles has a hypotenuse of length  $\frac{1}{2}$ , so the length of  $\overline{AC}$  is  $2 \times \frac{1}{2} \sin(\alpha + \beta)$ , i.e. simply  $\sin(\alpha + \beta)$ . The quadrilateral's other diagonal is the diameter of length 1, so the product of the diagonals' lengths is also  $\sin(\alpha + \beta)$ .

When these values are substituted into the statement of Ptolemy's theorem that  $|\overline{AC}| \cdot |\overline{BD}| = |\overline{AB}| \cdot |\overline{CD}| + |\overline{AD}| \cdot |\overline{BC}|$ , this yields the angle sum trigonometric identity for sine:  $\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta$ . The angle difference formula for  $\sin(\alpha - \beta)$  can be similarly derived by letting the side  $\overline{CD}$  serve as a diameter instead of  $\overline{BD}$ .<sup>[17]</sup>

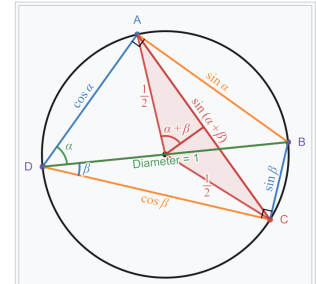


Diagram illustrating the relation between Ptolemy's theorem and the angle sum trig identity for sine. Ptolemy's theorem states that the sum of the products of the lengths of opposite sides is equal to the product of the lengths of the diagonals. When those side-lengths are expressed in terms of the sin and cos values shown in the figure above, this yields the angle sum trigonometric identity for sine:  $\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta$ .

$$\cos(\alpha) \cos(\beta) = \frac{\cos(\alpha + \beta)}{2} + \frac{\cos(\alpha - \beta)}{2}$$

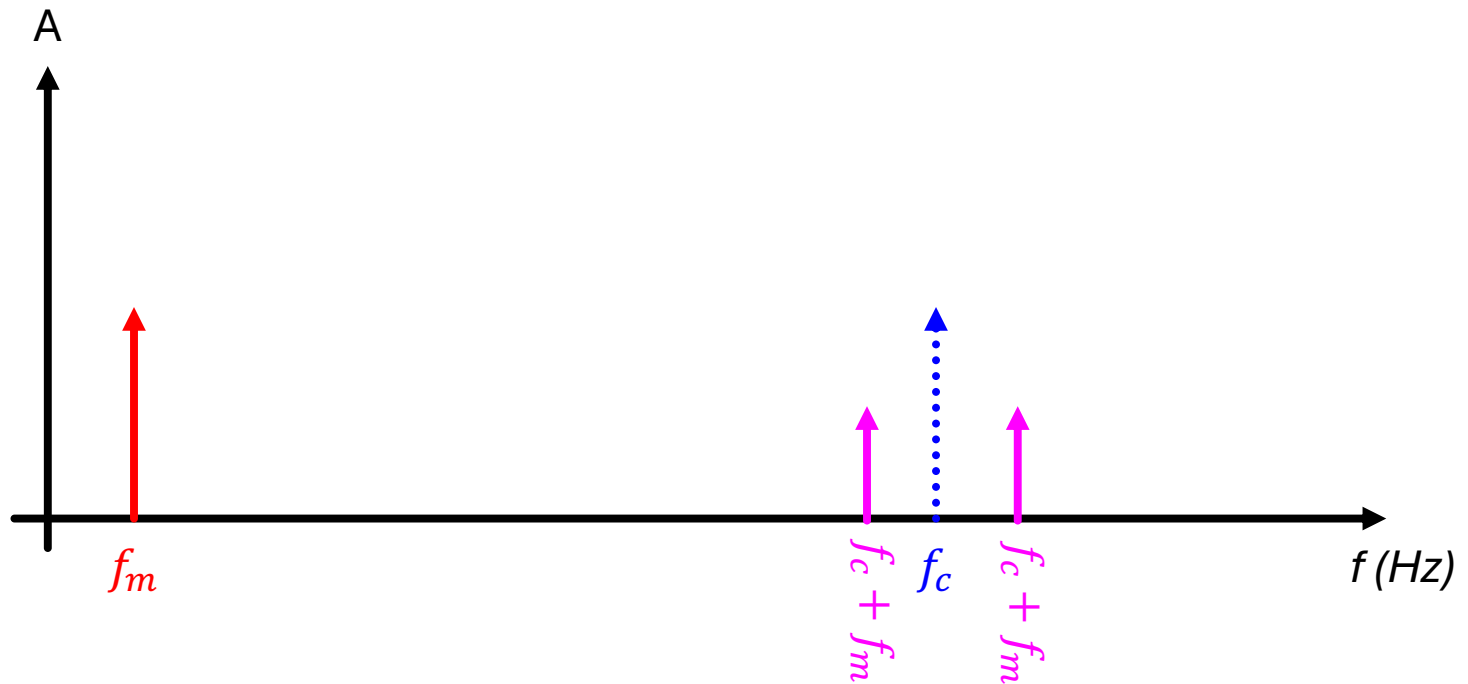
- This is probably one of the most important trig identities in all of EECS

# For AM

- Signal is  $v(t) = A \cos(2\pi f_m t) \cos(2\pi f_c t)$
- Which means if you amplitude modulate a sine wave onto your carrier how many sinusoids will you generate?
- Two:
- $$v(t) = \frac{A \cos(2\pi(f_c + f_m)t)}{2} + \frac{A \cos(2\pi(f_c - f_m)t)}{2}$$

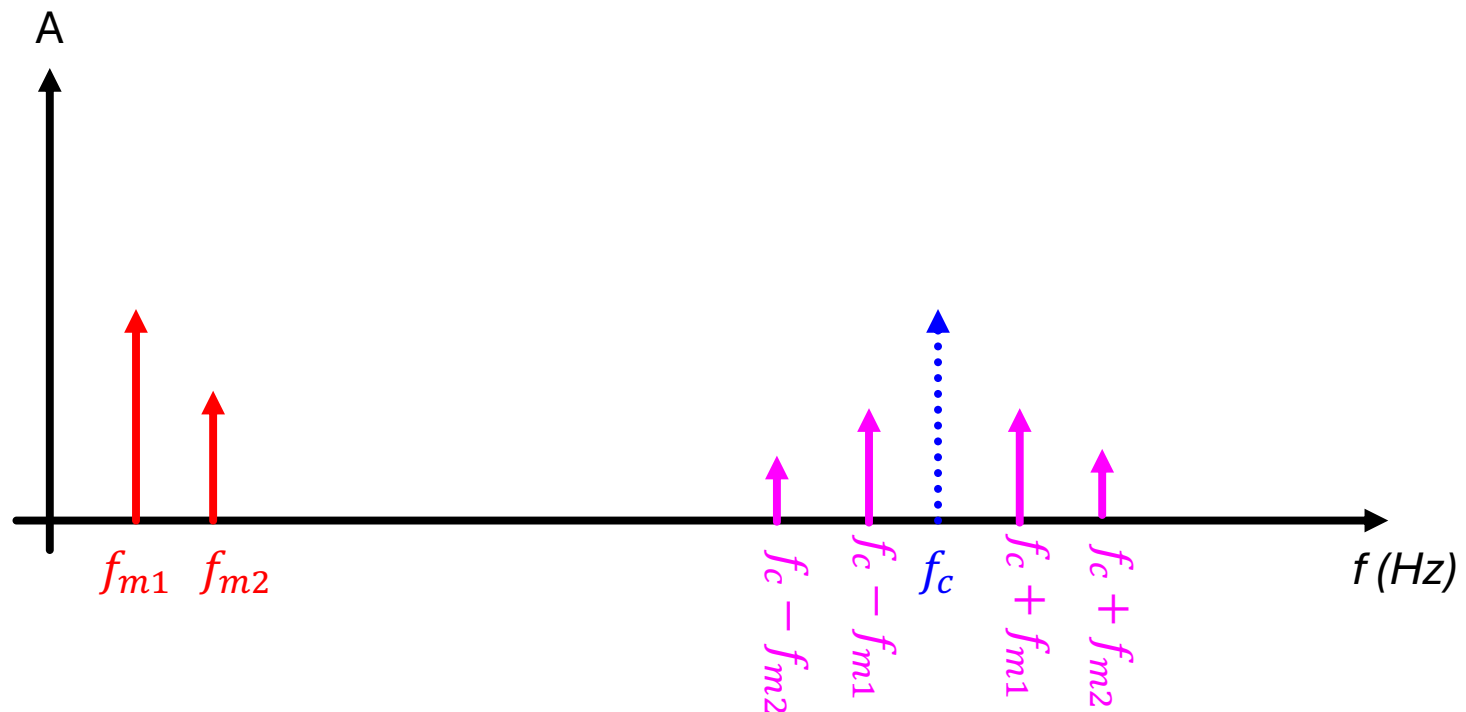
# Resulting Frequency Spectrum

- Assume  $f_m$  is much lower than  $f_c$



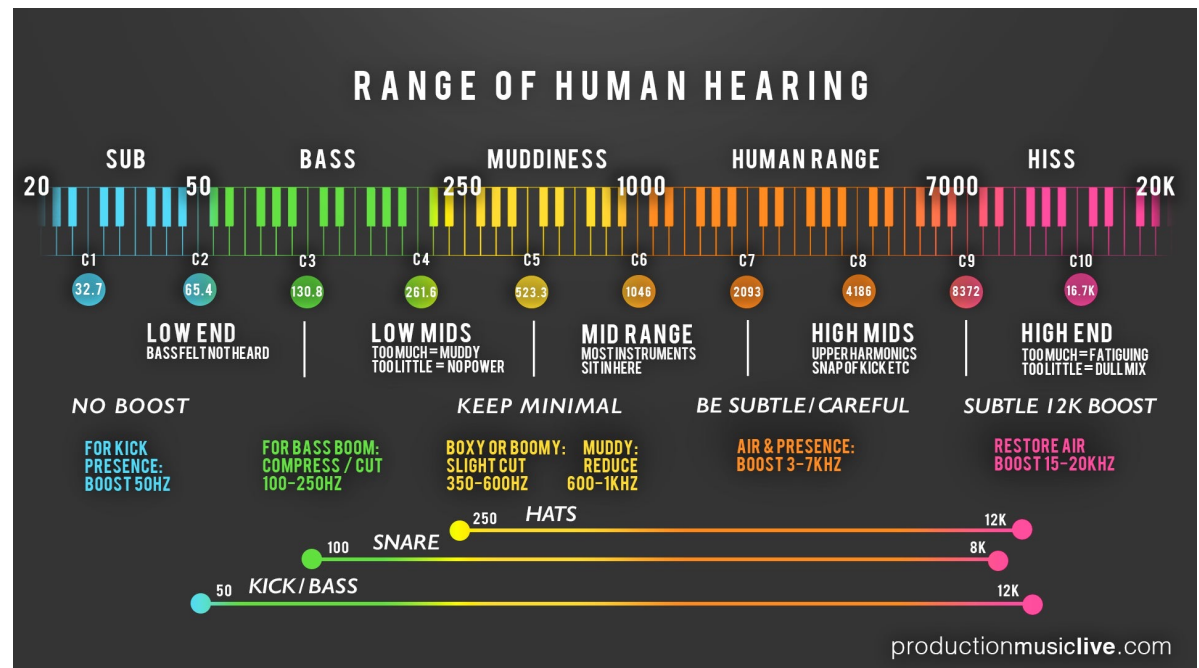
# Resulting Frequency Spectrum

- What if you sent two different tones?



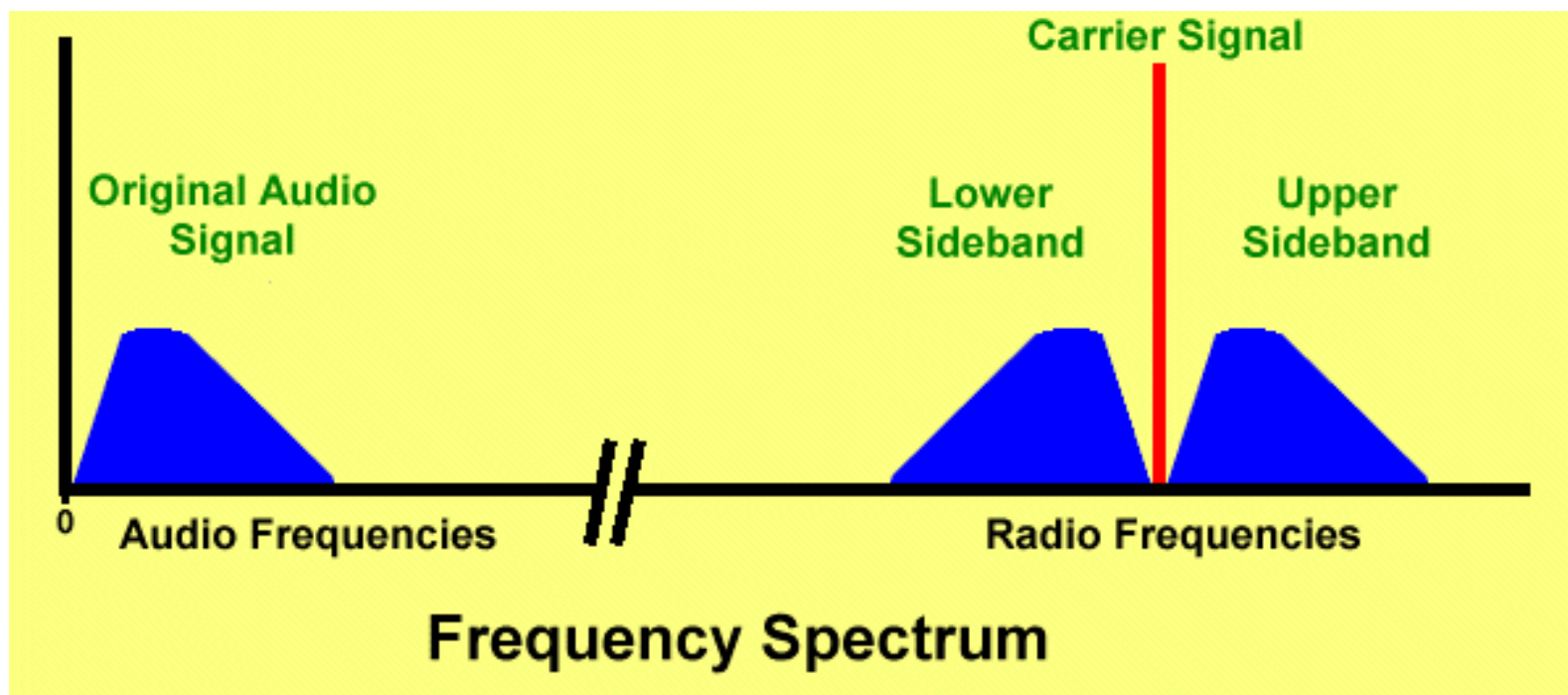
# Generalize

- If your signal is made up of tons of frequencies (such as audio/speech and/or data...)
- This same pattern will appear



# Therefore...

- In order to transmit information using AM
- **2X** the bandwidth of your signal must be used.





# AM...

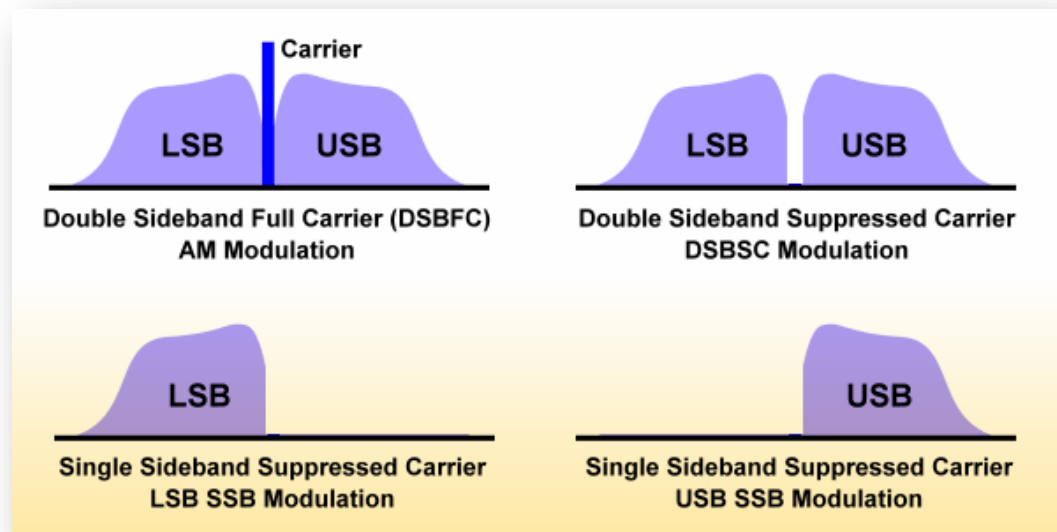
- AM transmits redundant information.
- The fact that it sends two signals for any given one input signal (at a specific frequency) means
- Bandwidth is directly correlated with the information you can send. If you use bandwidth inefficiently,

# AM benefits

- The double bandwidth of a standard AM signal does make demodulating it very, very easy.
- Those two signals mean that you *always* have a signal at the center frequency  $f_c$  and that makes it very easy to isolate and track.
- In fact you can demodulate it with basically anything...you just need a little bit of non-linearity from something and you can demodulate and get the signal back.

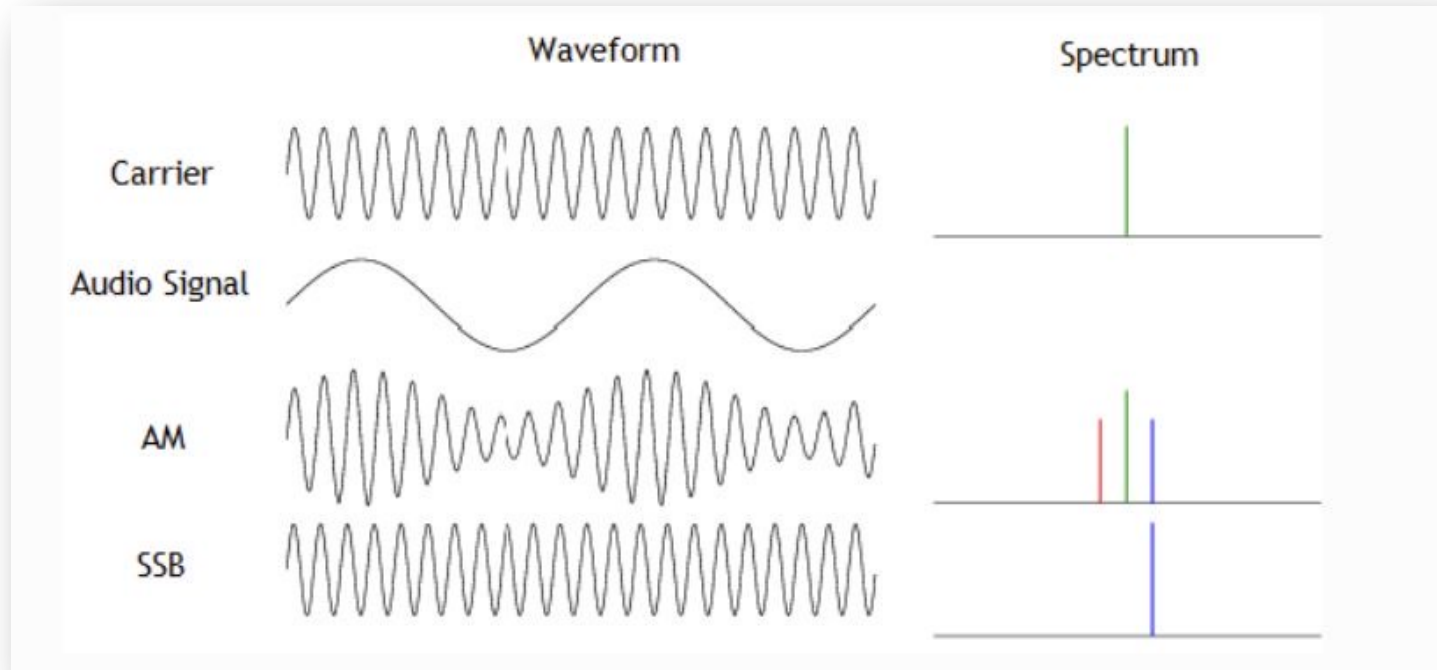
# Single-Sideband

- Instead of sending both sidebands in regular AM, before you send, suppress/bury one of the sidebands



<https://vu2nsb.com/wp-content/uploads/2020/03/AM-sideband-Gen.png>

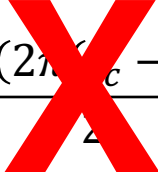
# SSB demodulation is tougher



<https://forum.audacityteam.org/t/what-is-ssb-modulation/66220>

# In SSB

- Let's say you're in USB

$$v(t) = \frac{A \cos(2\pi(f_c + f_m)t)}{2} + \frac{A \cos(2\pi(f_c - f_m)t)}{2}$$


- You can't just square this signal to separate out the modulation signal from the carrier
- In fact an SSB signal is just a time-varying frequency signal

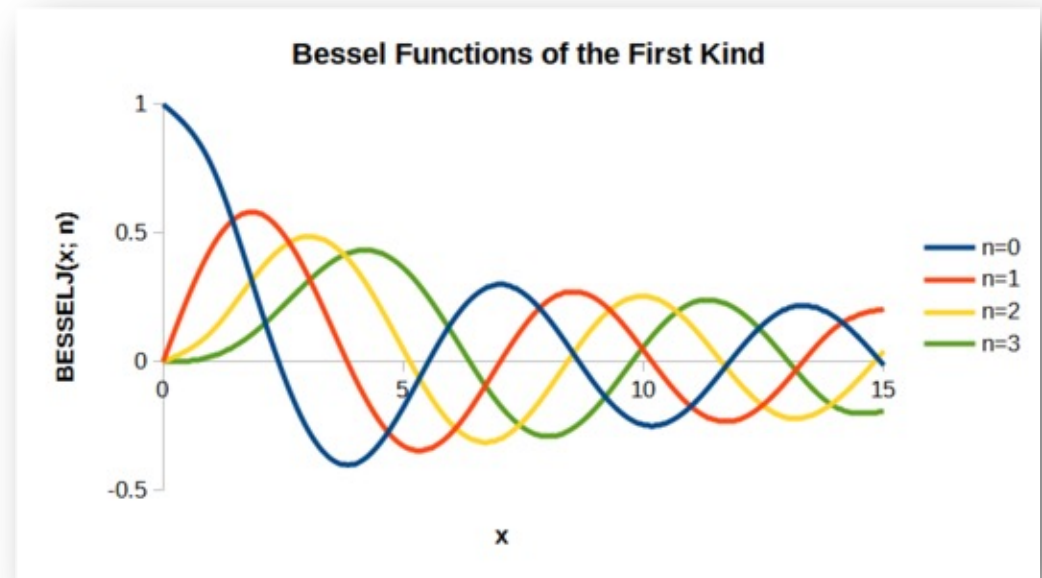
# FM and PM

- Both of these modulation schemes are built off of varying either  $f$  or  $\phi$  over time in our general sinusoid equation:

$$v(t) = A(t) \cos(2\pi f(t)t + \phi(t))$$

- This means in both cases, you'll be looking at math that takes on the shape of:
  - $\cos(\cos(t))$
- What class of functions describes these things?

# Bessel Functions!

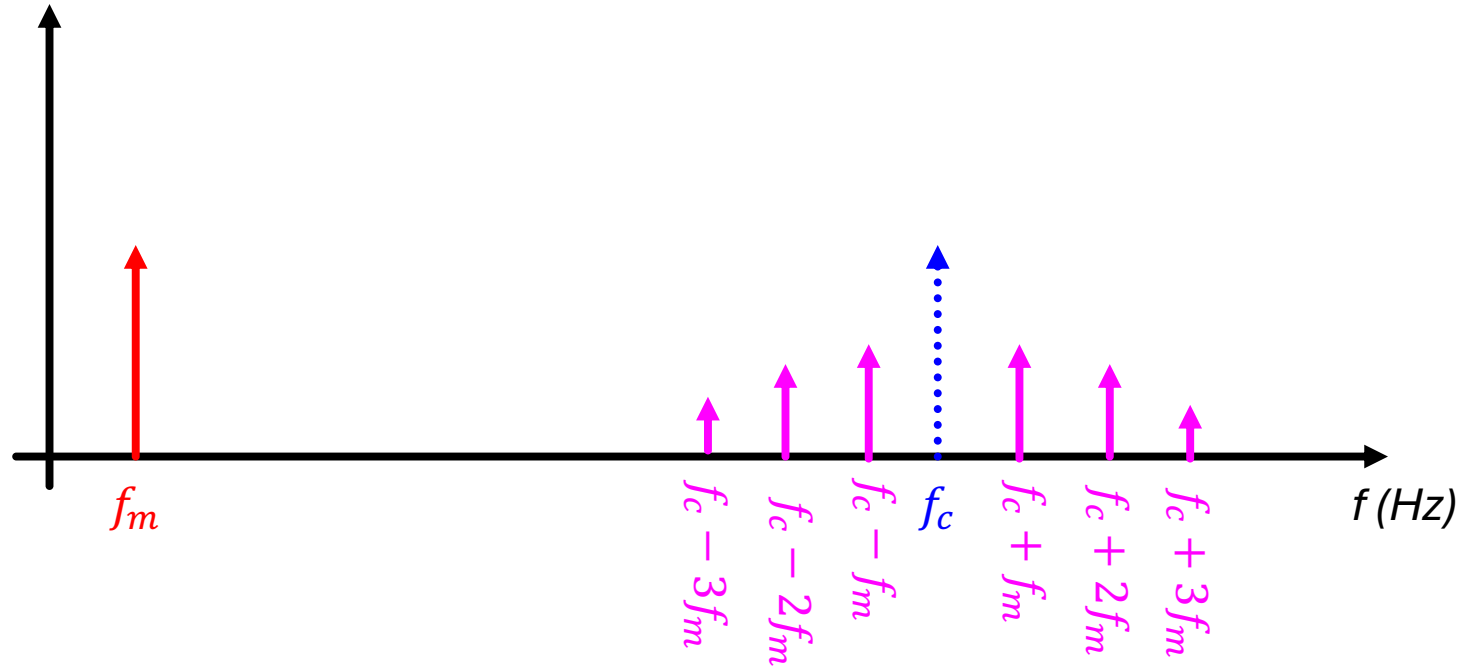


- Not the friendliest of equations
- See them a lot in RF/emag stuff

[https://en.wikipedia.org/wiki/Bessel\\_function](https://en.wikipedia.org/wiki/Bessel_function)

# Both FM and PM-only signals

- Will have numerous frequency components for every modulated frequency



<https://www.adsrsounds.com/fm8-tutorials/theory-behind-fm-synthesis/>



# FM and PM-only

- On their own these signals are harder to demodulate than regular AM
- Usually take up more bandwidth too (have the upper and lower sidebands)
- By nature of the math, you can't do SSB in FM or PM either!

# Is there a way to have something like AM but better?

- The math of AM is great, but the redundant bandwidth usage is annoying
- We get rid of one side band and it gets harder to demodulate
- Is there something else...?

# Any sine wave that is running...

$$v(t) = A(t) \cos(2\pi f t + \phi(t))$$

- Can be said to actually be made of two other sinusoidal waves:

$$\dots = \cos(2\pi f t) \cdot A(t) \cos(\phi(t)) - \sin(2\pi f t) \cdot A(t) \sin(\phi(t))$$

# I/Q signals

$$= \cos(2\pi ft) \cdot A(t) \cos(\phi(t)) - \sin(2\pi ft) \cdot A(t) \sin(\phi(t))$$

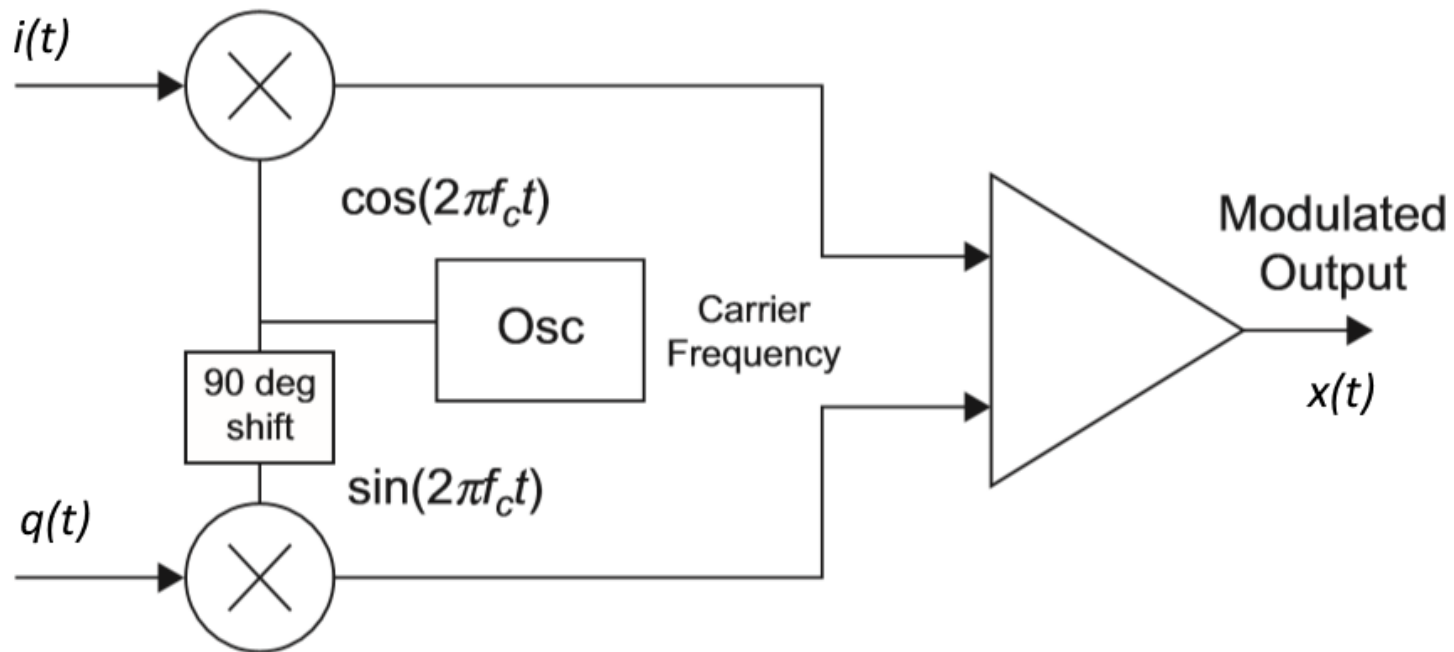
- You have two signals of fixed frequency and modulatable amplitudes.
- We don't need to think of those two amplitudes as related to the original signal, instead just think of them as modulatable signals on their own:

$$= \cos(2\pi ft) \cdot I(t) - \sin(2\pi ft) \cdot Q(t)$$

- This  $I(t)$  and  $Q(t)$  signal contain your information and can be useful in both creating and analyzing modulated signals.

# Quadrature Modulation

- Specify complex values over time



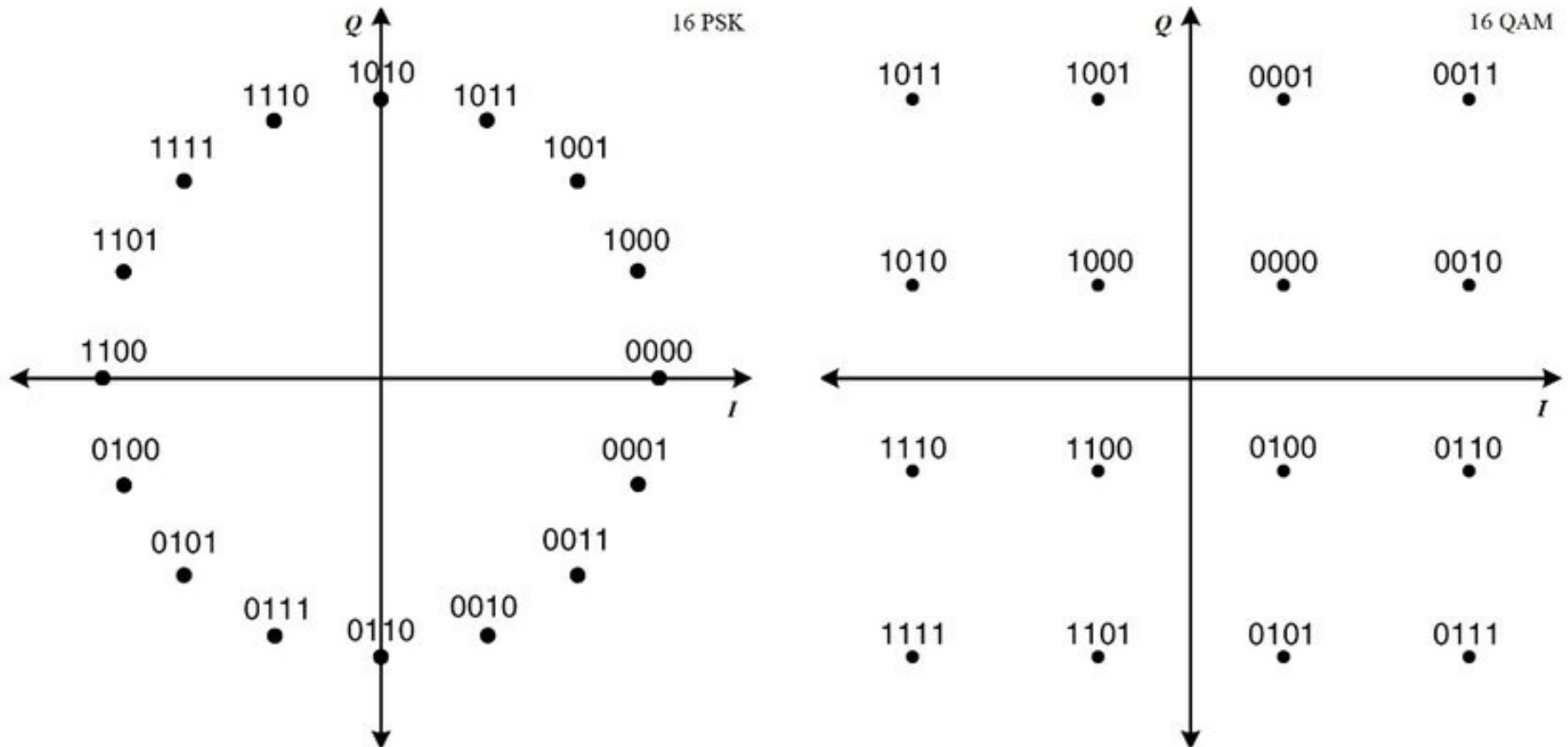
<https://www.edn.com/quadrature-modulation-the-signal-behind-digital-communications/>

# Dual Modulation

- The AM signal with the cosine and the AM signal with the sine cannot combine/destroy one another because they are 90 degrees out of phase
- Each half still takes up twice the bandwidth of its original signal, but we can superimpose them on top of each other in the frequency space.
- So we have 2X the  $\frac{1}{2}$  usage of before or basically full spectral usage.

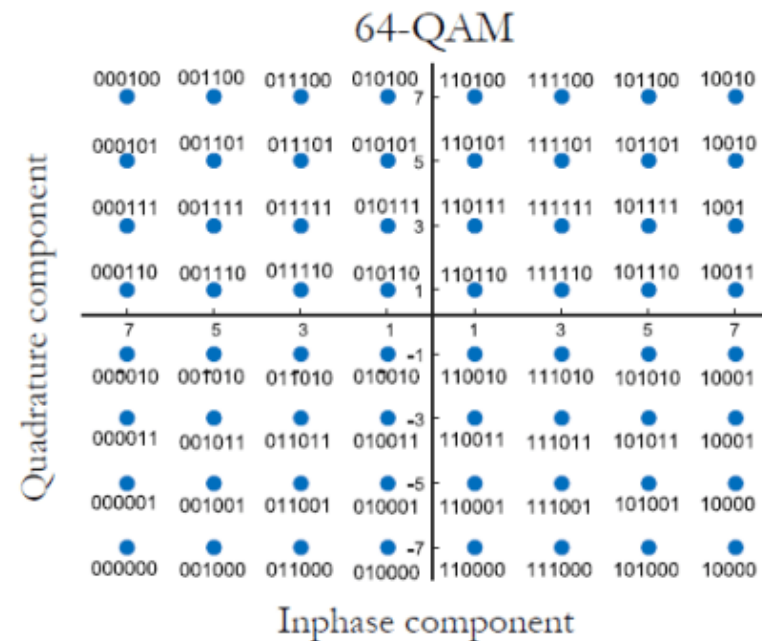
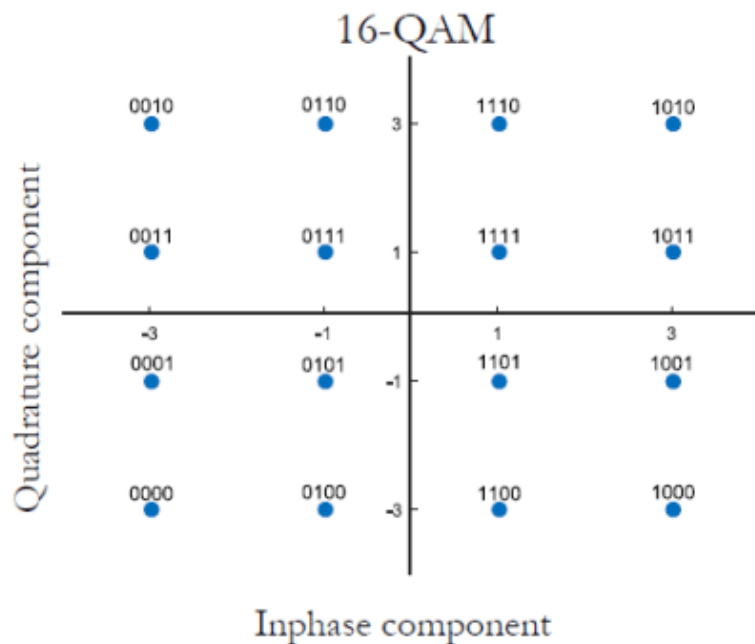
# As a result...

- When getting ready to transmit data for export you pack your bits and generate a complex number associated with them



# As a result...

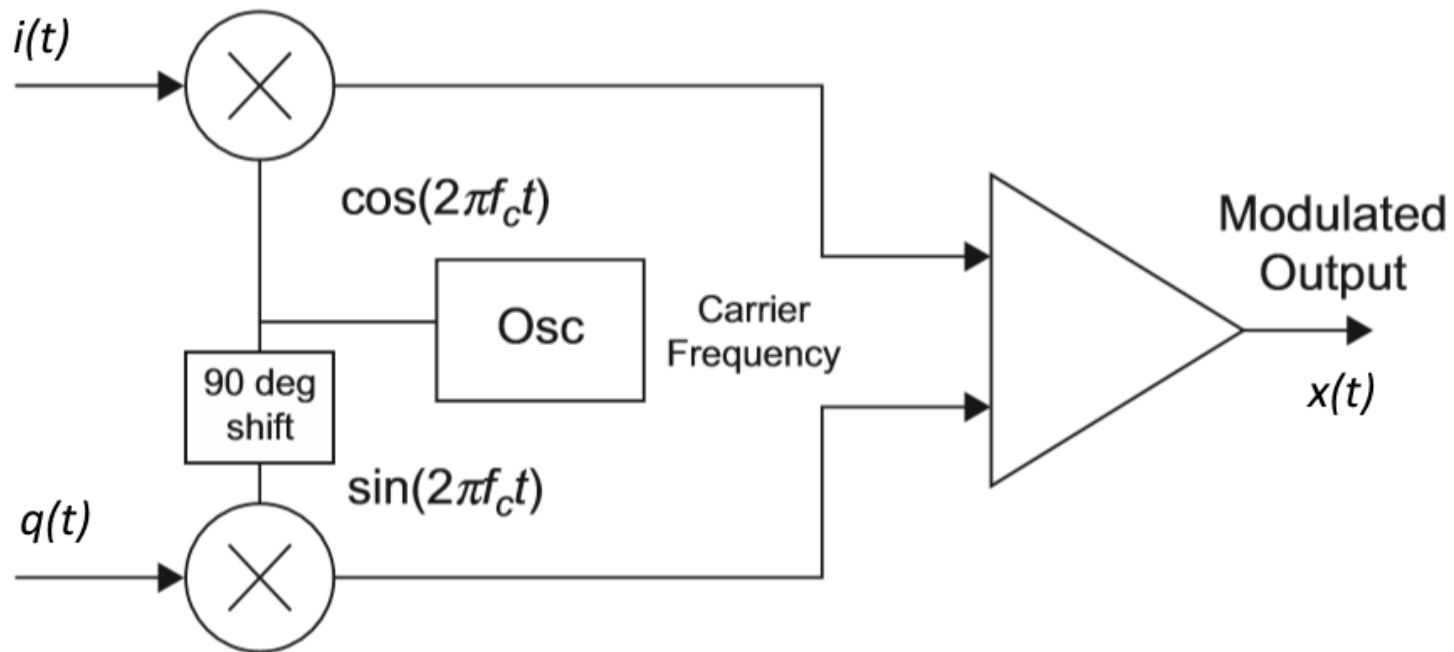
- When getting ready to transmit data for export you pack your bits and generate a complex number associated with them





# This is a Digital Upconverter (DUC)

- Specify complex values over time

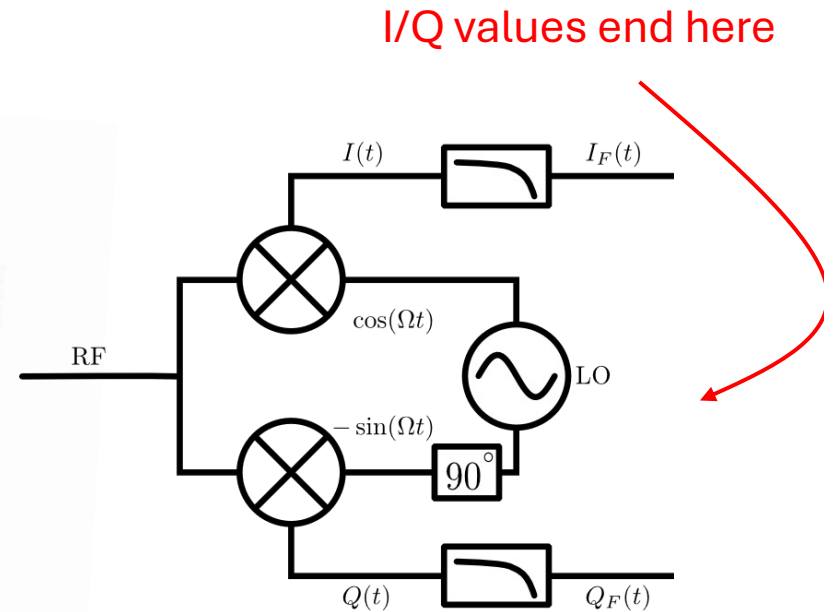
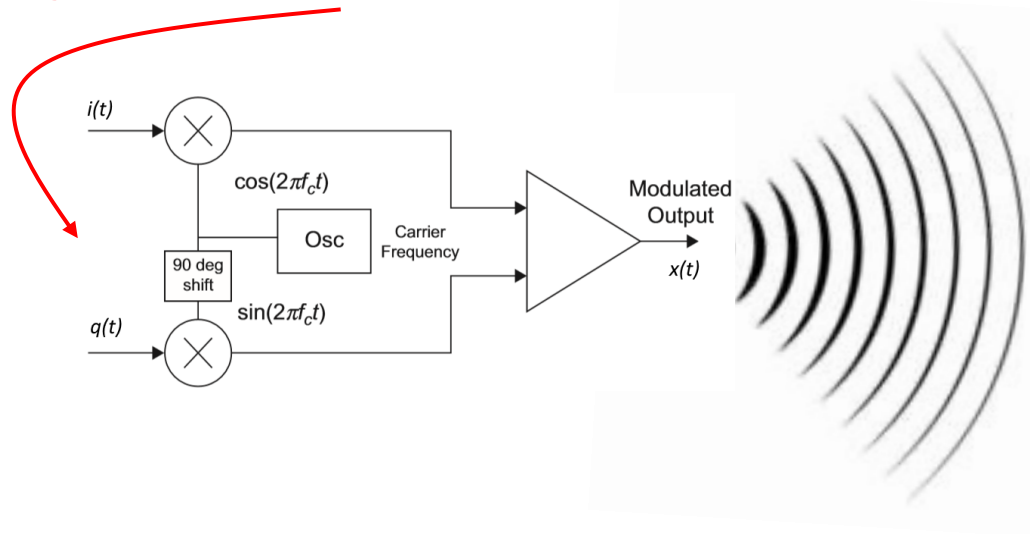


<https://www.edn.com/quadrature-modulation-the-signal-behind-digital-communications/>

# Travels through space

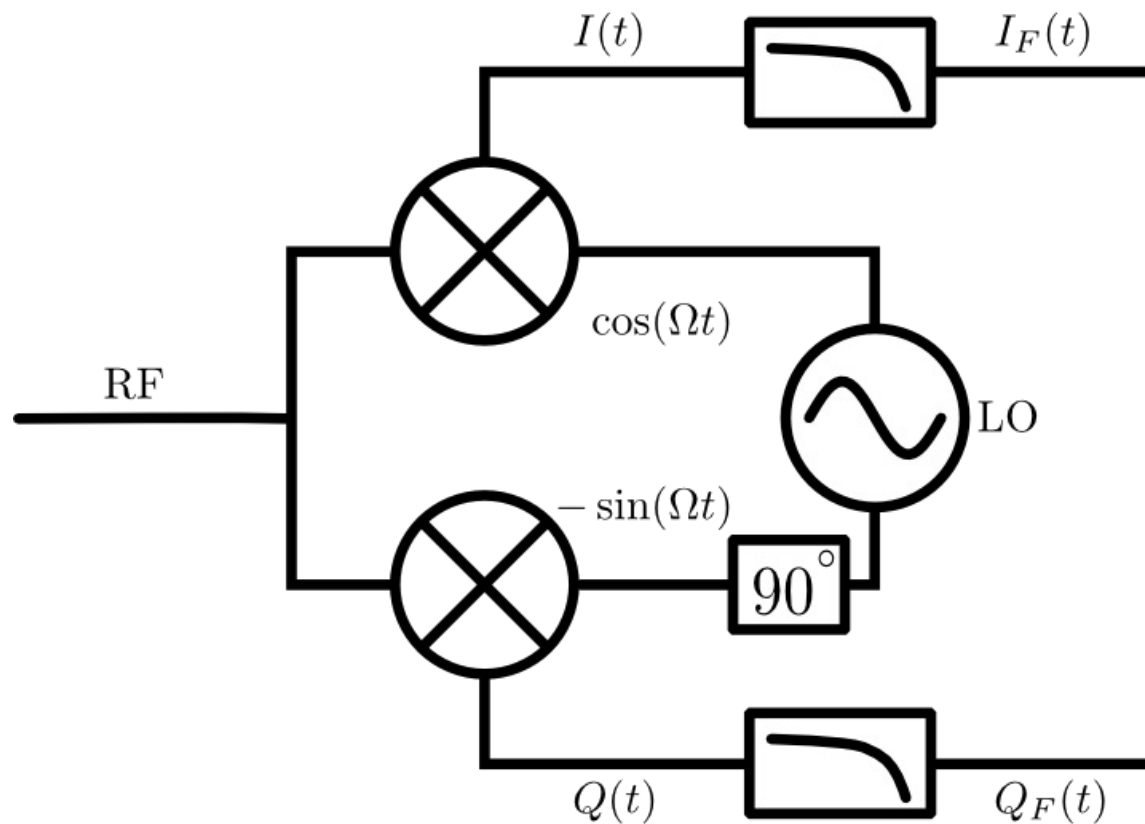
- Capture with antenna or something...

I/Q values start here



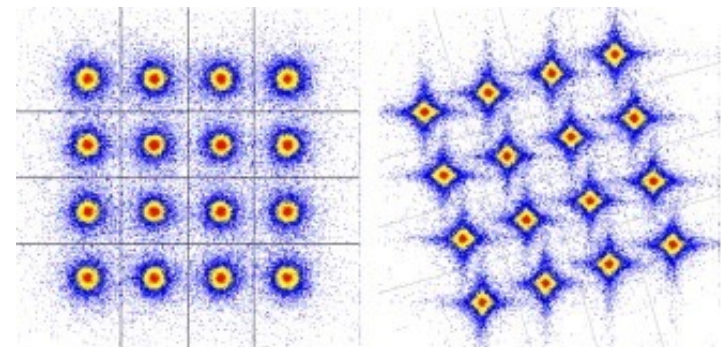
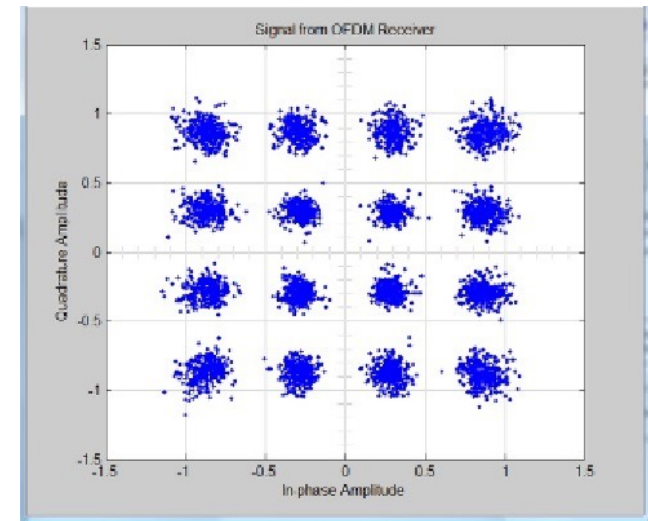
# On the receive side then...

- Basically do the same thing in reverse
- You'll build this in week 05



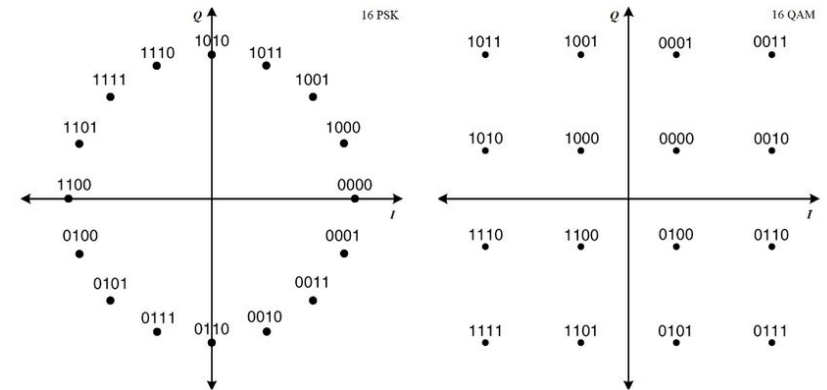
# Get IQ values out

- From those extract meaning...
- Isn't always clean



<https://www.analogictips.com/eye-and-constellation-diagrams-pt-2/>

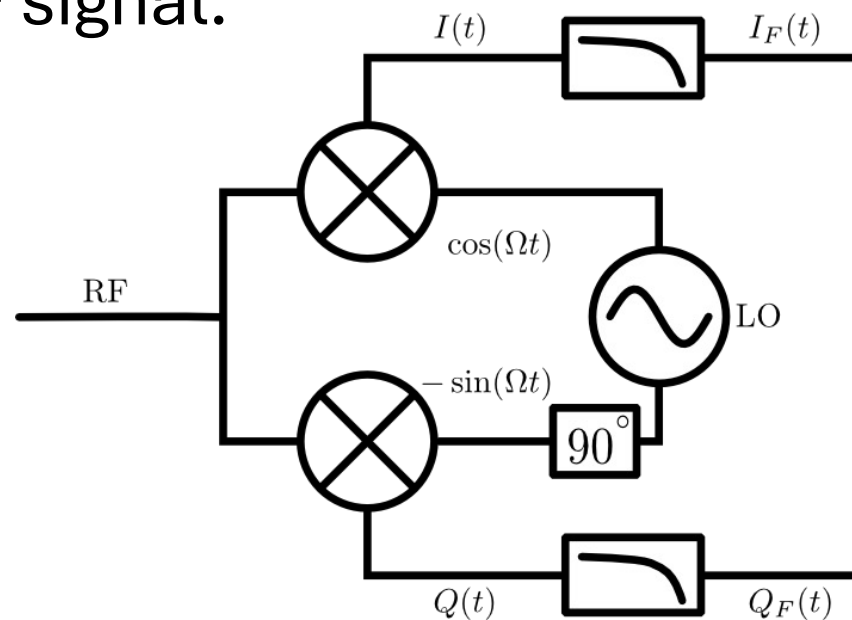
# QAM vs. QPSK



- Both done using same toolchain
- QAM is more complicated to decode (uses amplitude and phase)/Re/Imag location in I/Q plane
- QPSK really only needs phase, once aligned

# In Week 5

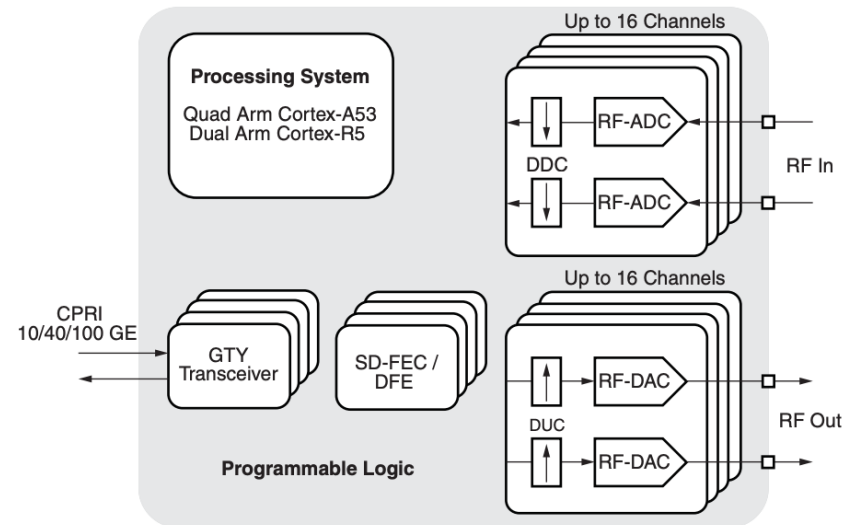
- You'll make your own form of DDC (Digital Down Converter) that generates the IQ signal from a single series of real-time information like would be captured on an RF signal.



# RFSoc Capabilities

- DDC == “Digital Down Converter”
- DUC == “Digital Up Converter”

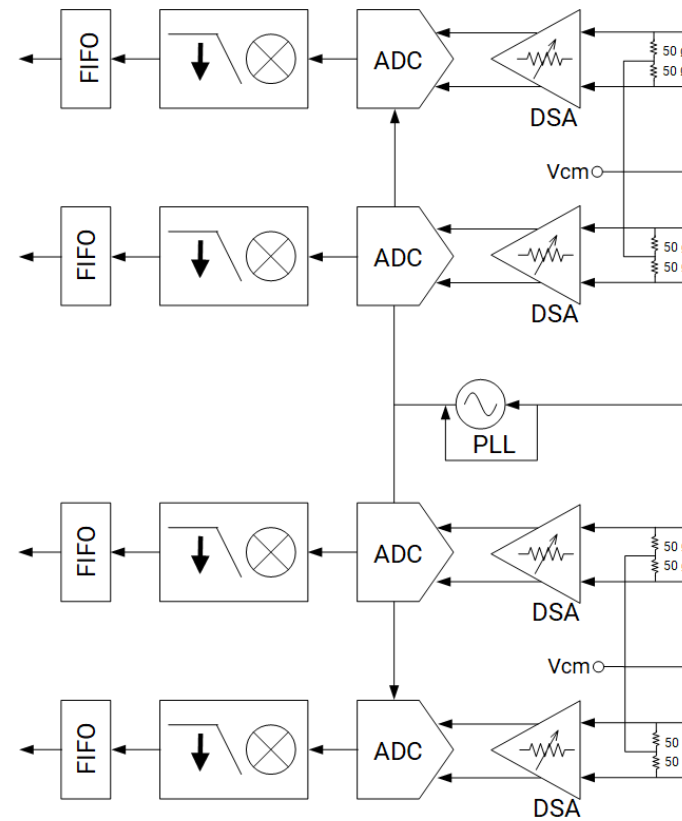
## Key Components of the Zynq UltraScale+ RFSoc



DS889\_01\_120721

# RFSoc Gen 3

- Their ADC's can be set to produce data in I/Q format



X23158-098019



# RFSoc Gen 3

## RF-ADC Features

- Tile configuration
  - Four or two RF-ADCs and one PLL per tile
  - Gen 1/Gen 2: 12-bit RF-ADC resolution, with 16-bit digital signal processing datapath; each 12-bit data stream is MSB-aligned to 16-bit samples at the output of the RF-ADC core before passing to the DDC block
  - Gen 3/DFE: 14-bit RF-ADC resolution, with 16-bit digital signal processing datapath; each 14-bit data stream is MSB-aligned to 16-bit samples at the output of the RF-ADC core before passing to the DDC block
  - Implemented as either four channels (Quad) or two channels (Dual) (the sampling rate is device dependent; for the actual sampling rate specifications, see the *Zynq UltraScale+ RFSoc Data Sheet: DC and AC Switching Characteristics (DS926)*)
- Decimation filters
  - Gen 1/Gen 2: 1x (bypass filter), 2x, 4x, 8x
  - Gen 3/DFE: 1x (bypass filter), 2x, 3x, 4x, 5x, 6x, 8x, 10x, 12x, 16x, 20x, 24x, 40x
  - 80% of Nyquist bandwidth, 89 dB stop-band attenuation
- Digital Complex Mixers
  - Full complex mixers support real or **I/Q** inputs from the RF-ADC
  - 48-bit Numeric Controlled Oscillator (NCO) per RF-ADC
  - Fixed  $F_s/4$ ,  $F_s/2$  low power frequency mixing mode, where  $F_s$  is the sample frequency
  - **I/Q** and real input signals supported
- Single/multi-band flexibility
  - 2x bands per RF-ADC pair
  - 4x bands per Quad RF-ADC tile
  - Can be configured for real or **I/Q** inputs
- Full bandwidth of the RF-ADC can be accessed in bypass mode
- Input signal amplitude threshold: Two programmable threshold flags per RF-ADC
- Built-in digital correction for external analog quadrature modulators:
  - Supports gain, phase, and offset correction for an **I/Q** input pair (two RF-ADCs)
- SYSREF input signal for multi-channel synchronization

# RFSoc Gen 3

- Their DAC's can be set to produce data in I/Q format

