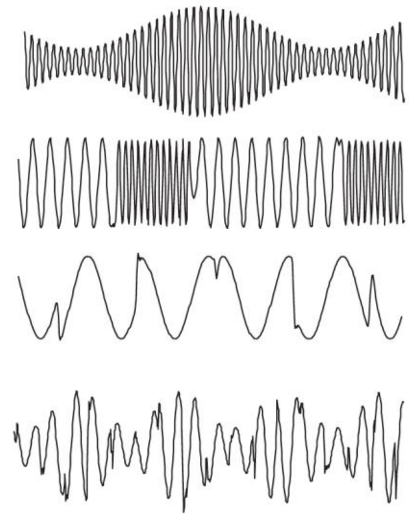
Physical

Lecture 3

- modulation
- coding, blocks / scramblers
- error correction / detection

Modulation (reprise)

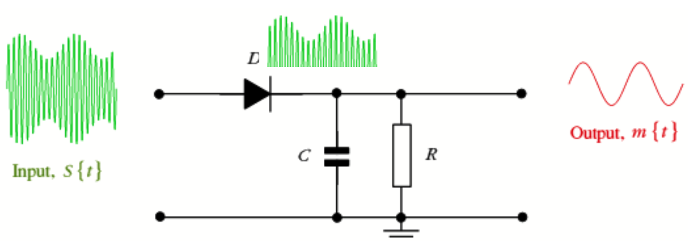


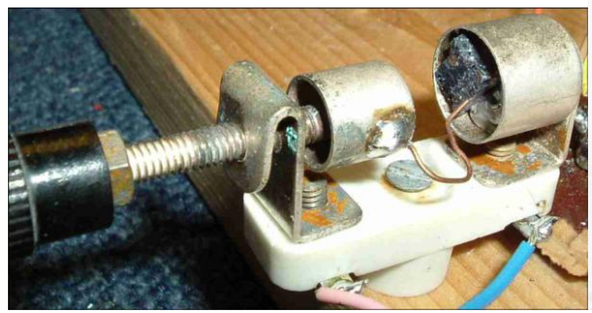
- Amplitude modulation
- Frequency modulation
- Phase shift modulation
- Amplitude and phase

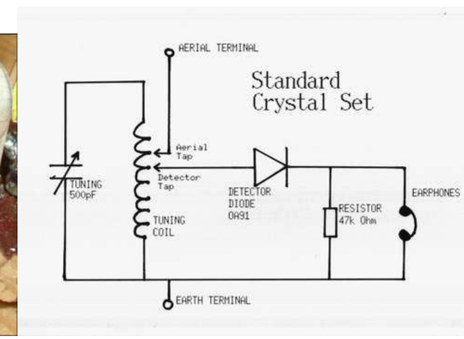
- Amplitude/Frequency/Phase shift *keying*
  - Use discrete set of levels/frequencies/phases
  - Just the thing for digital transmission!

Coherent detection

- Amplitude modulation can be detected/received by simply averaging the signal
  - As simple as a capacitor and a diode



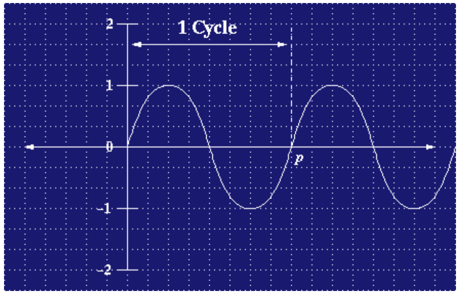




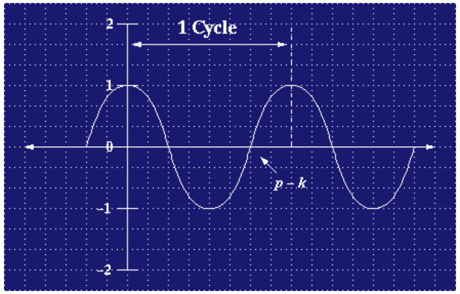
- Frequency and phase modulation
  - Are referred to as coherent receivers as they lock on to the frequency and phase…

Quadrature Amplitude Modulation

“Inphase”

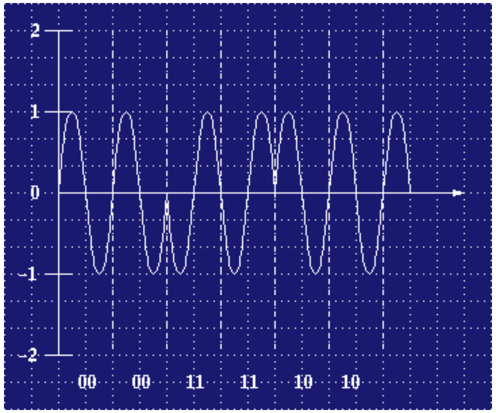


“Quadrature”



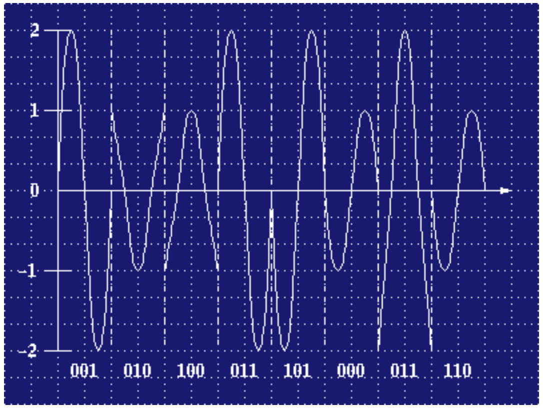
Phase shift keying

Value	Phase shift
00	None
01	¼ or 90º
10	½ or 180º
11	¾ or 270º

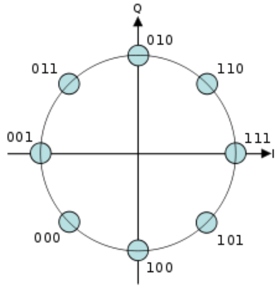


QAM

Value	Amp	Phase
000	1	0
001	2	0
010	1	¼
011	2	¼
100	1	½
101	2	½
110	1	¾
111	2	¾



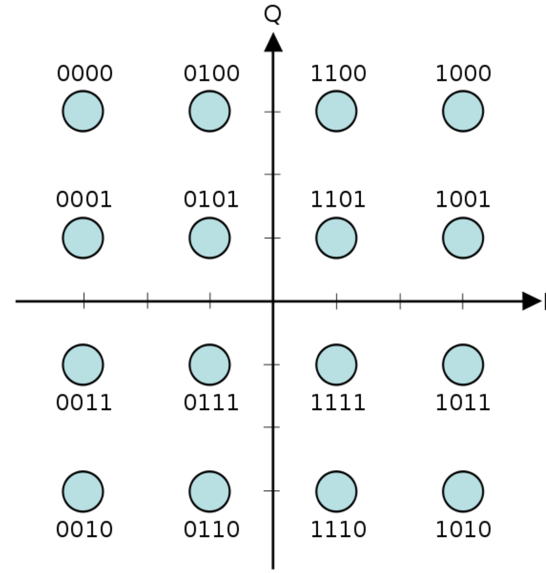
Constellations



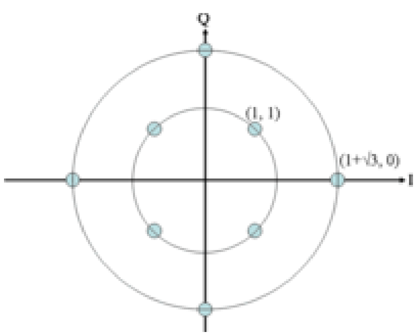
PSK8

- Way to visualize QAM schemes

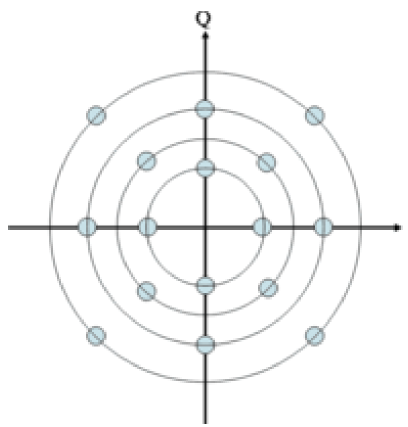
S(t) = I(t)cos(2πft) – Q(t)sin(2πft)



Rectangular QAM16
– gray code



Circular QAM8

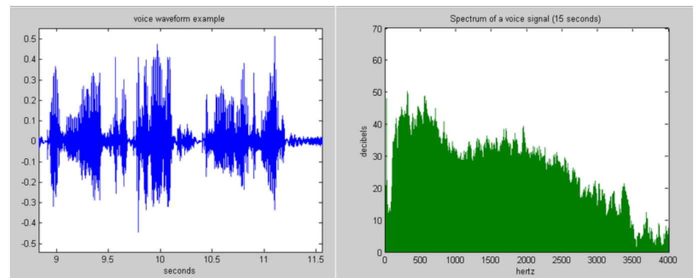


Circular QAM16

QAM64 – high speed modems
QAM256 – digital television

Telephones

- The telephone network was designed to carry voice
  - 0..5000 Hz



- The digital telephone network starts by digitizing the voice signal
  - Sample 12 bits (compressed to 8) at 8kHz
  - Nyquist theorem means we must “bandlimit” to < 4kHz - actually 3.4kHz is used

33.6k modem

3429Hz carrier and…

1664 QAM code points.

This picture is ¼ of the code points

45										408	396	394	400	414									
41							398	375	349	339	329	326	335	347	359	386							
37					412	371	340	314	290	279	269	265	273	281	302	322	353	390					
33				401	357	318	282	257	236	224	216	212	218	228	247	270	298	337	378				
29			406	350	306	266	234	206	185	173	164	162	170	181	197	220	253	288	327	379			
25			360	310	263	226	193	165	146	133	123	121	125	137	154	179	207	242	289	338	391		
21		384	324	277	229	189	156	131	110	96	87	83	92	100	117	140	172	208	254	299	354		
17		355	294	243	201	160	126	98	79	64	58	54	62	71	90	112	141	180	221	271	323	387	
13	392	330	274	222	177	135	102	77	55	41	35	31	37	48	65	91	118	155	198	248	303	361	
9	380	316	255	203	158	119	84	60	39	24	17	15	20	30	49	72	101	138	182	230	283	348	415
5	367	304	244	194	148	108	75	50	28	13	6	4	8	21	38	63	93	127	171	219	275	336	402
1	362	296	238	186	142	103	69	43	22	9	1	0	5	16	32	56	85	122	163	213	267	328	395
−3	365	300	240	190	144	106	73	45	25	11	3	2	7	18	36	59	88	124	166	217	272	331	397
−7	372	307	251	199	152	113	80	52	33	19	12	10	14	26	42	66	97	134	174	225	280	341	409
−11	388	320	261	210	167	128	94	67	47	34	27	23	29	40	57	81	111	147	187	237	291	351	
−15	410	343	284	232	183	149	115	89	68	53	46	44	51	61	78	99	132	168	209	258	315	376	
−19		369	311	259	214	175	139	116	95	82	74	70	76	86	104	129	157	195	235	285	342	399	
−23		403	345	292	249	205	176	150	130	114	107	105	109	120	136	161	191	227	268	319	373		
−27			382	332	287	250	215	184	169	153	145	143	151	159	178	202	231	264	308	358	413		
−31				377	333	293	260	233	211	200	192	188	196	204	223	245	278	312	352	404			
−35					383	346	313	286	262	252	241	239	246	256	276	295	325	363	407				
−39						405	370	344	321	309	301	297	305	317	334	356	385						
−43								411	389	374	366	364	368	381	393								
	−43	−39	−35	−31	−27	−23	−19	−15	−11	−7	−3	1	5	9	13	17	21	25	29	33	37	41	45

Recovering the bits…

- So far we have looked at schemes to send a series of bits
  - Phase receivers must learn the phase
  - Amplitude receivers must deal with attenuation
  - QAM must do both
- What if all the bits to be sent are zero, or one?
  - We send a constant signal
  - After some time receivers can’t tell if they are in phase or whether they are seeing ones of zeros.
- We need to ensure that the bits change sometimes even if the data to be sent is all ones or zeros….

Coding - scramblers

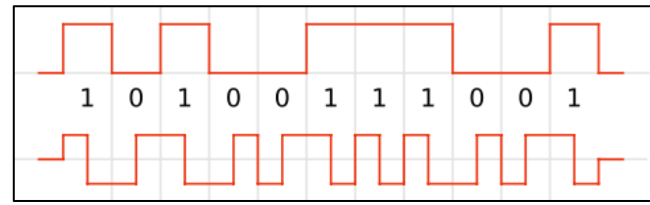
- Take a psuedo-random bit sequence – the *scrambler*
  – e.g. CRC polynomial (more later)
- XOR this *scrambler* with the data stream at the source and destination
  – But what if someone tried to send the pseudo-random bit sequence as the data!
  – In the phone network the trunks carry audio samples from many different phone calls
  – Very low probability that the data to be carried will conspire to conteract the scrambler….
- Scramblers
  – widely used in the traditional digital telephone network
  – traditionally not used much in computing…

Coding – block codes

Map bit or bits to be transmitted to more bits…

"1b2", Manchester encoding, 0 -> 01, 1 -> 10

- Used for original 3/10Mbps Ethernet



4b5, 16 symbols, 32 code words

- FDDI, 100Mbps Ethernet
- Some other code words used for control (e.g. token)

8b10b, 256 symbols, 1024 code words

- Gigabit Ethernet, PCIExpress (up to v3.0), Serial ATA, USB 3.0, DVB, DVI, HDMI, DAT, …

64b66b – Actually a scrambler with a 2 bit preamble…

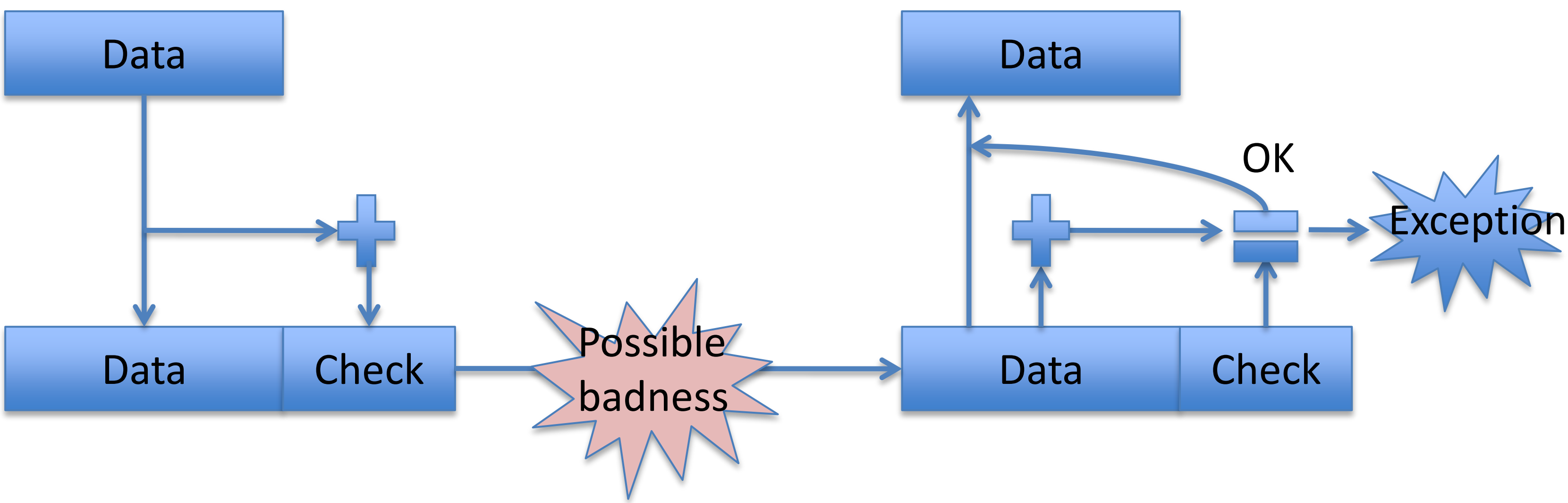
- 10GigE, actually just about everything 10G…

e.g. 4b5b

0x	data	code
0	0000	11110
1	0001	01001
2	0010	10100
3	0011	10101
4	0100	01010
5	0101	01011
6	0110	01110
7	0111	01111
8	1000	10010
9	1001	10011
A	1010	10110
B	1011	10111
C	1100	11010
D	1101	11011
E	1110	11100
F	1111	11101

Error detection

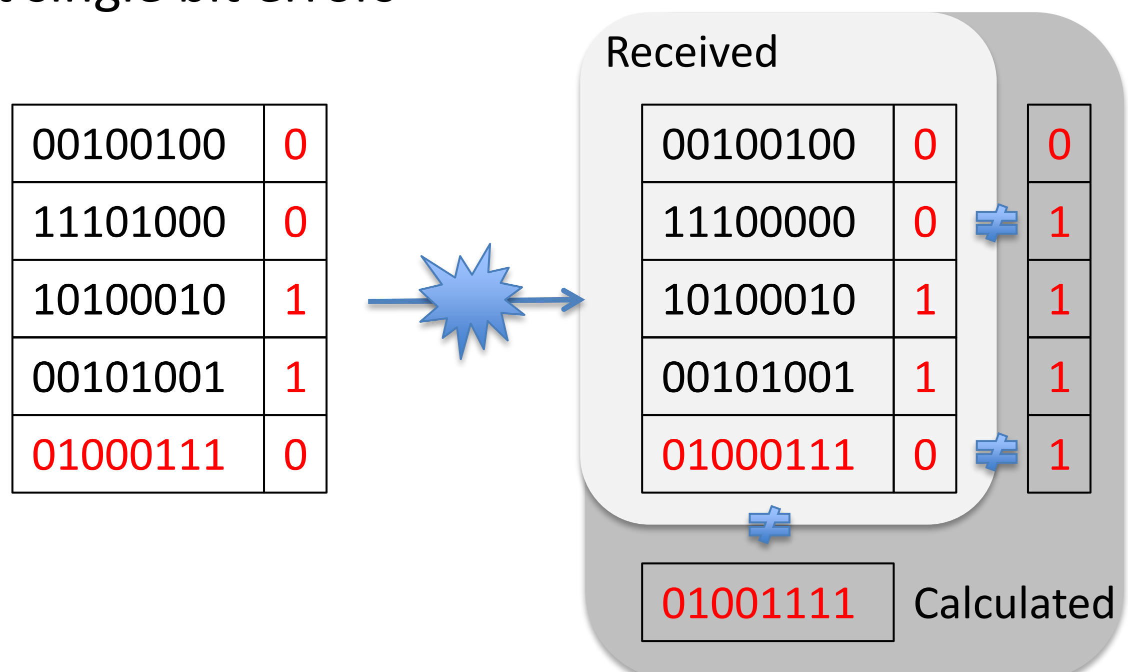
- Think of the 4b5b coding…what if I get 10001?
  - Was that a 1 bit error from 10101 or 10011?
  - Or 2 bits… or 3…
- In any case I know there was a an error



Parity

00100100	0

- Simple bit parity per byte/word spots odd number of bit flips (1 being common!)
  - Common anywhere in a Si chip that you have memory (registers, cache etc).
- 2D parity can correct single bit errors



More codes

2-out-of-5 codes

- detects single bit errors

0	01100
1	11000
2	10100
3	10010
4	01010
5	00110
6	10001
7	01001
8	00101
9	00011

Hamming Codes

- A general class of codes
  - correct single bit errors or detect single and 2 bit errors
  - have the rule $d+p+1 \leq 2^p$, d data bits, p parity bits

- However (7,4) is the canonical one…

  For data bits $d_1d_2d_3d_4$ transmit $p_1p_2d_1p_3d_2d_3d_4$ where:

  $p_1 = d_1+d_2+d_4$
  $p_2 = d_1+d_3+d_4$
  $p_3 = d_2+d_3+d_4$

- You need to change 3 bits to map any valid code to another one
- This is known as the *Hamming Distance*

- SECDED – a (72,64) Hamming code – much used by DRAM…

CRCs

- Cyclic Redundency Check in summary

  View data as a binary polynomial, call it D

  Choose a *generator* polynomial G, of length r+1

  Calculate the remainder $R = D*2^r/G$

  Transmit $D*2^r + R$

  At the receiver $(D*2^r + R)/G = 0$ or there is an error

CRC worked example

- G = 1001 (r = 3) can be viewed as:

$$1.x^3+0.x^2+0.x^1+1.x^0 = x^3+1$$

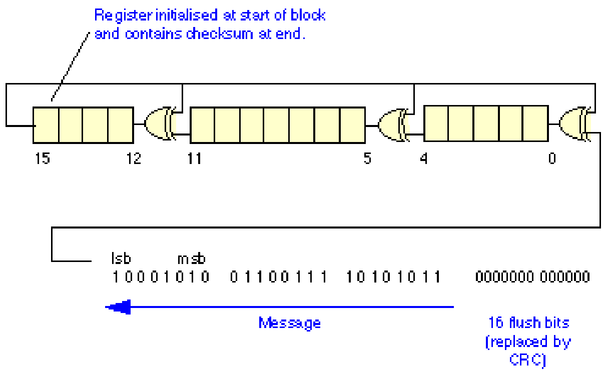
- D = 101110
- Hence R = 011
- Transmit 101110011

```
      101011
1001/101110000
    1001
     101
     000
     1010
     1001
      110
      000
      1100
      1001
       1010
       1001
        011
```

Why CRCs

- Good provable properties of error detection

- Real easy to implement:
  CRC-CCITT 16 - $G(x) = x^{16} + x^{12} + x^{5} + 1$
  Used in X.25, HDLC



- A bit grim in software
  – Sometimes have CRC instruction

- But very popular, e.g.
  CRC32 $G(x)=x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$
  All 802.3 (Ethernet, WiFi)

Another everyday code

- Reed Solomon (as originally described)
  - Message is a binary polynomial (again!) of length k
  - Evaluate the polynominal an n distinct points (n>k)
  - Send the n values
  - Can recover if I receive any k
  - Hamming distance is n-k+1

- CDs use two RS codes, (32,28) and (28, 24), and interleave the results
  - Can recover from 4000 bit error bursts
  - Or a scratch 2.5mm on the CD

- RS coding used for deep space transmissions to spacecraft

Summary

- Modulation – introducing QAM
  - widely used in radio and “copper”
  - uses coherent detection
  - constellation diagrams
- Underlying demodulator needs signal changes to work
  - Scramblers and block codes can be used
- Block codes get us into error detection and correction
  - e.g. Parity, Hamming Codes, CRC, Reed Solomon