Announcements

- Enrollment
  - Now 1 on the waitlist
- Reading
  - Chapter 3 (3.3-3.6)
- Project #2
  - Handout is on the web
  - Due on Monday Sept 24th (10 AM)
  - Reminder, no credit for late work

Error Codes (cont.)

- Error Recovery
  - Given m bits of data and r bits of error code
  - Want to correct any one bit error
  - There are n words one bit from each valid message
    - so need n+1 words for each valid message
    - thus $(n + 1) 2^m <= 2^n$
    - but n = m + r so $(m + r + 1) <= 2^r$
- Hamming Code
  - recovers from any one bit error
  - number bits from left (starting at 1)
    - power of two bits are parity
    - rest contain data
  - bit is checked by all parity bits in its sum of power expansion
    - bit 11 is used to compute parity bits 1, 2, and 8

Hamming Code Example

Char	ASCII	Hamming
H	1001 000	0011 0010 000
a	1100 001	1011 1001 001
m	1101 101	1110 1010 101
l	1101 001	0110 1011 001

- Burst Errors
  - can send hamming codes by column rather than row
  - if use k rows, then can detect any burst error up to k bits
    - uses kr bits to check a block km bits long

Computing a Hamming Code

Bit #s	1	2	3	4		5	6	7	8		9	10	11
Parity/Data	P	P	D	P		D	D	D	P		D	D	D
Data To Snd			1			0	0	1			0	0	0
Parity Bit 1	0		1			0		1			0		0
Parity Bit 2		0	1				0	1				0	0
Parity Bit 4				1		0	0	1					
Parity Bit 8									0		0	0	0
Message	0	0	1	1		0	0	1	0		0	0	0

Checking & Correcting a Hamming Code

Bit #s	1	2	3	4		5	6	7	8		9	10	11
Parity/Data	P	P	D	P		D	D	D	P		D	D	D
Data Sent	0	0	1	1		0	0	1	0		0	0	0
Data Recv	0	0	0	1		0	0	1	0		0	0	0
Parity Bit 1	1		0			0		1			0		0
Parity Bit 2		1	0				0	1				0	0
Parity Bit 4				1		0	0	1					
Parity Bit 8									0		0	0	0
XOR Paritys	1	1		0					0				
Corrected Msg	0	0	1	1		0	0	1	0		0	0	0

Binary # when XOR the parity is the bit position with the error (e.g. 0011 = bit 3 is wrong)

Error Detection

- Less bits are required
  - if errors are infrequent, then then this works better
  - assumes that re-transmission is possible
- Cyclic Redundancy Codes (CRC)
  - Use a generator function G(x) of degree r
    - r+1 bits long
    - $x^5 + x^2 + 1$ is degree 5 and represented as 100101
  - let M' be the message with r 0's on the end of it
  - divide M' into G(x) and compute remainder
    - use this as the r bit CRC code
  - a code with r bits will detect all burst errors less than r bits

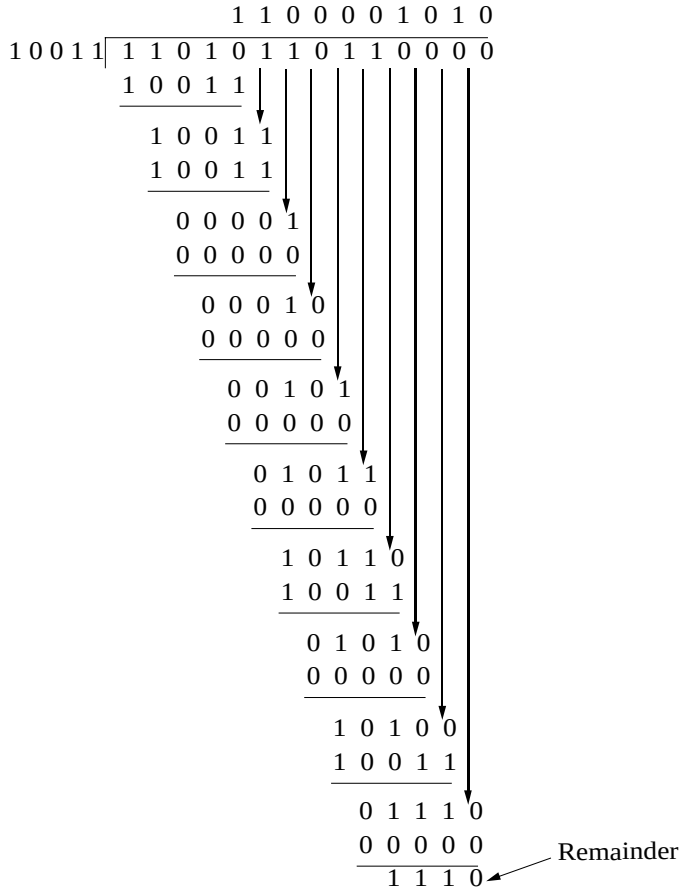
CRC's

- several G's are standardized
  - CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x + 1$
  - CRC-16 = $x^{16} + x^{15} + x^2 + 1$
  - CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$
- 16 bit CRC will catch
  - all single and double bit errors
  - all errors with an odd number of bits
  - all burst errors of length less than 16

CRC Example

Frame : 1 1 0 1 0 1 1 0 1 1
Generator: 1 0 0 1 1
Message after appending 4 zero bits: 1 1 0 1 0 1 1 0 0 0 0

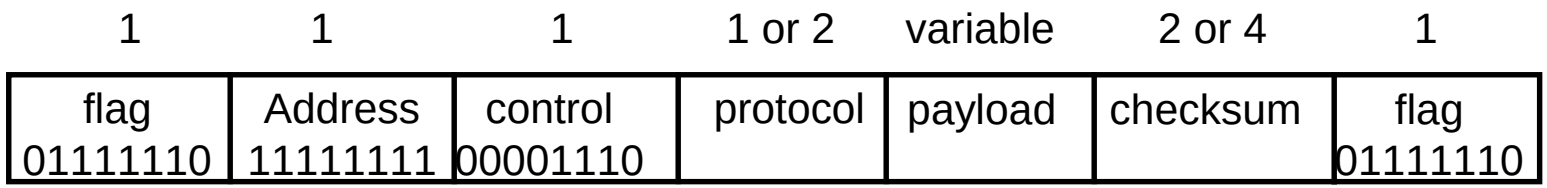
Division is done using XOR



Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 1 0

PPP Protocol

- Link Protocol for Serial Lines
  - Supports multiple network protocols: IP, IPX, CLNP, …
  - designed for dialup or leased lines
- Link Establishment (via LCP - Link Control Protocol)
  - Negotiate Options
    - configure-request: list of proposed options and values
    - configure-{ack/nack}: will (won't) use the requested option
  - Allows for authentication



From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

PPP Cont.

- NCP protocol
  - per network level protocol
  - used to establish network attributes (e.g. addresses)
  - high bit of protocol # is a one
- Notes on Link Format
  - character stuff flag byte in data
    - Escape Character is 0x7d (0111 1101)
    - Escape Character and Frame Marker sent at
      - <Esc-Char><data XOR 0x20>
  - option to skip address and control fields (since constant)
- IP
  - Protocol byte (0x21) or 0x8021 for IP NCP