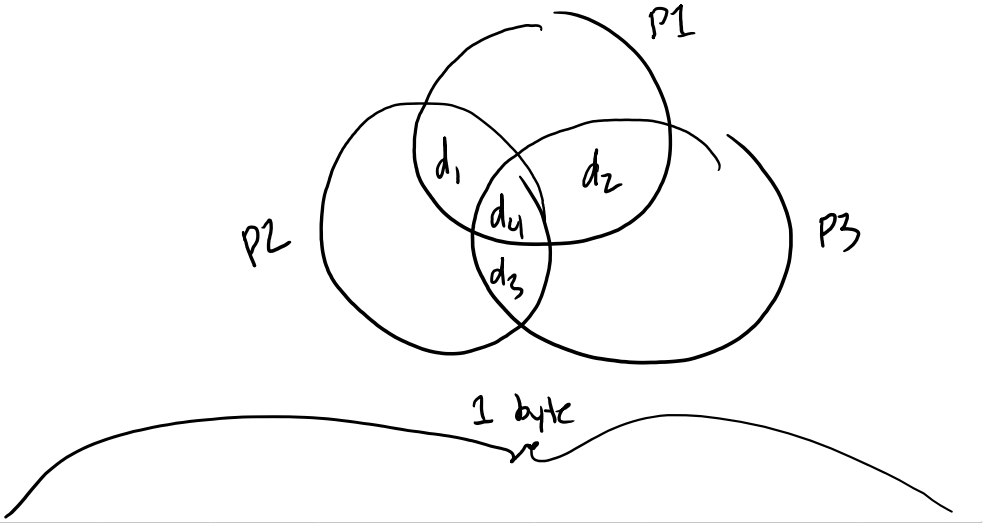
2019-04-22 Error Correction (PA 6)

Monday, April 22, 2019 3:04 PM

- When reading or receiving data either from RAM, HDD, network, etc., how do we know that what we read is what was written?
  - Or, did we unknowingly get into a game of telephone?
- Ensuring data integrity is of critical concern
  - "Artifacts" during movie watching
  - Banking transactions
  - Record keeping
- Goal: come up with a system that, for a reasonable price, can track and fix data errors
- Algorithm: Hamming Codes
  - Hamming codes takes every 4 bits and adds 3 data integrity bits
  - This scheme allows us to recover one bad bit per 4 bits and can detect failure if two or more bits are bad
  - Terms
    - Data bit - actual data that is of interest for storage / record keeping
    - Parity bits - bits that are used to track correctness of the data bits
  - Visualization of relationship between data and parity bits



Bit #	1	2	3	4	5	6	7	8
	P1	P2	D1	P3	D2	D3	D4	Not used
P1	Yes	No	Yes	No	Yes	No	Yes	-
P2	No	Yes	Yes	No	No	Yes	Yes	
P3	No	No	No	Yes	Yes	Yes	Yes	No

General Idea

- Take a 4-bit sequence and transcode into our matrix
- E.g. 1101

↗ concat

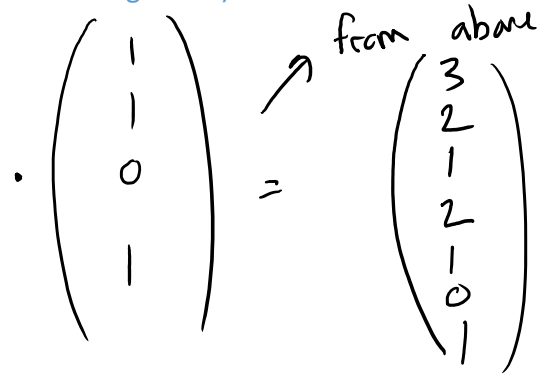
??	??	1	??	1	0	1	0

- Question: how do we generate the parity values?

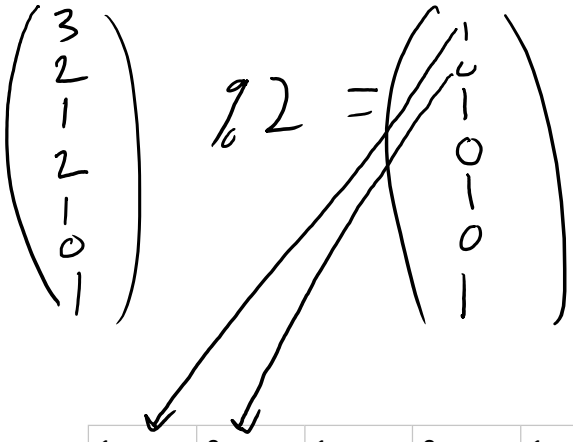
- Answer: we use another matrix (called Code Generation Matrix)

Code Generation Matrix (same for all Hamming codes)

1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	1



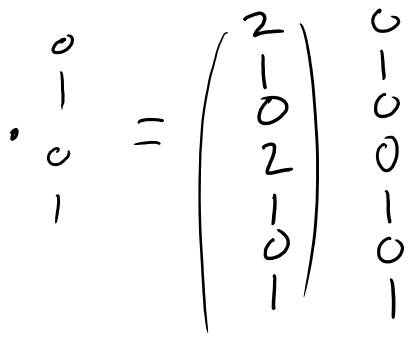
- To find the parity bits, we multiply the matrix by our data
- Next, mod the results by 2



1	0	1	0	1	0	1	0

- In the HW, you will be encoding text files. Consider, how would we encode the character 'T' (01010100). To encode the character, we must split into two 4-bit sequences
- Generation Matrix * 0101 = (x)01001012 (x) - dummy bit
- Generation Matrix * 0100 = (x)10011100

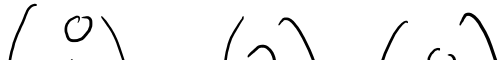
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	1



How does one decode?

- To decode, simply pull the data bits out of the hamming code.
- To check for data integrity, you multiply the parity check matrix by our encoded byte
-

Parity Check Matrix



by our encoded byte

Parity Check Matrix

1	0	1	0	1	0	1
0	1	1	0	0	1	1
0	0	0	1	1	1	1

$$\cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 2 \end{pmatrix} \% \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

What if there's an error?

all zeros means no errors!

Parity Check Matrix

1	0	1	0	1	0	1
0	1	1	0	0	1	1
0	0	0	1	1	1	1

$$\cdot \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \\ 2 \end{pmatrix} \quad g_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

Not zero, problem!

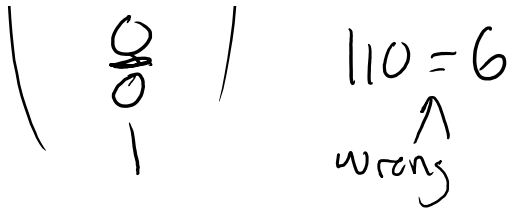
rotate $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$

011 → 3rd element in sequence is flipped

1	0	1	0	1	0	1
0	1	1	0	0	1	1
0	0	0	1	1	1	1

$$\cdot \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} \quad g_{1,2} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

110 = 6



- The hamming code scheme can either detect when one or more bits are flipped and throw an error, OR
- Attempt to correct error hoping that it flipped the correct bit.