Computer Organization/ Architecture (COMP2825)

Instructor: Maryam Tanha

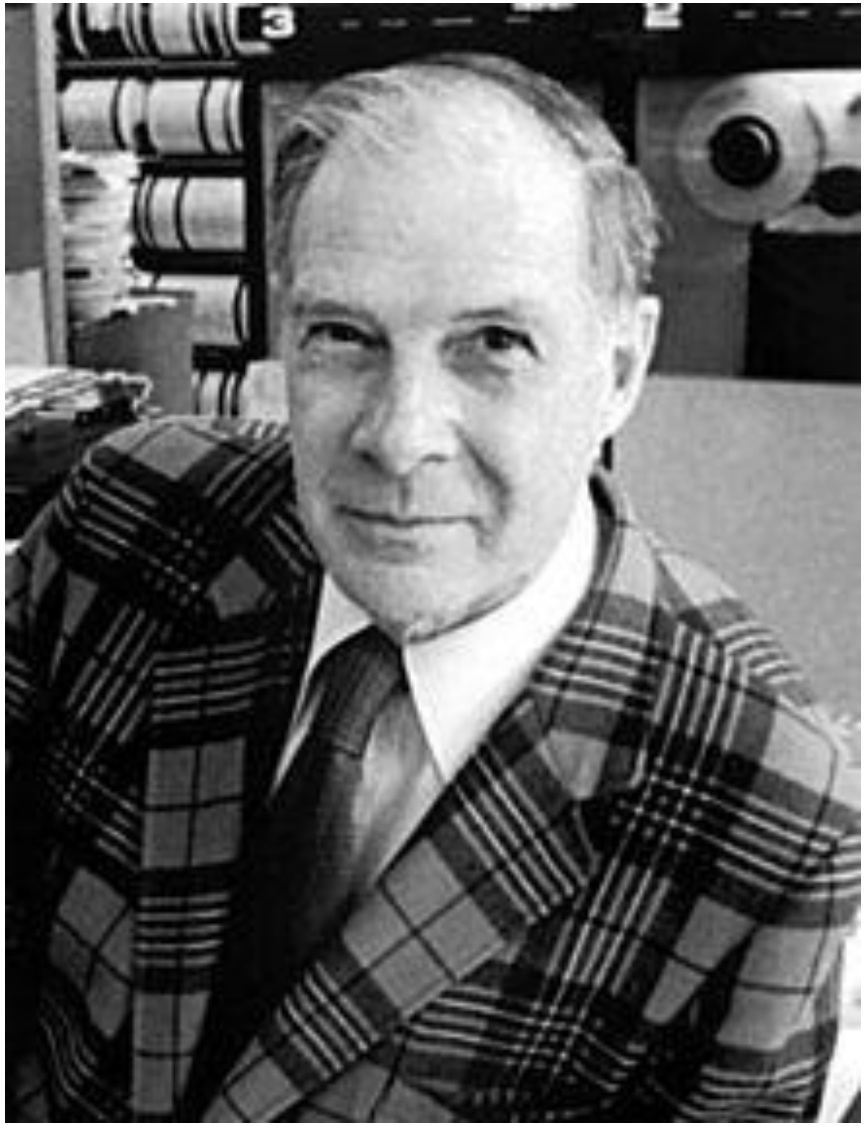
Fall 2021

Learning Outcomes of This Lecture

- **By the end of this lecture you will be able to**
  - Explain what is Hamming Code and how to use it.

The Hamming Code

Richard Hamming invented a popular redundancy scheme for memory, for which he received the **Turing Award** in 1968**.**



The Hamming Distance

- **Hamming distance**
  - number of bits that are different between two-bit patterns
  - e.g., the Hamming distance between 011011 and 001111 is two

The Hamming Distance

- **Hamming distance**
  - number of bits that are different between two-bit patterns
  - e.g., the Hamming distance between 011011 and 001111 is two

The Hamming Distance

- **Hamming distance**
  - number of bits that are different between two-bit patterns
  - e.g., the Hamming distance between 011011 and 001111 is two
- The **minimum Hamming distance**
  - a coding scheme *C* can detect ***s* errors,** if and only if, the minimum Hamming distance between any two of its valid codewords (correct bit patterns) is at least ***s*+1**.
    - e.g., minimum Hamming distance = 2 → single bit error detection
    - code that enables the detection of an error in data, but not the precise location.
  - a coding scheme *C* can correct **s** errors if the minimum Hamming distance among its valid codewords is ***2s + 1.***
    - e.g., for Hamming SEC (Single-Error Correcting), minimum Hamming distance = 3 → single-bit error correction

Calculating Hamming Distance

- XOR operation (⊕) on the two words and count the number of 1s in the result.
- **Example:** ***d***(10101, 11110) = 3 → 10101 ⊕ 11110 = 01011

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Hamming SEC Code

- Extra parity bits are used to identify the position of a single error.
- Single Error Correcting (SEC) is common in memory for servers today.
- **To calculate how many bits are needed for Hamming SEC code**, let *p* be total number of parity bits and *d* number of data bits in a *p* + *d* bit word. If *p* error correction bits are to point to error bit (*p* + *d* cases or bit positions) plus one case to indicate that no error exists, we need:

$$2^p >= p + d + 1 \text{ , and thus } p >= \log(p + d + 1)$$

- **Examples:**
  - for 8 bits data → $d = 8$ and $2^p >= p + 8 + 1$ → $p = 4$.
  - $p = 5$ for 16 bits of data, 6 for 32 bits, 7 for 64 bits, and so on.

Hamming SEC Code – Continued

- **Steps to calculate Hamming SEC (i.e., steps to create a codeword)**
  - indicate the length of the codeword by calculating the number of parity bits to be added to the dataword using the inequality formula.
  - number bits of the codeword from 1 starting from the leftmost bit.
  - all bit positions that are a power 2 are used for parity bits.
  - all other bit positions are used for data bits.
  - each parity bit checks certain data bits (each data bit is covered by 2 or more parity bits).
  - set parity bits to create even parity for each group.

Hamming SEC Code – Example 1

Show parity bits, data bits, and field coverage in a Hamming SEC code for **eight data bits**.

Example 1

Answer:

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits

Then, follow the steps.

Example 1

Answer:

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits

Then, follow the steps.

2^0 2^1 2^2 2^3

Bit position	1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits (codewords)	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Bit at position 3 is checked by p1 and p2 because 1+2 = 3

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Bit at position 3 is checked by p1 and p2 because 1+2 = 3

Bit at position 4 is checked by p4

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Bit at position 3 is checked by p1 and p2 because 1+2 = 3

Bit at position 4 is checked by p4

Bit at position 5 is checked by p1 and p4 because 1+4 =5

Example 1

Answer:

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Bit at position 3 is checked by p1 and p2 because 1+2 = 3

Bit at position 4 is checked by p4

Bit at position 5 is checked by p1 and p4 because 1+4 = 5

Bit at position 6 is checked by p2 and p4 because 2+4 = 6

Example 1

Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Answer:

Bit at position 1 is checked (covered) by p1

Bit at position 2 is checked by p2

Bit at position 3 is checked by p1 and p2 because 1+2 = 3

Bit at position 4 is checked by p4

Bit at position 5 is checked by p1 and p4 because 1+4 = 5

Bit at position 6 is checked by p2 and p4 because 2+4 = 6

…

Bit at position 12 is checked by p4 and p8 since 4+8 = 12

Example 2

Use Hamming code with even parity to create the codeword for the dataword 10001010.

Example 2 – Answer

Here, we have a dataword with 8 bits, so 4 parity bits required.

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits. We have shown the bit positions below.

bit position	1	2	3	4	5	6	7	8	9	10	11	12
codeword			1		0	0	0		1	0	1	0

Example 2 – Answer

Here, we have a dataword with 8 bits, so 4 parity bits required.

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits. We have shown the bit positions below.

			1 + 2		1 + 4		1+2+4		1+8		1+2+8	
bit position	1	2	3	4	5	6	7	8	9	10	11	12
codeword	1		1		0	0	0		1	0	1	0
	p1	p2	↑	p4	↑		↑	p8	↑		↑	

p1 covers data bit positions 3, 5, 7, 9, 11 and the number of 1s is odd so, to make it have even parity, p1= 1

Example 2 – Answer

Here, we have a dataword with 8 bits, so 4 parity bits required.

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits. We have shown the bit positions below.

		1+2			4 + 2	4+2+1			2+8	2+8+1	
bit position	1	2	3	4	5	6	7	8	9	10	11	12
codeword	1	0	1		0	0	0		1	0	1	0
	p1	p2	↑	p4		↑	↑	p8		↑	↑	

p2 covers data bit positions 3, 6, 7, 10, 11 and the number of 1s is even so, to make it have even parity, p2=0

Example 2 – Answer

Here, we have a dataword with 8 bits, so 4 parity bits required.

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits. We have shown the bit positions below.

					1+4	2+4	1+2+4					8+4
bit position	1	2	3	4	5	6	7	8	9	10	11	12
codeword	1	0	1	0	0	0	0		1	0	1	0
	p1	p2		p4	↑	↑	↑	p8				↑

p4 covers data bit positions 5, 6, 7, 12 and the number of 1s is zero (which is even) so, to make it have even parity p4=0

Example 2 – Answer

Here, we have a dataword with 8 bits, so 4 parity bits required.

d = 8 → $2^p >= p + 8 + 1$ → $2^p >= p + 9$ → p =4

Total number of bits for a codeword = 4 + 8 = 12 → a bit pattern with 12 bits. We have shown the bit positions below.

									1+8	8+2	1+2+8	4+8
bit position	1	2	3	4	5	6	7	8	9	10	11	12
codeword	1	0	1	0	0	0	0	0	1	0	1	0
	p1	p2		p4				p8	↑	↑	↑	↑

p8 covers data bit positions 9, 10, 11, 12 and the number of 1s is even. so, to make it have even parity p8 = 0

Example 3

Consider a Hamming codeword as follows: 1000001001

Extract the dataword (decode the dataword).

Example 3 – Answer

codeword : 1000001001 (bit positions that are powers of 2, i.e., parity bits, are colored in red) → dataword = 000101

How to Detect and Correct an Error using Hamming SEC? (Example: d=8)

Case 1: (p8,p4,p2,p1) = 0000 → no error

Case 2: Only one parity bit indicates an error. In this case, the parity bit itself is in error. For example, assume we count the number of 1s for the bit positions covered by p2 and we get odd parity while we get even parity for other bits covered by other parity bits. This means that the bit position 2 is in error.

Case 3: Multiple parity bits indicate an error. In this case, we compute the sum of the parity bit positions to indicate the position of error. For example, if parity bits p1, p2, and p8 indicate an error, then bit position 1+2+8 = 11 is in error.

Example 4

Consider a Hamming code that protects 8-bit words with 4 parity bits. Assume even parity has been used. If we read the value 001101110101 from memory, is there an error? If so, correct the error.

Example 4 – Answer

Our codeword has 12 bit: 001101110101

We indicate which bits are covered by each parity bit (using red color) and create the following (similar to Example 1).

Example 4 – Answer

0011 0111 0101

p1 → covers bit positions 1, 3, 5, 7, 9, 11 → 010100 → two 1s → even parity → no error

Example 4 – Answer

0011 0111 0101

p2 → covers bit positions 2, 3, 6, 7, 10, 11 → 011110→ four 1s → even parity → no error

Example 4 – Answer

0011 0111 0101

p4 → covers bit positions 4, 5, 6, 7,12→ 10111 → four 1s → even parity → no error

Example 4 – Answer

0011 0111 0101

p8 → covers bit positions 8, 9, 10, 11, 12 → 10101 → three 1s → odd parity → there is an error

Example 4 – Answer

0011 0111 0101

only one parity bit indicates an error and it is p8. Therefore, bit position 8 has an error. Its value is 1. To correct it, we change the value to 0. Thus, the correct codeword is **001101100101**.

Reference

[1] David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 6th Ed, 2020, Morgan Kaufmann Publishers, Inc.