University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

# ECE 120: Introduction to Computing

## Parity and Hamming Distance

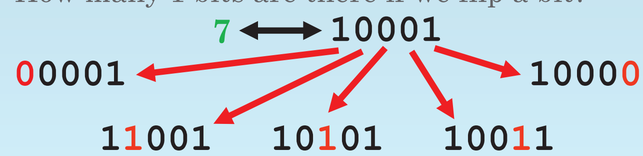
  

## A 2-out-of-5 Code Can Detect Any Single Bit Flip

A 2-out-of-5 code maps decimal digits into 5-bit code words.

Each code word has exactly two 1 bits.

How many 1 bits are there if we flip a bit?




Result: **Either ONE 1 bit or THREE 1 bits.**

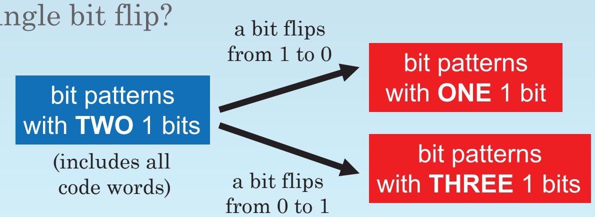
  

## Can We Generalize This Approach to Error Detection?

A 2-out-of-5 code represents decimal digits.

What about binary numbers? Letters? Colors?

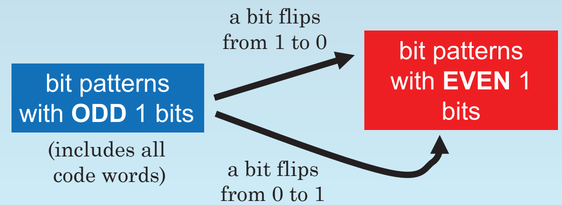
Is there a general strategy for handling a single bit flip?






## Generalize by Using Even and Odd Numbers of 1 Bits

What if we choose all code words with an ODD number of 1 bits?




Result: **Any bit flip gives a non-code-word!**

## Add a Parity Bit to Any Representation!

starting with any representation

- Add one extra **parity bit** to each code word.
- Choose parity bit's value to make total number of 1 bits ODD (called odd parity).

For example, 3-bit unsigned with odd parity…

| | | | |
|---|---|---|---|
| 0 ⟷ | 0001 | 4 ⟷ | 1000 |
| 1 ⟷ | 0010 | 5 ⟷ | 1011 |
| 2 ⟷ | 0100 | 6 ⟷ | 1101 |
| 3 ⟷ | 0111 | 7 ⟷ | 1110 |



## Hamming Distance: The Number of Bits that Differ

Let's define a way to measure distance

- between two bit patterns
- as the number of bits that must change/flip

0101 ↕ 0100 DISTANCE 1

0101 ↕↕↕ 0010 DISTANCE 3

We call this measure **Hamming distance** (after Richard Hamming, a UIUC alumnus).



## Define the Hamming Distance for a Representation

Let's also define the Hamming distance for a representation (let's call a representation a **code** now):

- Given the set of code words (bit patterns) that have meaning,
- the Hamming distance of the code
  - is the **minimum Hamming distance**
  - between **any two distinct code words**.



## Example: The Hamming Distance of BCD is 1

What is the Hamming distance (H.D.) of BCD?

- Choose two code words,
- say those representing digits **0** and **1**.

0 ⟷ 0000

↕ DISTANCE 1

1 ⟷ 0001

H.D. of BCD is min. over all code word pairs.

**Thus BCD has Hamming distance 1.**

## Example: The Hamming Distance of 2-out-of-5 is 2

What is the H.D. of a 2-out-of-5 code?

- Choose two distinct code words, **A** and **B**.
- Each has two 1s (cannot be the same two).
- So **A** must have at least one 1 in a position where **B** has a 0.
- And **B** must have at least one 1 in a position where **A** has a 0.

H.D. from **A** to **B** is thus at least 2.

**Thus a 2-out-of-5 code has H.D. 2.**



## Example: H.D. with Odd Parity

What is the H.D. of a code with odd parity?

- Choose two distinct code words, **A** and **B**.

```
A ???0?????
     ↕ (SAME)
B ???1?????
```

**A** must differ from **B** in some location.

- Assume that the location is unique.
- A has odd parity, so B has even parity (contradiction, so location cannot be unique).



## Example: H.D. with Odd Parity is at Least 2

There must be **at least two** locations in which **A** to **B** differ.

```
A ???0??0??
     ↕  ↕
B ???1??1??
```

**Thus H.D. with odd parity is at least 2.**



## H.D. of 2's Complement with Odd Parity is 2

Add an odd parity bit to 3-bit 2's complement:

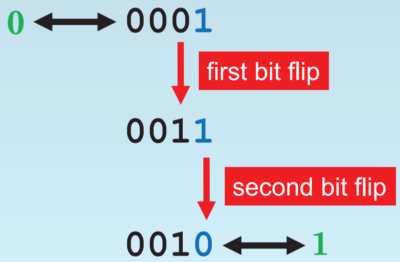
| | | | |
|---|---|---|---|
| -4 | ⟷ 1000 | 0 | ⟷ 0001 |
| -3 | ⟷ 1011 | 1 | ⟷ 0010 |
| -2 | ⟷ 1101 | 2 | ⟷ 0100 |
| -1 | ⟷ 1110 | 3 | ⟷ 0111 |

```
0 ⟷ 0001
      ↕↕  DISTANCE 2
1 ⟷ 0010
```

**Thus H.D. for this code is exactly 2.**

## With H.D. 2, Two-Bit Errors Can be Undetectable

What happens if two bit errors occur
when using 3-bit 2's complement with parity?




**In this case, no error can be detected!**



## A Code with H.D. of d Allows Detection of (d-1) Errors

More generally…
- Start with a code with H.D. given by **d**
- Ask: How many errors can be detected?

H.D. of **d** implies
- Any code word is at least **d** flips from any other.
  - Thus (**d**-1) bit errors cannot transform any code word into any other.
  - Thus up to (**d**-1) bit errors can be detected.
- There exist code words **A** and **B** separated by **d** flips, so **d** bit errors can transform **A** into **B**.