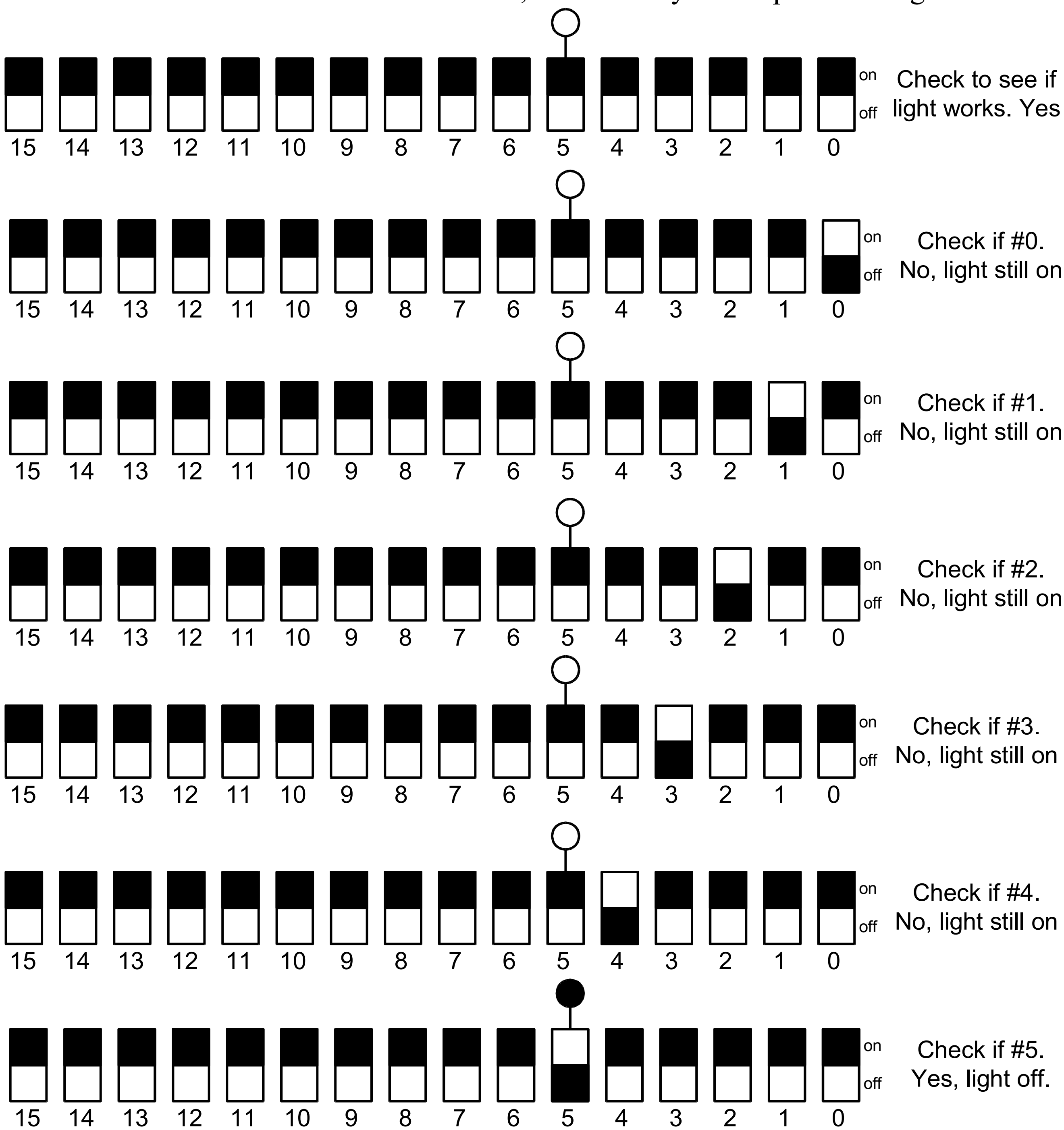
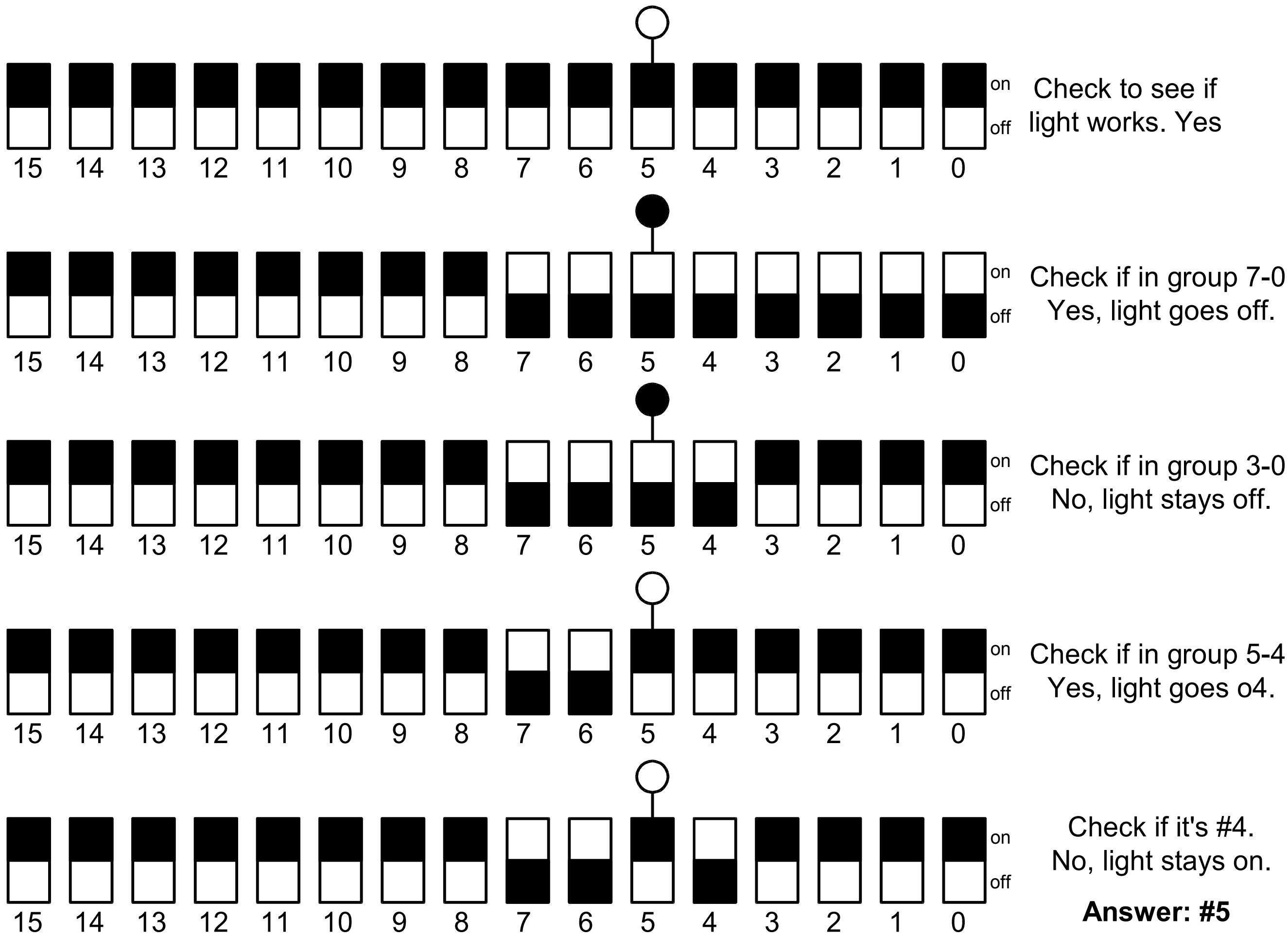
Lecture 3: Error Correction

Binary Search

Suppose you're home alone and want to find which breaker circuit a particular light is on. (Let's say it' on breaker #5.) You turn on the light, go down in the basement and turn breaker #0 off, and go upstairs to see if the light is off. If so, you found the breaker circuit it's on. If not, you repeat the process with breaker #1, etc. After six tries you find it's on breaker circuit #5. With 16 breakers, it will take you 8 trips on average.



A quicker procedure is to use what's called a binary search. You test half of the breakers in one trip to find which group of 8 breakers the light is on. Then you test half of that group to see which group of 4 breakers the light is on, etc. “Test” means reverse the position of the breakers. This method always takes exactly four trips up and down the stairs.



Error Correction

A single parity bit can only tell you there's a error somewhere in a block of bits, but it can't tell you which bit. We could add a second parity bit to test which half of the bits the error is in, a third parity bit to test which half of that group the error is in, etc. For a block of 16 bits we'll need 5 parity bits altogether, as in the binary search problem above. But the 5 parity bits are necessarily part of the block of 16 bits, so there are only 11 information bits in the block. The scheme is called a Hamming Code after its inventor. We'll look at a simple example with a block of 8 bits.

Hamming code

$n = 3$	determines the size of an error-correction block
$2^n = 8$	number of bits in a block
$n + 1 = 4$	number of parity (p) bits in a block
$2^n - (n + 1) = 4$	number of information (i) bits in a block

p_4	p_3	p_2	i_3	p_1	i_2	i_1	i_0
x	x	x	x	x	x	x	x

x	x	x	x	<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	p1 field
x	x	<u>x</u>	<u>x</u>	x	x	<u>x</u>	<u>x</u>	p2 field
x	<u>x</u>	x	<u>x</u>	x	<u>x</u>	x	<u>x</u>	p3 field
<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	<u>x</u>	p4 field

Number N represented by the i bits:

$\underline{N} = 2^0 \cdot i_0 + 2^1 \cdot i_1 + 2^2 \cdot i_2 + 2^3 \cdot i_3 = 1 \cdot i_0 + 2 \cdot i_1 + 4 \cdot i_2 + 8 \cdot i_3$

for odd parity in each field:

0	1	1	0	1	0	0	0	$N = 0$
0	0	0	0	0	0	0	1	$N = 1$
0	1	0	0	0	0	1	0	$N = 2$
1	0	0	0	1	0	1	1	$N = 3$
0	0	1	0	0	1	0	0	$N = 4$
1	1	0	0	1	1	0	1	$N = 5$
0	0	0	0	1	1	1	0	$N = 6$
0	1	1	0	0	1	1	1	$N = 7$
1	0	0	1	1	0	0	0	$N = 8$
1	1	1	1	0	0	0	1	$N = 9$
0	0	1	1	0	0	1	0	$N = 10$
0	1	0	1	1	0	1	1	$N = 11$
0	1	0	1	0	1	0	0	$N = 12$
0	0	1	1	1	1	0	1	$N = 13$
1	1	1	1	1	1	1	0	$N = 14$
1	0	0	1	0	1	1	1	$N = 15$

Problem 1:
The received pattern is 0 1 1 0 0 1 0 1. Determine if there is an error, and correct the error if there is. (Errors are so rare that it's assumed there's no more than one error.)

p_4	p_3	p_2	i_3	p_1	i_2	i_1	i_0	
0	1	1	0	0	1	0	1	p1 field - EVEN parity
0	1	1	0	0	1	0	1	p2 field - EVEN parity
0	1	1	0	0	1	0	1	p3 field - odd parity
0	1	1	0	0	1	0	1	p4 field - EVEN parity

EVEN parity in the p4 field says there is an error. The error is the p1, p2, and p4 fields, but not the p3 field. Which bit is wrong?

Answer 1:
The only bit in the p1, p2, and p4 fields but not the p3 field is the **i_1 bit**. So the received bits should have been not

p_4	p_3	p_2	i_3	p_1	i_2	i_1	i_0
0	1	1	0	0	1	0	1

but

0	1	1	0	0	1	1	1

which corresponds to $N = 7$.

Problem 2:
The received pattern is 0 1 0 0 0 1 1 1. Determine if there is an error, and correct the error if there is.

p_4	p_3	p_2	i_3	p_1	i_2	i_1	i_0	
0	1	0	0	0	1	1	1	p1 field - odd parity
0	1	0	0	0	1	1	1	p2 field - EVEN parity
0	1	0	0	0	1	1	1	p3 field - odd parity
0	1	0	0	0	1	1	1	p4 field - EVEN parity

Even parity in the p4 field says there is an error. The error is in the p2 and p4 fields, but not the p1 or p3 field. Which bit is wrong?

Answer 2:
The only bit in the p2 and p4 fields, but not the p1 or p3 field is the **p_2 bit**. So the received bits should have been not

p_4	p_3	p_2	i_3	p_1	i_2	i_1	i_0
0	1	0	0	0	1	1	1

but

0	1	1	0	0	1	1	1

So all the information bits were, in fact, correct.

Efficiency

2^n number of bits in a block
$n + 1$ number of parity bits in a block
$2^n - (n + 1)$ number of information bits in a block

efficiency = [information bits in a block] / [total bits in a block]
= $[2^n - (n + 1)] / 2^n$

n	block bits	parity bits	information bits	efficiency
	2^n	$n + 1$	$2^n - (n + 1)$	$[2^n - (n + 1)] / 2^n$
2	4	3	1	25.000%
3	8	4	4	50.000%
4	16	5	11	68.750%
7	128	8	120	93.750%
10	1024	11	1013	98.926%