

# Numbering Systems and Digital Electronics

**Objective:** To understand and appreciate the binary number system

1. Quantities can be represented in any base

Example: The number 2157 is actually:

$$2 \times 10^3 + 1 \times 10^2 + 5 \times 10^1 + 7 \times 10^0$$

Or

$$2000 + 100 + 50 + 7 = 2157$$

Known as Base-10 system (decimal). Perhaps evolved because people have 10 fingers

2. Computers use a Base-2 system (binary)

3. Bits represent on-off switches, 1 and 0 respectively

Example: The (binary) number 10011 (to a computer) is actually:

$$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

Or

$$16 + 0 + 0 + 2 + 1 = 19$$

This process is called a binary-to-decimal conversion

#### 4. Decimal-to-binary conversion

Example: Convert  $234_{10}$  to binary

Step 1: Form table

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1

(1)

Step 2: Populate table

$234_{10} =$

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
1	1	1	0	1	0	1	0

Sanity Check:  $128 + 64 + 32 + 0 + 8 + 0 + 2 + 0 = 234$

Answer:  $234_{10} = 11101010_2$

5. n-bit machine Largest decimal number represented by n-bits:  $2^n - 1$

Example:

$255_{10} =$

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1

$n = 8 \text{ bits}, 2^8 - 1 = 255$

$256_{10} =$

$2^8$	$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
256	128	64	32	16	8	4	2	1
1	0	0	0	0	0	0	0	0

$256_{10}$  appears to be  $0_{10}$  on 8-bit machines.  
This is called overflow

6. C Programming: `int` is a 16-bit number (2 bytes)

$$n = 16 \text{ bits, } 2^{16}-1 = 65535$$

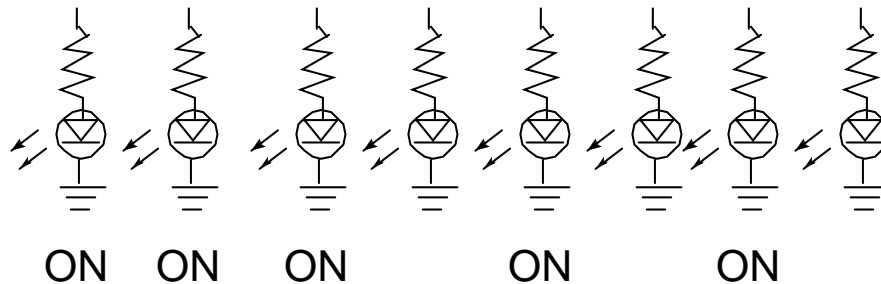
7. Digital Electronics: +5 Volts =  $1_2$  and GND =  $0_2$

Example:

$234_{10} =$

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
1	1	1	0	1	0	1	0

Digital Output Port  
(e.g. parallel or I2C)



8. Turning on LEDs is a big deal