Topic 02 Digital Representation

RECAP SUMMARY

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	*[NULL]*	32	20	*[SPACE]*	64	40	**@**	96	60	**`**
1	1	*[START OF HEADING]*	33	21	**!**	65	41	**A**	97	61	**a**
2	2	*[START OF TEXT]*	34	22	**"**	66	42	**B**	98	62	**b**
3	3	*[END OF TEXT]*	35	23	**#**	67	43	**C**	99	63	**c**
4	4	*[END OF TRANSMISSION]*	36	24	**$**	68	44	**D**	100	64	**d**
5	5	*[ENQUIRY]*	37	25	**%**	69	45	**E**	101	65	**e**
6	6	*[ACKNOWLEDGE]*	38	26	**&**	70	46	**F**	102	66	**f**
7	7	*[BELL]*	39	27	**'**	71	47	**G**	103	67	**g**
8	8	*[BACKSPACE]*	40	28	**(**	72	48	**H**	104	68	**h**
9	9	*[HORIZONTAL TAB]*	41	29	**)**	73	49	**I**	105	69	**i**
10	A	*[LINE FEED]*	42	2A	*****	74	4A	**J**	106	6A	**j**
11	B	*[VERTICAL TAB]*	43	2B	**+**	75	4B	**K**	107	6B	**k**
12	C	*[FORM FEED]*	44	2C	**,**	76	4C	**L**	108	6C	**l**
13	D	*[CARRIAGE RETURN]*	45	2D	**-**	77	4D	**M**	109	6D	**m**
14	E	*[SHIFT OUT]*	46	2E	**.**	78	4E	**N**	110	6E	**n**
15	F	*[SHIFT IN]*	47	2F	**/**	79	4F	**O**	111	6F	**o**
16	10	*[DATA LINK ESCAPE]*	48	30	**0**	80	50	**P**	112	70	**p**
17	11	*[DEVICE CONTROL 1]*	49	31	**1**	81	51	**Q**	113	71	**q**
18	12	*[DEVICE CONTROL 2]*	50	32	**2**	82	52	**R**	114	72	**r**
19	13	*[DEVICE CONTROL 3]*	51	33	**3**	83	53	**S**	115	73	**s**
20	14	*[DEVICE CONTROL 4]*	52	34	**4**	84	54	**T**	116	74	**t**
21	15	*[NEGATIVE ACKNOWLEDGE]*	53	35	**5**	85	55	**U**	117	75	**u**
22	16	*[SYNCHRONOUS IDLE]*	54	36	**6**	86	56	**V**	118	76	**v**
23	17	*[ENG OF TRANS. BLOCK]*	55	37	**7**	87	57	**W**	119	77	**w**
24	18	*[CANCEL]*	56	38	**8**	88	58	**X**	120	78	**x**
25	19	*[END OF MEDIUM]*	57	39	**9**	89	59	**Y**	121	79	**y**
26	1A	*[SUBSTITUTE]*	58	3A	**:**	90	5A	**Z**	122	7A	**z**
27	1B	*[ESCAPE]*	59	3B	**;**	91	5B	**[**	123	7B	**{**
28	1C	*[FILE SEPARATOR]*	60	3C	**<**	92	5C	****	124	7C	**|**
29	1D	*[GROUP SEPARATOR]*	61	3D	**=**	93	5D	**]**	125	7D	**}**
30	1E	*[RECORD SEPARATOR]*	62	3E	**>**	94	5E	**^**	126	7E	**~**
31	1F	*[UNIT SEPARATOR]*	63	3F	**?**	95	5F	**_**	127	7F	*[DEL]*

ASCII Table

Stores only 128 Characters

- Most are words in English

Unicode to replace ASCII code

- So that European, Cyrillic, Chinese Mandarin can be stored
- Allows computers to used by people of different languages

Decimal (Base 10)

Radix	10	10	10	10
Position In	3	2	1	0
Calculate	10^3	10^2	10^1	10^0
Positional Value	1000	100	10	1

People use the **denary** (or decimal) number system in their day-to-day lives. This system has 10 digits that we can use: 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9.

1234

	Thousands	Hundreds	Tens	Ones
Positional Value	1000	100	10	1
Decimal Number	1	2	3	4
Calculate	1x1000	2x100	3x10	4x1
Add them	1000	+200	+30	+4
Result	1234			

4321

	Thousands	Hundreds	Tens	Ones
Positional Value	1000	100	10	1
Decimal Number	4	3	2	1
Calculate	4x1000	3x100	2x10	1x1
Add them	4000	+300	+20	+1
Result	4321			

Binary (Base 2)

Simply a string of 0’s and 1’s

- For example, 4-bit binary code could look like 0001
- Each digit in the code links to a set value determined
- Starting from right and moving to the left, taking the number 1 and then doubling as you move along to the next slot (1, 2, 4, 8, 16, and so on)

Decimal Digit Value	256	128	64	32	16	8	4	2	1
Binary Digit Value	1	0	1	1	0	0	1	0	1

By adding together ALL the decimal number values from right to left at the positions that are represented by a “1” gives
us: (256) + (64) + (32) + (4) + (1) = 357_{10} or three hundred and fifty seven as a decimal number.

Convert decimal numbers to binary

Converting the number 19 to binary

- The procedure is to keep dividing the number by 2, and keeping track of the remainder until we get 0 as a result of the division. Then read the remainder values in reverse order

Division	Quotient	Remainder	Value	
19 / 2	9	1	2^0	LSB
9 / 2	4	1	2^1	
4 /2	2	0	2^2	
2 / 2	1	0	2^3	
1 / 2	0	1	2^4	MSB

Answer: 19 is 10011 binary

Convert decimal numbers to binary

Converting the number 19 to binary

- The procedure is to keep dividing the number by 2, and keeping track of the remainder until we get 0 as a result of the division. Then read the remainder values in reverse order

Division	Quotient	Remainder	Value	
19 / 2	9	1	2^0	LSB
9 / 2	4	1	2^1	
4 /2	2	0	2^2	
2 / 2	1	0	2^3	
1 / 2	0	1	2^4	MSB

Answer: 19 is 10011 binary

Number Representation - Unsigned

Unsigned numbers used for Memory address, cluster number (file system) and process identifier (PID).

If there are N bits in the binary number, the range of the number is: 2^N-1. Assume 4 bits: $2^4 – 1 = 16 – 1 = 15$

- Range for 8 bits: 28 – 1 = 256 – 1 = 255

Number Representation (Hexadecimal to Decimal)

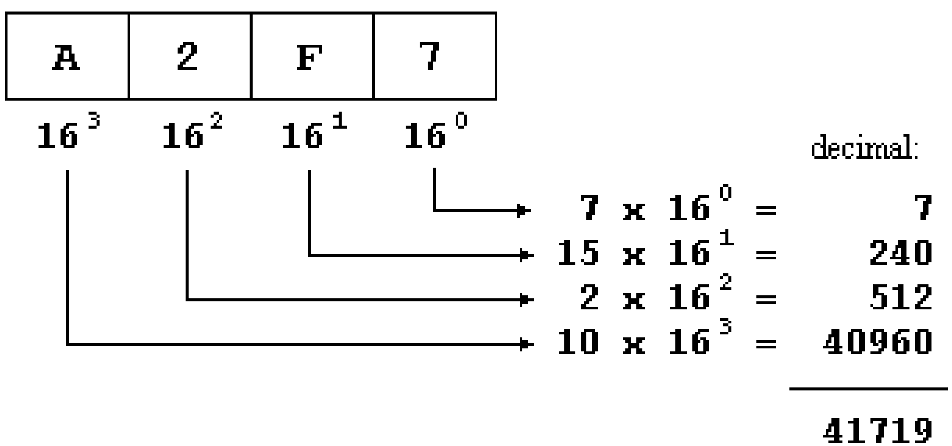
Hexadecimal (Base 16):

$H = h_3\ h_2\ h_1\ h_0$ (h = hexadecimal ranging from 0 to 9, A to F)

If $h_i = 1$, then 16^i is added into the sum that determines the value of H.

For values > 9, we use A,B,C,D,E,F to represent digits up to 15.

Let's assume that H = A 2 F 7



Binary	Hex	Decimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Binary / Hexadecimal / Decimal

Notation for binary, hexadecimal and decimal (summary)

Binary (base 2)

- 10101111_2
- 10101111

Hexadecimal (base 16)

- AF_{16}
- 0xAF

Decimal (base 10)

- 175

Number Representation - Signed

For signed integers, the leftmost bit is used to indicate the sign:

0 for positive	1 for negative

There are three ways to represent signed integers:

Sign and magnitude	1's complement	2's complement

Number Representation - Signed

B	Values represented		
$b_3b_2b_1b_0$	Sign and magnitude	1's complement	2's complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

Notation for binary, hexadecimal and decimal (summary)

Positive values have identical representations in all systems.

Negative numbers have different representations

- **Sign-and-magnitude**: negative values are represented by changing the most significant bit (b3).
- **1’s-complement**: negative values are obtained by complementing each bit of the corresponding positive number.
- **2’s-complement**: obtain by forming bit complement of that number, then add 1.

2’s-complement integers

2’s-complement representation is used in current computers

Consider a four-bit signed integer example, where the value +5 is represented as:

0 1 0 1

To form the value **-5**, complement all bits of

0 1 0 1 to obtain

1 0 1 0 (1’s-complement)

and then add 1 to obtain

1 0 1 1 (2’s-complement)

2's-Complement

2's-complement system is the most efficient way to carry out addition and subtraction operations.

It is the most often used in modern computers

Using 1's complement	Using 2's complement
1100 (-3)	1101 (-3)
+ 1101 (-2)	+ 1110 (-2)
-----------------	------------------
1001 (-6)	1011 (-5)
Overflow check needed	Ignore overflow

Addition

There are four rules that need to be followed when adding two binary numbers.

- 0 + 0 = 0
- 1 + 0 = 1
- 1 + 1 = 10 (binary for 2)
- 1 + 1 + 1 = 11 (binary for 3)

Two examples of adding four-bit numbers:

0001	+1		0100	+4
+ 0101	+5		+ 1010	-6
0110	+6		1110	-2

Subtraction

Form the 2's-complement of the subtrahend (the bottom value) and then perform normal addition

This operation is done in exactly the same manner for both positive and negative numbers

1110	-2		1110	
- 1011	-5	⇒	+ 0101	
			0011	+3

Overflow

Overflow occurs when the answer does not fit in the number range

0110	+6	1110	-2
+ ^{1}0101	+5	+ ^{1}1001	-7
1011	+11	0111	-9

The answers are incorrect!

- For signed integers, the leftmost bit is used to indicate the sign:
  - 0 for positive
  - 1 for negative
- Range: -2^{n-1} to $+2^{n-1}-1$
  - 4 bits: -8 to 7 (-2^3 to 2^3-1) , 8 bits: -128 to 127 (-2^7 to 2^7-1)

Sign extension

Replicate the sign bit to extend 4-bit signed integers to 8-bit signed integers

0 1 0 1	→	0 0 0 0 0 1 0 1
1 1 1 0	→	1 1 1 1 1 1 1 0