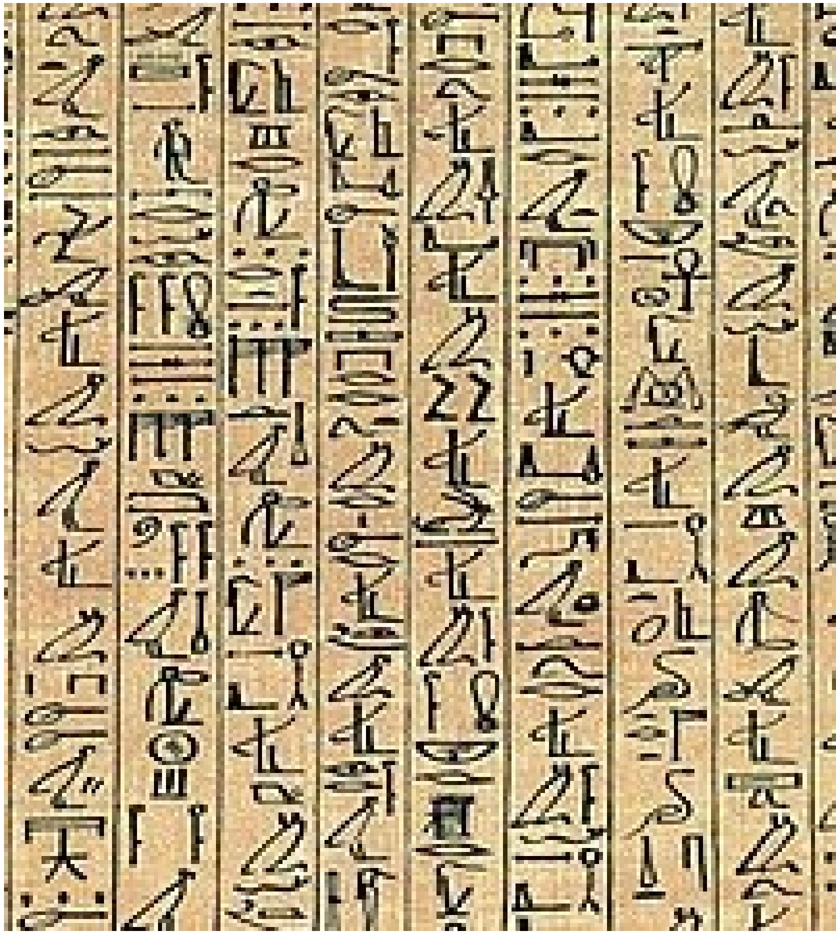
CHARACTER CODES

Compiled By: Afaq Alam Khan

Character Codes

- Unless you know the encoding scheme, there is no way that you can decode the data.
- Computer memory location merely *stores a binary pattern*. It is entirely up to you, as the programmer, to decide on how these patterns are to be *interpreted*.
- The 8-bit binary pattern "0100 0001" can be interpreted as an unsigned integer 65, or an ASCII character 'A‘



Egyptian hieroglyphs

Introduction

- Numerical symbols (0 to 9), Alphabets (A to Z, a to z) and special characters (#, &, …….) are represented by **codes** using binary digits 0 and 1, arranged according to the rules of specific scheme.
- **Schemes/ Types of codes**
  - BCD (Binary coded decimal)
  - Excess-3 Code
  - Gray Code
  - Alphanumeric Code
    - ASCII
    - EBCDIC
    - Unicode

BCD (Binary Coded Decimal)

- Four-bit code used to represents one of the ten decimal digits(Symbols) from 0 to 9.
- The following are the 4-bit binary representation of decimal values (Symbols):

0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001

- Remaining combinations 1010, 1011, 1100, 1101, 1110, 1111 are not used.
- Each bit position has a weight associated with it (weighted code). Weights are: 8, 4, 2, and 1 from MSB to LSB (**called 8-4-2-1 code**)
- Example: $(37)_{10}$ is represented as 0011 0111 using BCD code, rather than $(100101)_2$ in straight binary code.

BCD (Binary Coded Decimal)

- Example

Decimal Number	BCD Code	Binary Equal
5	0101	0101
9	1001	1001
58	0101 1000	111010
170	0001 0111 0000	10101010

Excess-3 Code (XS-3)

- Four bit code
- Excess-3 code is derived from 8421(BCD) code by adding 3(0011) to all code groups.
- Example - decimal **2** is coded as 0010 + 0011 = 0101 as Excess-3 code.
- It not weighted code.
- Its self-complimenting code, means 1's complement of the coded number yields 9's complement of the number itself.

Decimal Number	Excess-3 Code
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

Excess-3 Code (XS-3)

- Example

Decimal	BCD	Excess-3
8	1000	8 + 3 = 11→ **1011**
13	0001 0011	1 + 3 = 4 → 0100 , 3 + 3 = 6 → 0110 **0100 0110**
562	0101 0110 0010	5+3 =8 → 1000, 6+3=9→1001, 2+3=5→ 0101 **100010010101**

- **Self complementing property**

Decimal	Excess -3
2	**0101**
9's Complement 9-2 = 7	**I's Complement = 1010**
7	**1010**

Excess-3 Code (XS-3)

- **Exercise 1**: Encode the following decimal numbers into BCD and excess-3 codes
  - A) 1548
  - B) 7896

- **Exercise 2**: Decode the following Excess-3 numbers
  - A) 01110100
  - B) 100001010110

Gray Code

- It is the non-weighted code and it is not arithmetic codes. That means there are no specific weights assigned to the bit position. It has a very special feature that, only one bit will change each time the decimal number is incremented

- The gray code is a cyclic code

- the Gray code exhibits only a single bit change from one code word to the next in sequence. This property is important in many applications, such as shaft position encoders.

Decimal Number	Gray Code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

Gray Code

- **Example: Show the Gray code of 22**

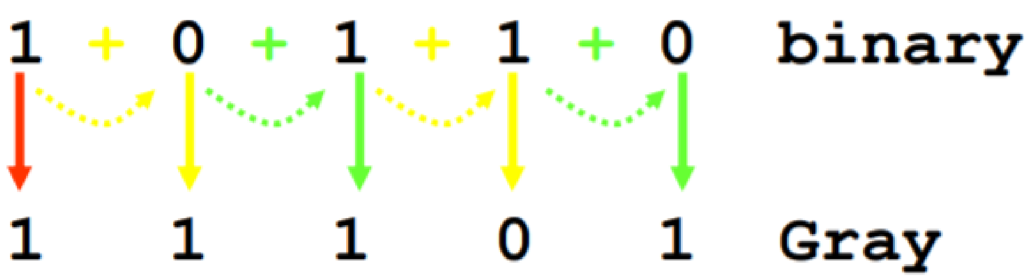
e.g., $(22)_{10} = (\ ?\ )_{Gray}$

Solution:

Step 1: $(22)_{10} = (\ 10110\ )_2 = (\ ?\ )_{Gray}$

Step 2: The MSB in the Gray code is the same as corresponding MSB in the binary number.

Step 3: Going from left to right, add each adjacent pair of binary code bits to get the next Gray code bit, discarding carries.



$(22)_{10} = (\ 10110\ )_2 = (\ 11101\ )_{Gray}$

Gray Code

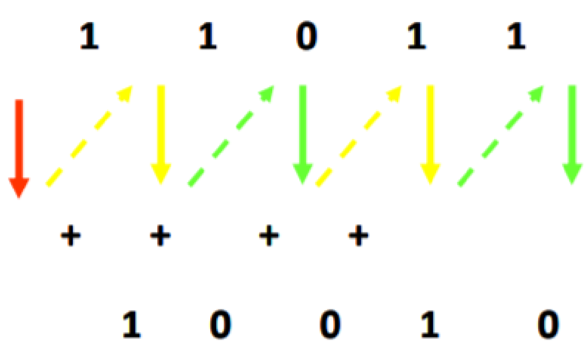
- **Example: Convert the Gray code 11011 to Binary and then to decimal.**

$(11011)_{Gray} = (\;?\;)_2$

Solution:

Step 1: The MSB in the binary code is the same as the corresponding bit in the Gray code.

Step 2: Add each binary code bit generated to the Gray code bit in the next adjacent position, discarding carries.



$$(11011)_{Gray} = (10010)_2 = 18$$

Gray Code

- **Exercise 3** :Find the Gray equivalent of the following binary numbers
  - A) 100010111
  - B) 111010110

- **Exercise 4**: Find the binary equivalent of the following gray codes
  - A) 101010101
  - B) 10010101111

Alphanumeric codes

- Represent numbers and alphabetic characters. Also represent other characters such as symbols and various instructions necessary for conveying information.
- Most Common
  - ASCII
  - EBSDIC
  - UniCode

ASCII

- American Standard Code for Information Interchange
- ASCII is originally a 7-bit code. It has been extended to 8-bit to better utilize the 8-bit computer memory organization.
- Code numbers 32D (20H) to 126D (7EH) are printable (displayable) characters as tabulated (arranged in hexadecimal and decimal) as follows
- Code number 32D (20H) is the *blank* or *space* character.

ASCII – Arranged in Hexadecimal

<table>
<tr><th>Hex</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><td>2</td><td>SP</td><td>!</td><td>"</td><td>#</td><td>$</td><td>%</td><td>&</td><td>'</td><td>(</td><td>)</td><td>*</td><td>+</td><td>,</td><td>-</td><td>.</td><td>/</td></tr>
<tr><td>3</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>:</td><td>;</td><td><</td><td>=</td><td>></td><td>?</td></tr>
<tr><td>4</td><td>@</td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td><td>O</td></tr>
<tr><td>5</td><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td><td>Y</td><td>Z</td><td>[</td><td>\</td><td>]</td><td>^</td><td>_</td></tr>
<tr><td>6</td><td>`</td><td>a</td><td>b</td><td>c</td><td>d</td><td>e</td><td>f</td><td>g</td><td>h</td><td>i</td><td>j</td><td>k</td><td>l</td><td>m</td><td>n</td><td>o</td></tr>
<tr><td>7</td><td>p</td><td>q</td><td>r</td><td>s</td><td>t</td><td>u</td><td>v</td><td>w</td><td>x</td><td>y</td><td>z</td><td>{</td><td>|</td><td>}</td><td>~</td><td></td></tr>
</table>

ASCII – Arranged in Decimal

Dec	0	1	2	3	4	5	6	7	8	9
3			SP	!	"	#	$	%	&	'
4	(	)	*	+	,	-	.	/	0	1
5	2	3	4	5	6	7	8	9	:	;
6	<	=	>	?	@	A	B	C	D	E
7	F	G	H	I	J	K	L	M	N	O
8	P	Q	R	S	T	U	V	W	X	Y
9	Z	[	\	]	^	_	`	a	b	c
10	d	e	f	g	h	i	j	k	l	m
11	n	o	p	q	r	s	t	u	v	w
12	x	y	z	{	|	}	~			

DEC	HEX	Meaning		DEC	HEX	Meaning	
0	00	NUL	Null	17	11	DC1	Device Control 1
1	01	SOH	Start of Heading	18	12	DC2	Device Control 2
2	02	STX	Start of Text	19	13	DC3	Device Control 3
3	03	ETX	End of Text	20	14	DC4	Device Control 4
4	04	EOT	End of Transmission	21	15	NAK	Negative Ack.
5	05	ENQ	Enquiry	22	16	SYN	Sync. Idle
6	06	ACK	Acknowledgment	23	17	ETB	End of Transmission
7	07	BEL	Bell	24	18	CAN	Cancel
8	08	BS	Back Space '\b'	25	19	EM	End of Medium
9	**09**	**HT**	**Horizontal Tab '\t'**	26	1A	SUB	Substitute
10	**0A**	**LF**	**Line Feed '\n'**	27	1B	ESC	Escape
11	0B	VT	Vertical Feed	28	1C	IS4	File Separator
12	0C	FF	Form Feed 'f'	29	1D	IS3	Group Separator
13	**0D**	**CR**	**Carriage Return '\r'**	30	1E	IS2	Record Separator
14	0E	SO	Shift Out	31	1F	IS1	Unit Separator
15	0F	SI	Shift In				
16	10	DLE	Datalink Escape	127	7F	DEL	Delete

Code numbers 0D (00H) to 31D (1FH), and 127D (7FH) are special control characters, which are non-printable (non-displayable)

EBCDIC

- **EBCDIC** (Extended Binary Coded Decimal Interchange Code)

Self study

Unicode (aka ISO/IEC 10646 Universal Character Set)

- Before Unicode, no single character encoding scheme could represent characters in all languages.
- For example, western european uses several encoding schemes. single language like Chinese has a few encoding schemes. Many encoding schemes are in conflict of each other, i.e., the same code number is assigned to different characters.

- Unicode aims to provide a standard character encoding scheme, which is universal, efficient, uniform and unambiguous. Unicode standard is maintained by a non-profit organization called the Unicode Consortium (www.unicode.org). Unicode is an ISO/IEC standard 10646.
- Unicode is backward compatible with ASCII etc. That is, the first 128 characters are the same as US-ASCII

- Unicode originally uses 16 bits (called UCS-2 or Unicode Character Set - 2 byte), which can represent up to 65,536 characters. It has since been expanded to more than 16 bits, currently stands at 21 bits. covering all current and ancient historical scripts.

Unicode (aka ISO/IEC 10646 Universal Character Set)

- The original 16-bit range of U+0000H to U+FFFFH (65536 characters) is known as *Basic Multilingual Plane* (BMP), covering all the major languages in use currently. The characters outside BMP are called *Supplementary Characters*, which are not frequently-used.

- **Unicode has two encoding schemes:**
- **UCS-2** (Universal Character Set - 2 Byte): Uses 2 bytes (16 bits), covering 65,536 characters in the BMP. BMP is sufficient for most of the applications. UCS-2 is now obsolete. [UTF-16]

- **UCS-4** (Universal Character Set - 4 Byte): Uses 4 bytes (32 bits), covering BMP and the supplementary characters.

  [UTF-32]

Thank you

Egyptian hieroglyphs

- Egyptian hieroglyphs were used by the ancient Egyptians since 4000BC. Unfortunately, since 500AD, no one could longer read the ancient Egyptian hieroglyphs, until the re-discovery of the Rosette Stone in 1799 by Napoleon's troop (during Napoleon's Egyptian invasion) near the town of Rashid (Rosetta) in the Nile Delta.
- The Rosetta Stone is inscribed with a decree in 196BC on behalf of King Ptolemy V. The decree appears in *three* scripts: the upper text is *Ancient Egyptian hieroglyphs*, the middle portion Demotic script, and the lowest *Ancient Greek*. Because it presents essentially the same text in all three scripts, and Ancient Greek could still be understood, it provided the key to the decipherment of the Egyptian hieroglyphs.
- The moral of the story is unless you know the encoding scheme, there is no way that you can decode the data.