

Binary Number Example

The number **1101 1011** is ...

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	0	1	1	0	1	1

$$128 + 64 + 16 + 8 + 2 + 1 = 219$$

Spring 2023

Seckman@Stan - CS&E - CS103

7

7

Hexadecimal Numbers

- Writing out long binary numbers is cumbersome and error prone
- As a result, computer scientists often write computer numbers in hexadecimal
- Hexadecimal is base-16
 - we only have 0 ... 9 to represent digits
 - So, hex uses **A ... F** to represent **10 ... 15**

Spring 2023

Seckman@Stan - CS&E - CS103

8

8

Hexadecimal Numbers

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

Spring 2023

Seckman@Stan - CS&E - CS103

9

9

Hex Example

The number **7AC** is ...

16^4	16^3	16^2	16^1	16^0
65536	4096	256	16	1
0	0	7	A	C

$$(7 \times 256) + (10 \times 16) + (12 \times 1) = 1964$$

Spring 2023

Seckman@Stan - CS&E - CS103

10

10

Converting Binary to Hex = Easy

- Since $16^1 = 2^4$, a single hex character can represent a total of 4 bits
- Convert every 4-bits to a single hexadecimal digit

5				C			
0	1	0	1	1	1	0	0

Spring 2023

Seckman@Stan - CS&E - CS103

11

11

Bits and Bytes

- Everything in a *modern* computer is stored using combination of ones and zeros
- Bit** is one binary digit
 - either 1 or 0
 - shorthand for a bit is b
- Byte** is a group of 8 bits
 - e.g. **1101 0100**
 - shorthand for a byte is B

Spring 2023

Seckman@Stan - CS&E - CS103

12

12

The Byte

0 1 1 0 1 1 0 1

Nibble

bit

Spring 2023 Beckmann, Stein - Comp. CSU 20 13

13

Text in Programming Languages

Text in Programming Languages

Press Any Key to Continue

Spring 2023 Beckmann, Stein - Comp. CSU 20 14

14

How Text Is Stored

- Computer often store and transmit textual data
- Examples:
 - punctuation
 - numerals 0 – 9
 - letter
- Each of these symbols is called a *character*

Spring 2023 Beckmann, Stein - Comp. CSU 20 15

15

Characters

- Processors rarely know what a "character" is, and instead store each as an integer
- In this case, each character is given a unique value
- For instance
 - "A", could have the value of 1
 - "B" is 2
 - "C" is 3, etc...

Spring 2023 Beckmann, Stein - Comp. CSU 20 16

16

Characters

- Characters and their matching values are a *character set*
- There have been many characters sets developed over time

Spring 2023 Beckmann, Stein - Comp. CSU 20 17

17

Character Sets

- ASCII
 - 7 bits – 128 characters
 - uses a full byte, one bit is not used
 - created in the 1967
- EBCDIC
 - Alternative system used by old IBM systems
 - Not used much anymore

Spring 2023 Beckmann, Stein - Comp. CSU 20 18

18

ASCII Chart

Control characters

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	sp	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

19

ASCII Codes

- Each character has a unique value
- The following is how "OMG" is stored in ASCII

	Binary	Hex	Decimal
O	0100 1111	4F	79
M	0100 1101	4D	77
G	0100 0111	47	71

20

ASCII Codes

- ASCII is laid out very logically
- Alphabetic characters (uppercase and lowercase) are 32 "code points" apart

	Decimal	Hex	Binary
A	65	41	01000001
a	97	61	01100001

21

ASCII Codes

- $32^1 = 2^5$
- 1-bit difference between upper and lowercase letters
- Printers can easily convert between the two

	Decimal	Hex	Binary
A	65	41	01000001
a	97	61	01100001

22

ASCII: Number Characters

- ASCII code for 0 is 30h
- Notice that the actual value of a number character is stored in the lower nibble
- So, the characters 0 to 9 can be easily converted to their binary values

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

23

ASCII: Number Characters

- Character → Binary
 - clear the upper nibble
 - Bitwise And: 0000 1111
- Binary → Character
 - set the upper nibble to 0011
 - Bitwise Or: 0011 0000

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

24

Times have changed

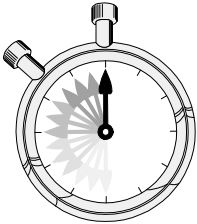


A New Character Set Enters the Scene

25

Times have changed...

- Computers have changed quite a bit since the 1960's
- As a result, most of these clever control characters are no longer needed
- Backspace, DEL, and numerous others are **obsolete**



26

Only Control Characters Still Used

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENO	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	sp	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

27

Unicode Character Set

- ASCII is only good for the United States
 - Other languages need additional characters
 - Multiple competing character sets were created
- Unicode was created to support every spoken language
- Developed in Mountain View, California

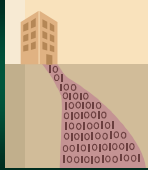
28

Unicode Character Set

- Originally used 16 bits
 - that's over 65,000 characters!
 - includes every character used in the World
- Expanded to 21 bits
 - 2 million characters!
 - now supports every character ever created
 - ... and emojis
- Unicode can be stored in different formats

29

Computer Memory

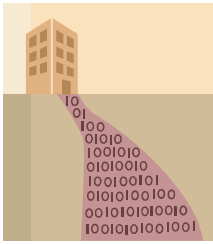


Its... um.... I forgot....

30

Computer Memory

- Programs access and manipulate memory far more than you realize
- So, understanding it...
 - is vital to becoming a great assembly programmer
 - and understanding computer architecture

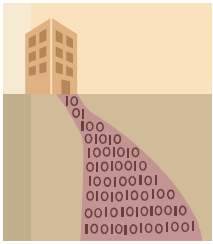


Spring 2023 Sacramento State - CS&E - CSJ 35 31

31

What is Memory?

- Memory is essentially an **enormous** array
- It is also, sometimes, referred to as **storage**
- It stores **both** running programs and their related data



Spring 2023 Sacramento State - CS&E - CSJ 35 32

32

Memory Addresses

- Memory is divided into storage locations that can hold 1 byte (8 bits)
- Each can be accessed using an **address**
 - unique value that refers to that specific byte
 - used to locate the exact byte the processor wants

Memory	
0	01000100
1	01000011
2	01101111
3	01101111
4	01101011

Spring 2023 Sacramento State - CS&E - CSJ 35 33

33

What is Memory?

- Each address is conceptually the same as an "index" in arrays
- ... and you will write access memory as would an array

Memory	
0	01000100
1	01000011
2	01101111
3	01101111
4	01101011

Spring 2023 Sacramento State - CS&E - CSJ 35 34

34

Memory is a Hardware Array

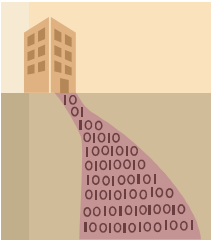
Memory	
00000000	
00000001	
00000002	
00000003	
⋮	
FFFFFFFF	

Spring 2023 Sacramento State - CS&E - CSJ 35 35

35

Memory Contains Data & Programs

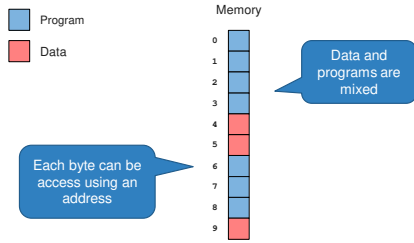
- Data and programs are just binary numbers (stored in a series of bytes)
- ...and are stored together
- Appreciating this is vital to understanding computer architecture



Spring 2023 Sacramento State - CS&E - CSJ 35 36

36

Memory Contains Data & Programs



Spring 2023

Seckman's Book - C++ - CSU 38

37