ÖZYEĞİN UNIVERSITY

CS105
Introduction to Object-Oriented Programming

Prof. Dr. Nizamettin AYDIN

naydin@itu.edu.tr

nizamettin.aydin@ozyegin.edu.tr

Information Systems Fundamentals

Outline

- Definition of Informatics
- Data - Information - I Knowledge
- Definition of System
- Information Systems
- Digital Systems
- A Digital Computer Example
- Transducers
- Analogue/Digital Conversion
- Sampling
- Measures in Computers
- Data Formats
- Data Representation
- Number Systems
- Codes
- Logic

Informatics

- The term informatics broadly describes the study and practice of creating, storing, finding, manipulating, sharing information.
- Etymology:
  - In 1956 the German computer scientist Karl Steinbuch coined the word Informatik
    - [*Informatik: Automatische Informationsverarbeitung*
      - ("Informatics: Automatic Information Processing")]
  - The French term informatique was coined in 1962 by Philippe Dreyfus
    - [Dreyfus, Phillipe. L'informatique. Gestion, Paris, June 1962, pp. 240–41]
  - The term was coined as a combination of information and automatic to describe the science of automating information interactions

Informatics

- The morphology
  - informat-ion + -ics
    - uses the accepted form for names of sciences,
      - as conics, linguistics, optics,
    - or matters of practice,
      - as economics, politics, tactics

- Linguistically, the meaning extends easily
  - to encompass both
    - the science of information
    - the practice of information processing

Data - Information - Knowledge

- Data
  - unprocessed facts and figures without any added interpretation or analysis.
    - {The price of crude oil is $80 per barrel.}
- Information
  - data that has been interpreted so that it has meaning for the user.
    - {The price of crude oil has risen from $70 to $80 per barrel}
- Process
  - Set of logically related tasks
- Knowledge
  - a combination of information, experience and insight that may benefit the individual or the organisation.
    - {When crude oil prices go up by $10 per barrel, it's likely that petrol prices will rise by 2p per litre.}
      - [insight: the capacity to gain an accurate and deep understanding of someone or something; an accurate and deep understanding]

Converting data into information

- Collecting data is expensive
  - you need to be very clear about why you need it and how you plan to use it.
  - One of the main reasons that organisations collect data is to monitor and improve performance.
    - if you are to have the information you need for control and performance improvement, you need to:
      - collect data on the indicators that really do affect performance
      - collect data reliably and regularly
      - be able to convert data into the information you need.

Converting data into information

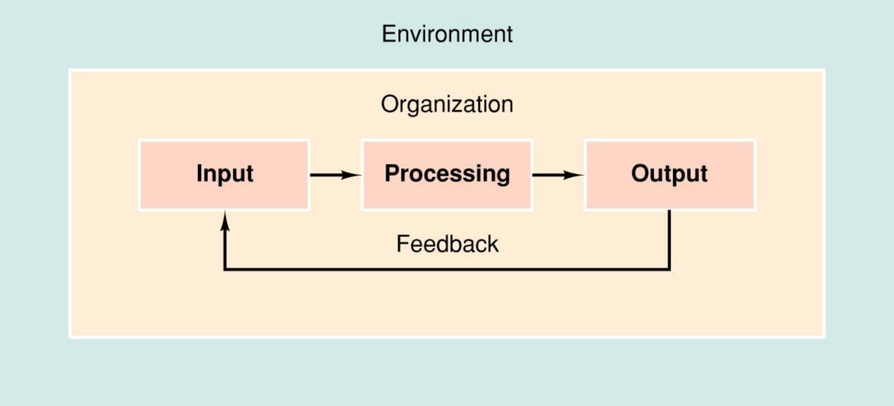
- To be useful, data must satisfy several conditions.
  - It must be:
    - relevant to the specific purpose
    - complete
    - accurate
    - timely
      - data that arrives after you have made your decision is of no value
    - in the right format
      - information can only be analyzed using a spreadsheet if all the data can be entered into the computer system
    - available at a suitable price
      - the benefits of the data must merit the cost of collecting or buying it.
- The same criteria apply to information.
  - It is important
    - to get the right information
    - to get the information right

Definition of system

- A system is an assembly of parts where:
  - –the parts or components are connected together in an organized way,
  - –the parts or components are affected by being in the system (and are changed by leaving it),
  - –the assembly does something,
  - –the assembly has been identified by a person as being of special interest.
- Any arrangement which involves the handling, processing or manipulation of resources of whatever type can be represented as a system.
- A system is defined as multiple parts working together for a common purpose or goal.

Input, Processing, Output, Feedback

- Systems are usually explained using a model.
  - A model helps to illustrate the major elements and their relationship



- Major elements of a model:
  - Input:
    - Activity of gathering and capturing raw data
  - Processing:
    - Converting data into useful outputs
  - Output:
    - Production of useful information, usually in the form of documents and reports
  - Feedback:
    - Information from the system that is used to make changes to input or processing activities

System Performance

- Efficiency
  - A measure of what is produced divided by what is consumed.
    - occurs when you reduce waste to produce a given number of goods or services

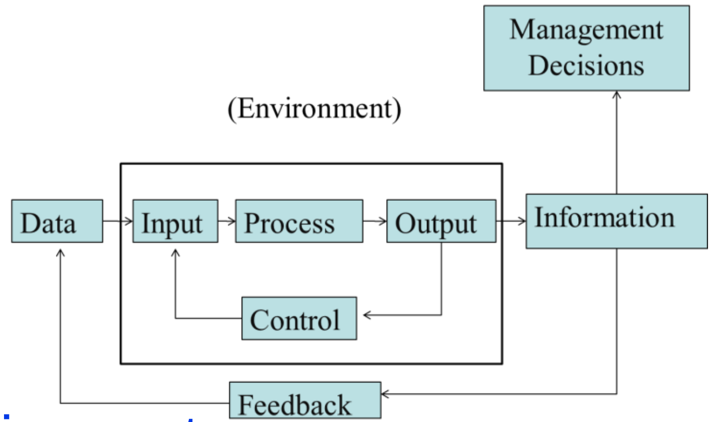
- Effectiveness
  - A measure of what is achieved divided by the stated goal.
  - An organization's ability to achieve its stated goals and objectives

Information Systems

- The ways that organizations
  - Store
  - Move
  - Organize
  - Process

  their information

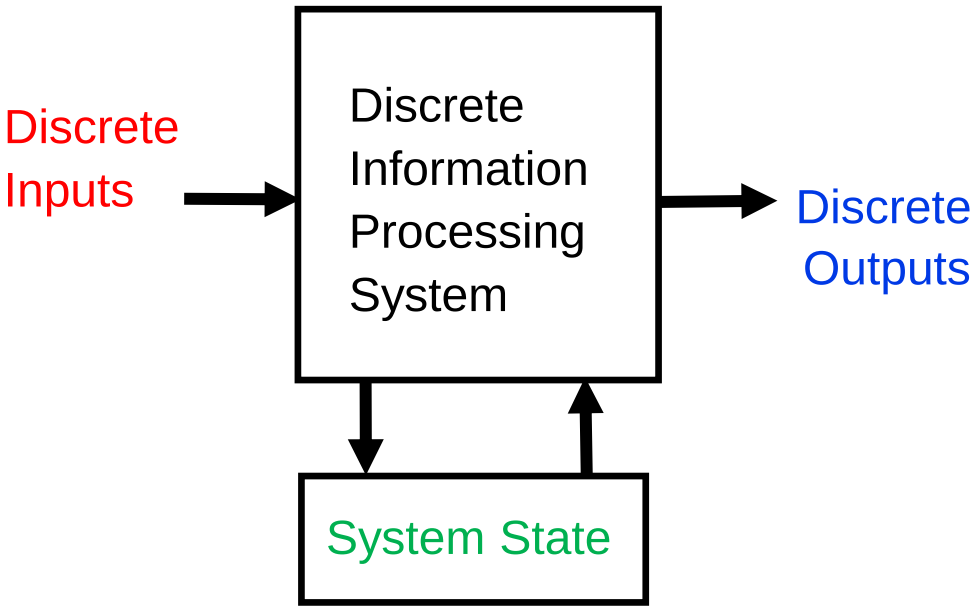
General Information Systems Diagram



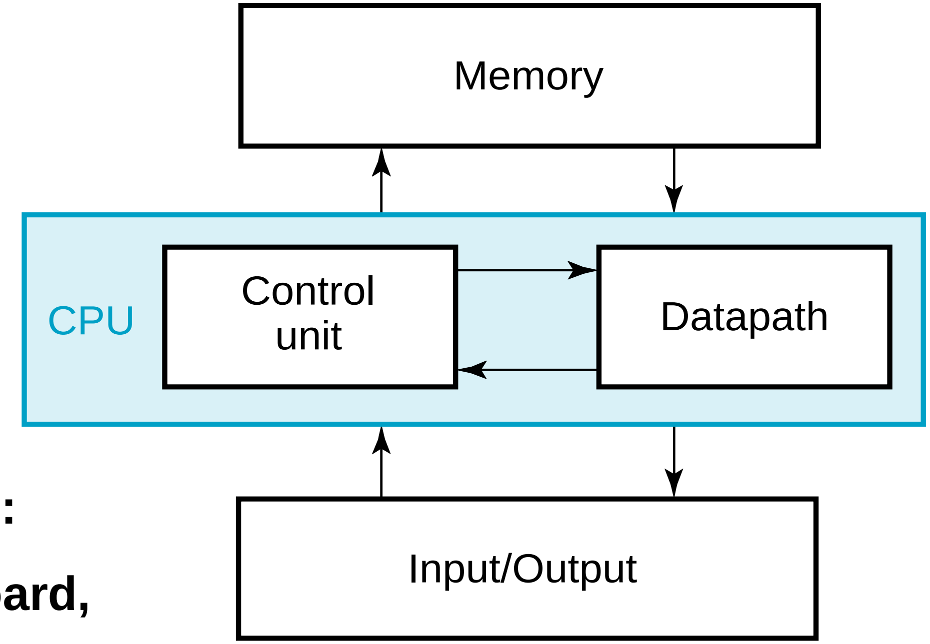
- Components that implement information systems,
  - Hardware
    - physical tools:
      - computer and network hardware, but also low-tech things like pens and paper
  - Software
    - (changeable) instructions for the hardware
  - People
  - Procedures
    - instructions for the people
  - Data/databases

Digital System

- Takes a set of discrete information (inputs) and discrete internal information (system state) and generates a set of discrete information (outputs).



A Digital Computer Example



Inputs:

Keyboard,

mouse,

modem,

microphone

Synchronous or Asynchronous?

Outputs:

CRT,

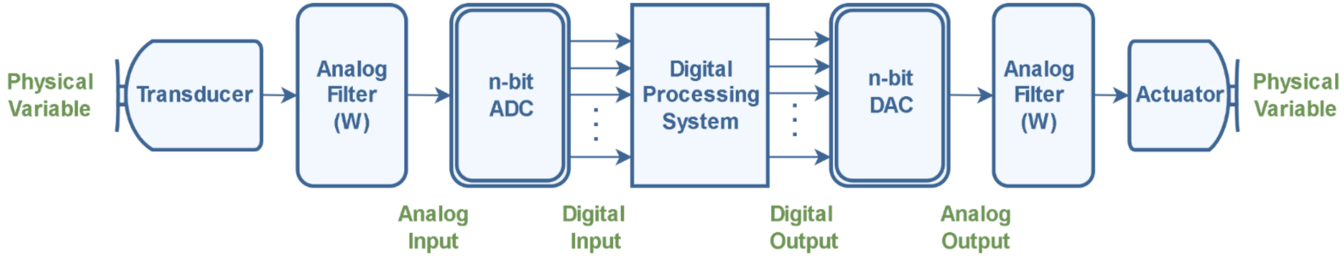
LCD,

modem,

speakers

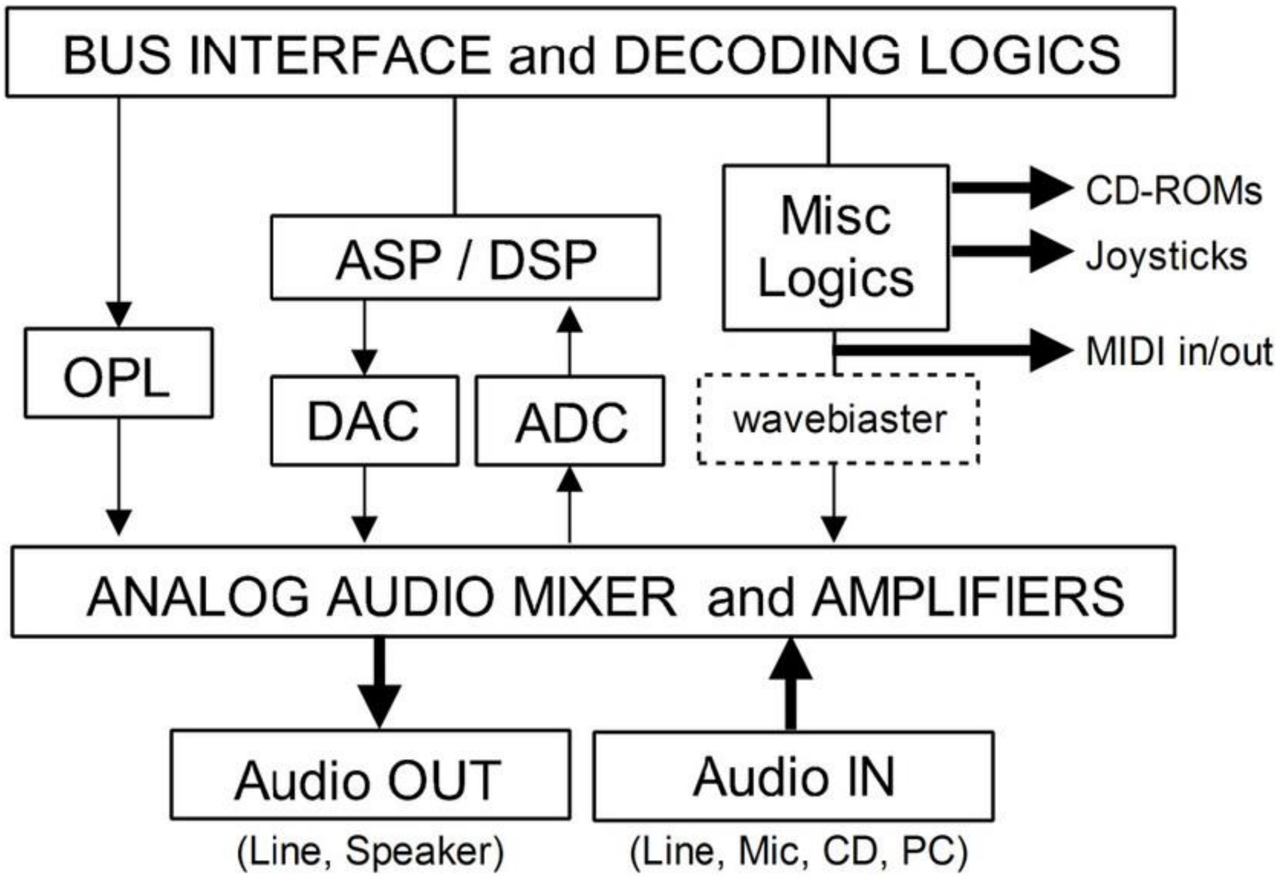
A generic complete digital processing system

- Interfacing digital processing system with the analog world using n-bit ADC and DAC



- A transducer is a device that converts energy from one form to another.
  - In signal processing applications, the purpose of energy conversion is to transfer information, not to transform energy.
- In measurement systems, transducers may be
  - input transducers (or sensors)
    - to convert a non-electrical energy into an electrical signal.
      - for example, a microphone.
  - output transducers (or actuators)
    - to convert an electrical signal into a non-electrical energy.
      - for example, a speaker.

Block diagram of sound card modules

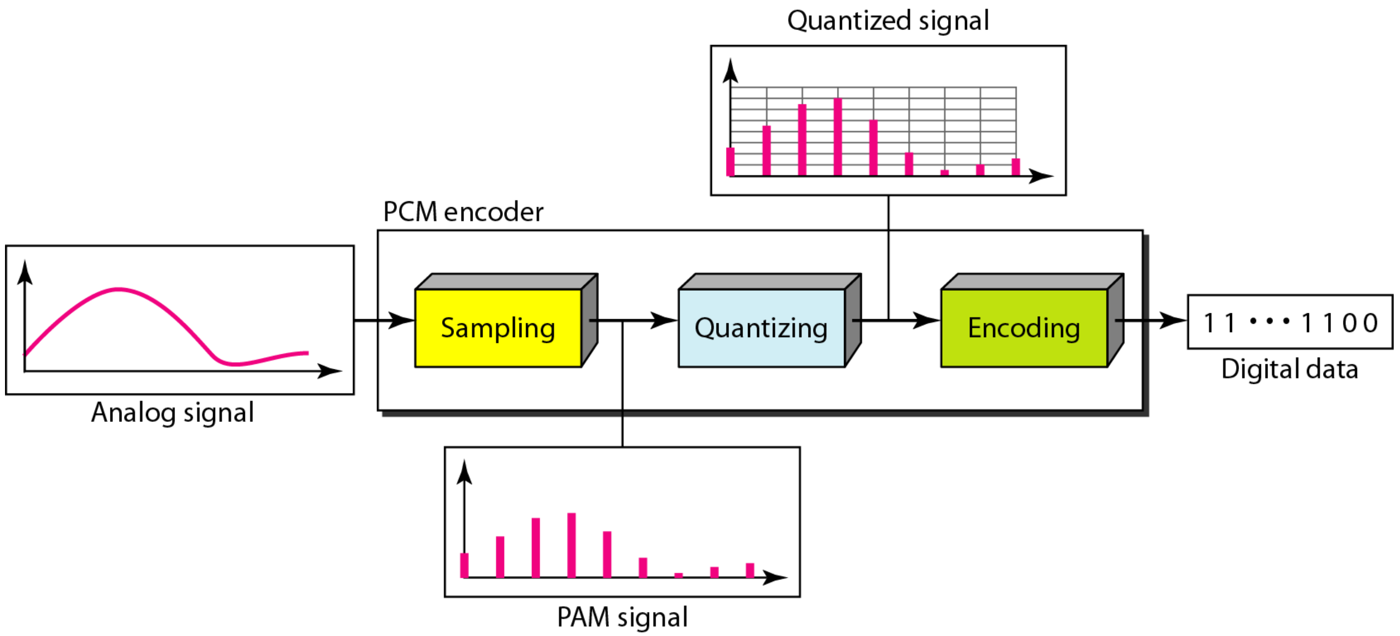


Analogue/Digital signal

- The analogue signal
  - –a continuous variable defined with infinite precision

  is converted to a discrete sequence of measured values using sampling and quantization and represented digitally
- Information is lost in converting from analogue to digital, due to:
  - –inaccuracies in the measurement
  - –uncertainty in timing
  - –limits on the duration of the measurement
- These effects are called quantization errors

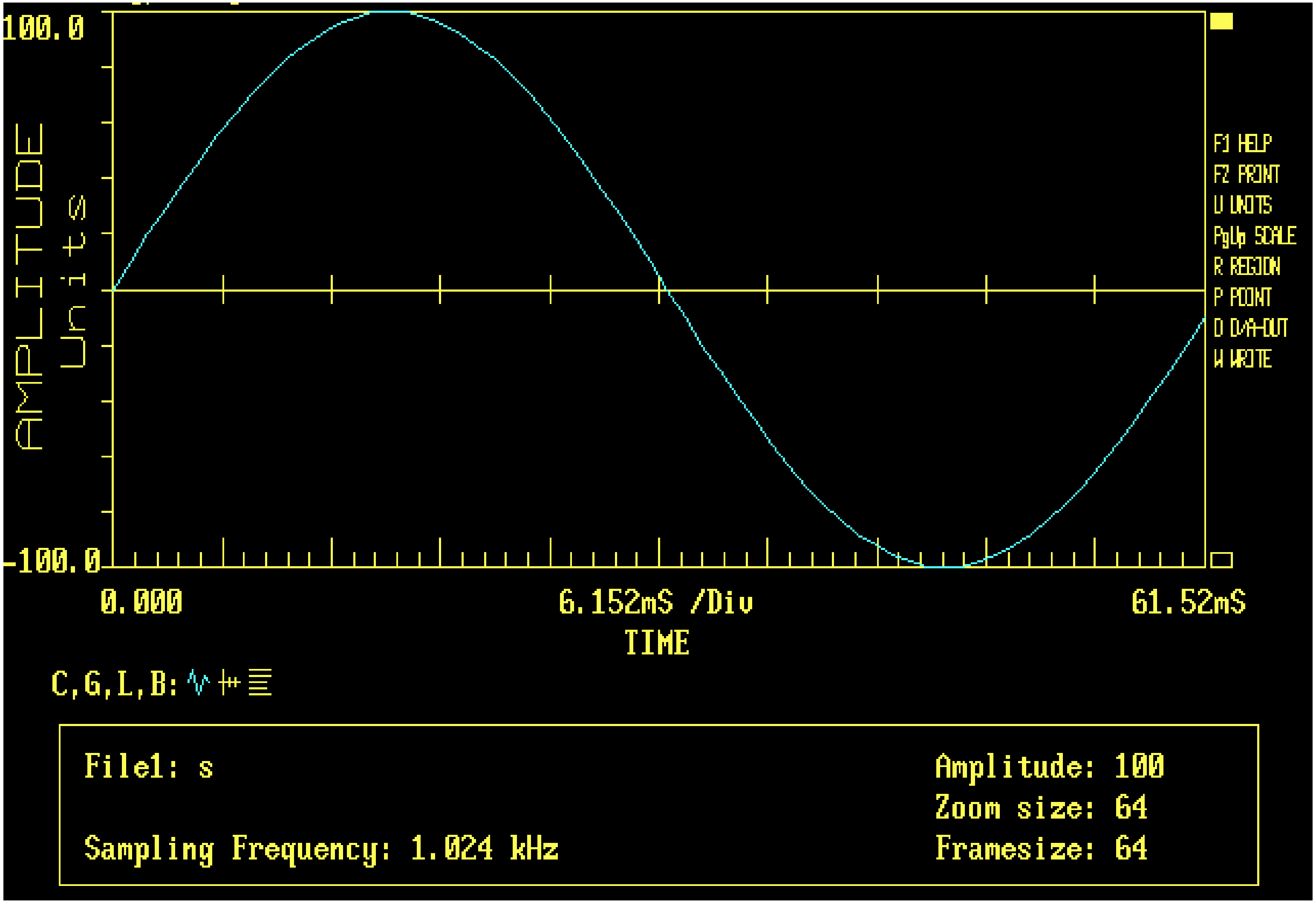
Analog-to Digital Conversion



Sampling

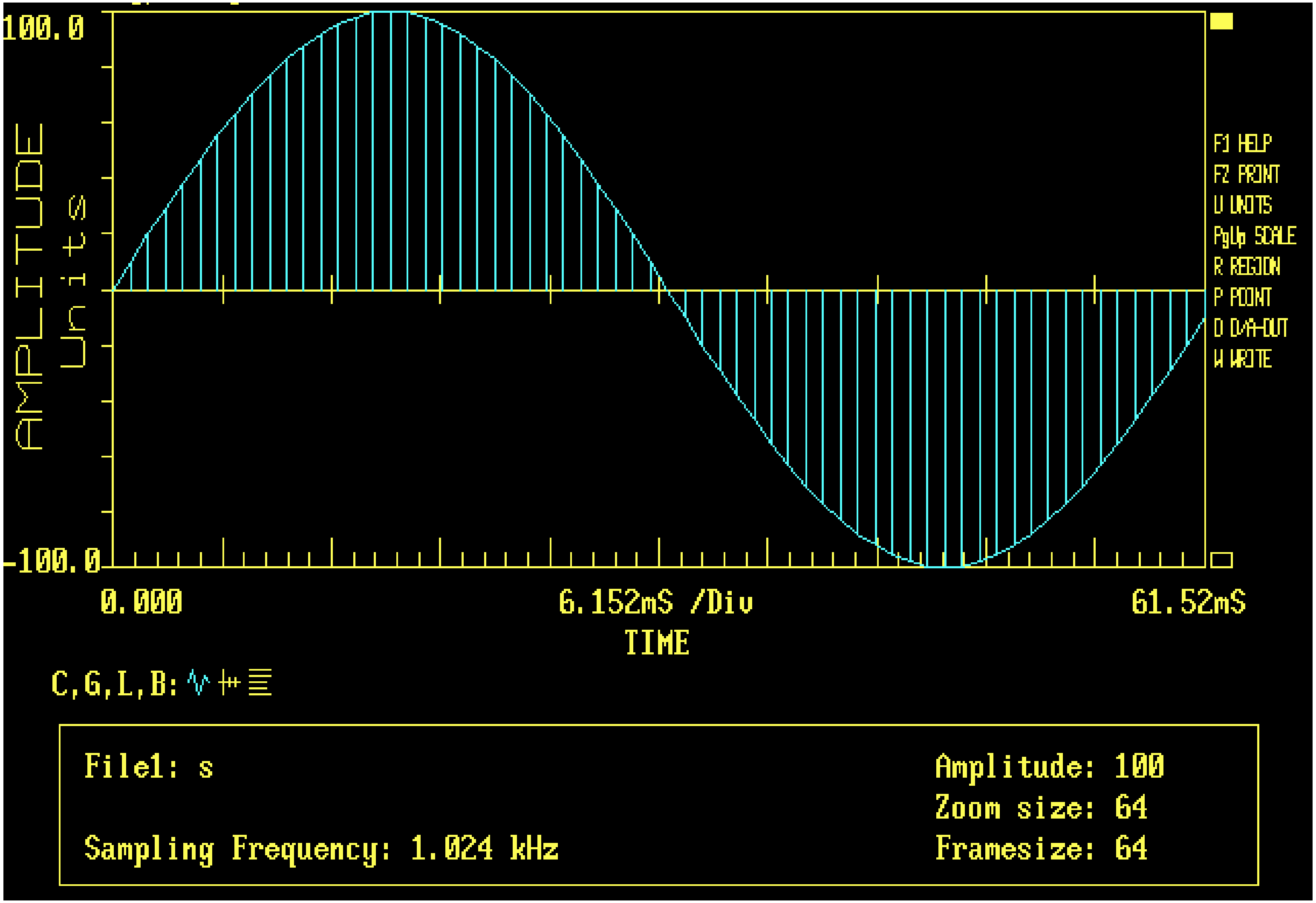
- The sampling results in a discrete set of digital numbers that represent measurements of the signal
  - usually taken at equal intervals of time
- Sampling takes place after the hold
  - The hold circuit must be fast enough that the signal is not changing during the time the circuit is acquiring the signal value
- We don't know what we don't measure
- In the process of measuring the signal, some information is lost

Sampling

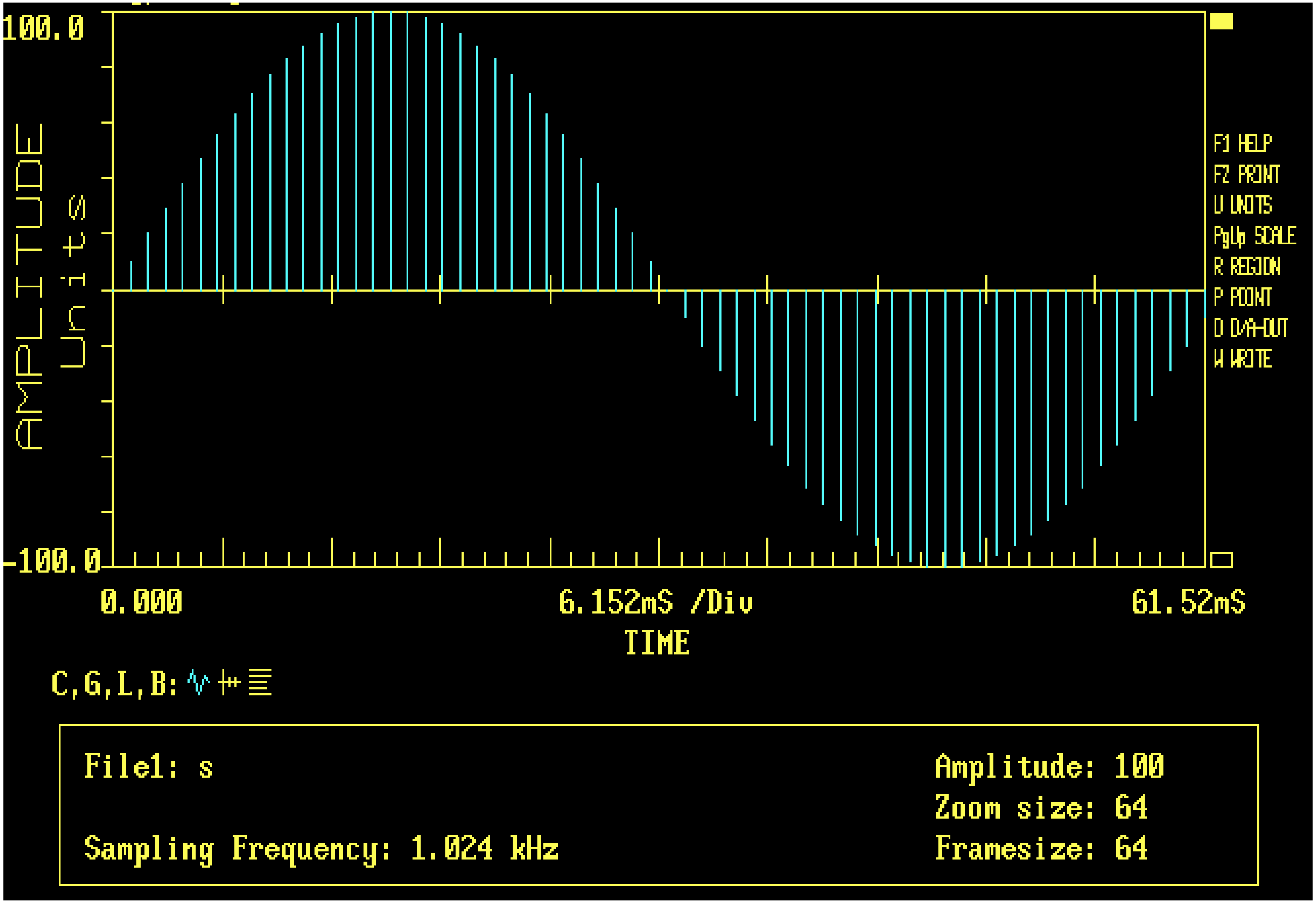


100.0
-100.0
AMPLITUDE
Units
F1 HELP
F2 PRINT
U UNITS
PgUp SCALE
R REGION
P POINT
D D/A-OUT
W WRITE
0.000
6.152mS /Div
61.52mS
TIME
C,G,L,B:
File1: s
Sampling Frequency: 1.024 kHz
Amplitude: 100
Zoom size: 64
Framesize: 64

Sampling


100.0
-100.0
AMPLITUDE
Units
F1 HELP
F2 PRINT
U UNITS
PgUp SCALE
R REGION
P POINT
D D/A-OUT
W WRITE
0.000
6.152mS /Div
61.52mS
TIME
C,G,L,B:
File1: s
Sampling Frequency: 1.024 kHz
Amplitude: 100
Zoom size: 64
Framesize: 64

Sampling

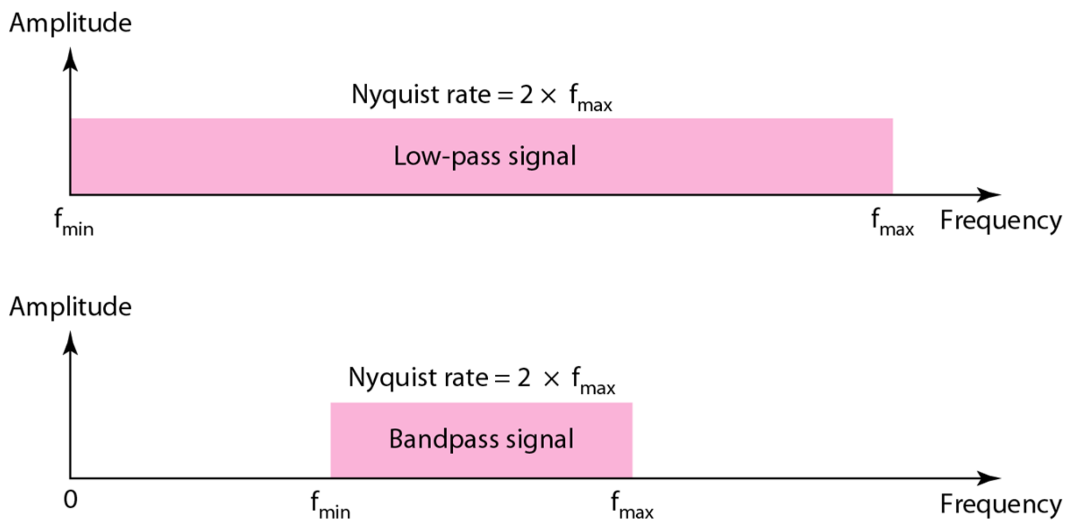


100.0
-100.0
AMPLITUDE
Units
F1 HELP
F2 PRINT
U UNITS
PgUp SCALE
R REGION
P POINT
D D/A-OUT
W WRITE
0.000
6.152mS /Div
61.52mS
TIME
C,G,L,B:
File1: s
Sampling Frequency: 1.024 kHz
Amplitude: 100
Zoom size: 64
Framesize: 64

Sampling Theorem

- According to the Nyquist theorem,
  - the sampling rate must be at least 2 times the highest frequency contained in the signal.

$$F_s \geq 2f_m$$



Quantization

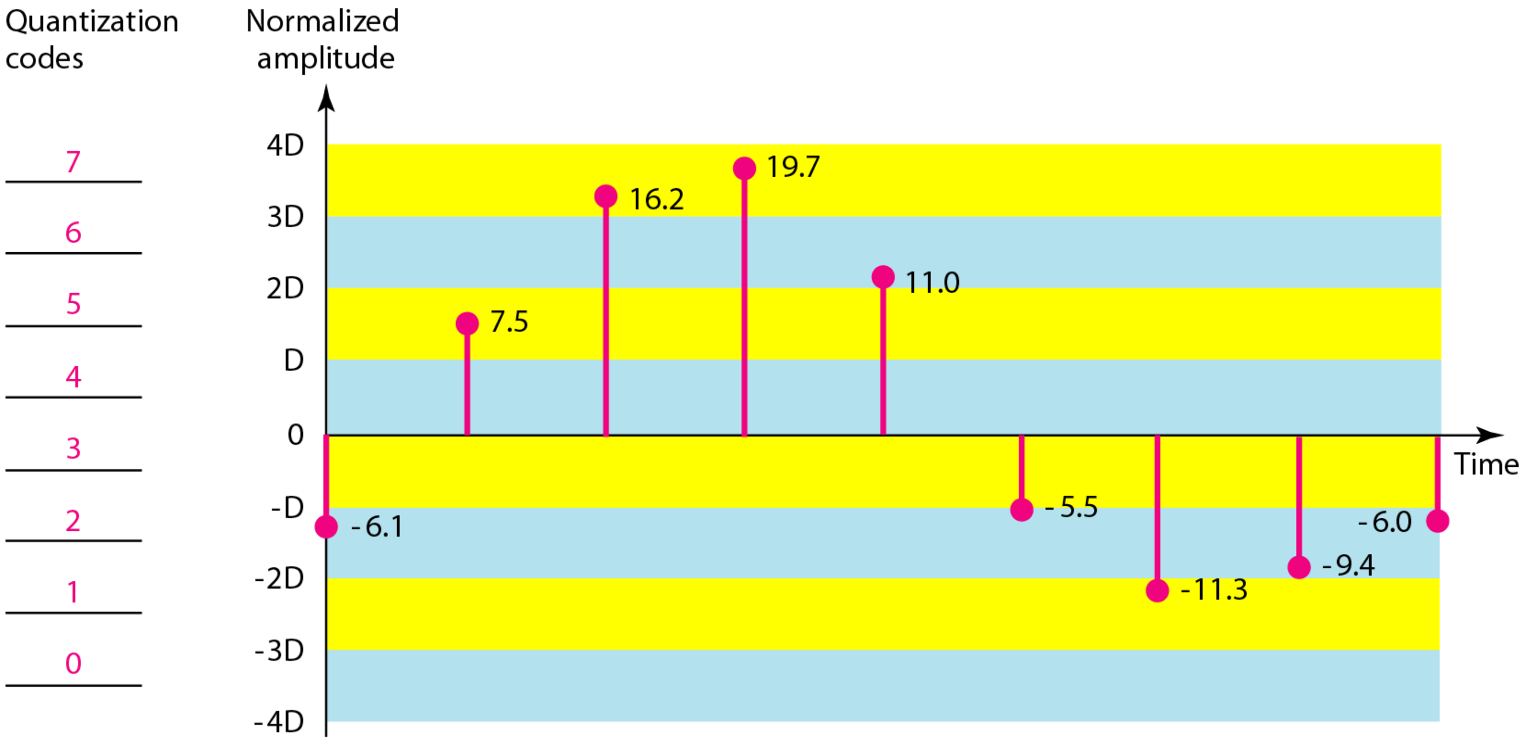
- Sampling results in a series of pulses of varying amplitude values ranging between two limits:
  - a min and a max.
- The amplitude values are infinite between the two limits.
- We need to map the infinite amplitude values onto a finite set of known values.
  - This is achieved by dividing the distance between min and max into L zones, each of height Δ.

$$\Delta = (max - min)/L$$

- Each zone is then assigned a binary code.
  - The number of bits required to encode the zones, or the number of bits per sample as it is commonly referred to, is obtained as follows:

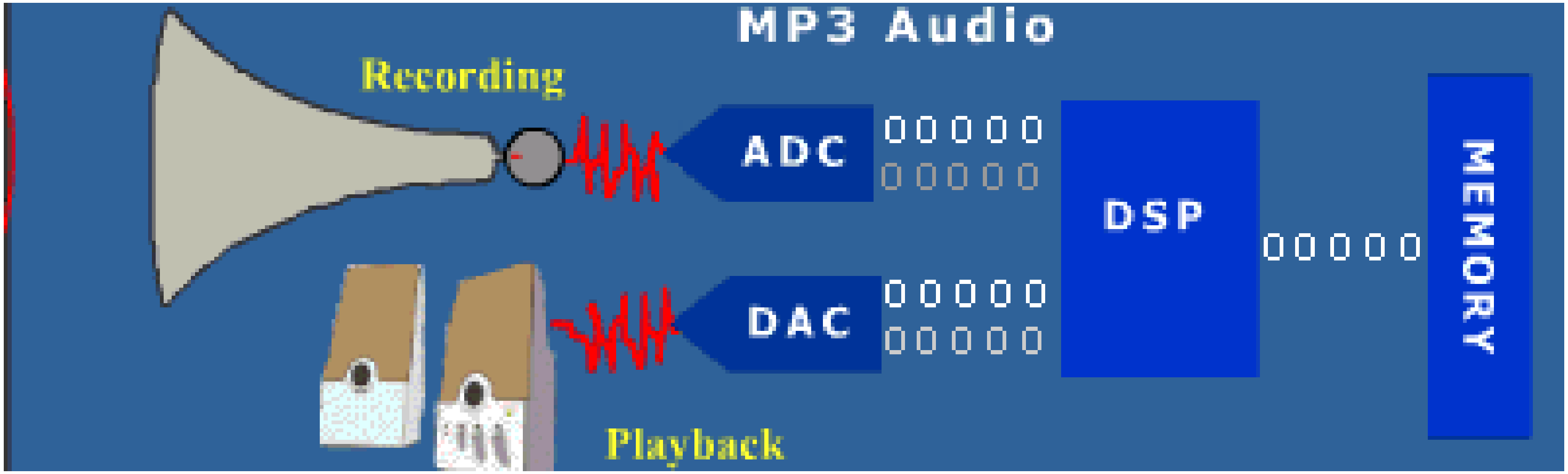
$$n_b = \log_2 L$$

Quantization and encoding of a sampled signal



Normalized PAM values	-1.22	1.50	3.24	3.94	2.20	-1.10	-2.26	-1.88	-1.20
Normalized quantized values	-1.50	1.50	3.50	3.50	2.50	-1.50	-2.50	-1.50	-1.50
Normalized error	-0.38	0	+0.26	-0.44	+0.30	-0.40	-0.24	+0.38	-0.30
Quantization code	2	5	7	7	6	2	1	2	2
Encoded words	010	101	111	111	110	010	001	010	010

Example - Digital Sound System



Measures in Computers

- Speed (power of 10) and Capacity (power of 2)
  - –Kilo- (K) = 1 thousand = 10^3 and 2^{10}
  - –Mega- (M) = 1 million = 10^6 and 2^{20}
  - –Giga- (G) = 1 billion = 10^9 and 2^{30}
  - –Tera- (T) = 1 trillion = 10^{12} and 2^{40}
  - –Peta- (P) = 1 quadrillion = 10^{15} and 2^{50}
- Time (power of 10) and Space (power of 10)
  - –Milli- (m) = 1 thousandth = 10^{-3}
  - –Micro- (μ) = 1 millionth = 10^{-6}
  - –Nano- (n) = 1 billionth = 10^{-9}
  - –Pico- (p) = 1 trillionth = 10^{-12}
  - –Femto- (f) = 1 quadrillionth = 10^{-15}

Example

- Hertz = clock cycles per second (frequency)
  - 1MHz = 1,000,000Hz
  - Processor speeds are measured in MHz or GHz.

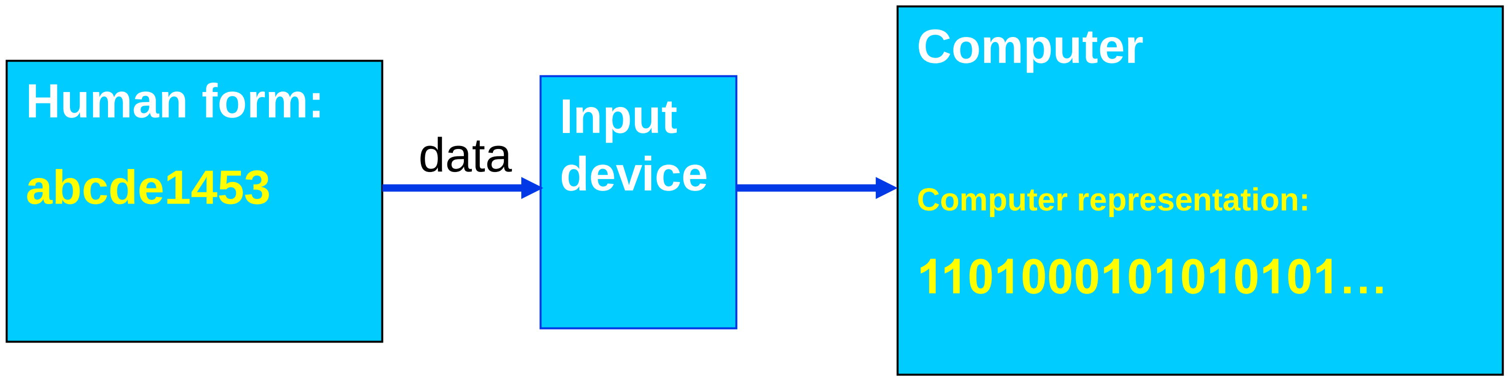
- Byte = a unit of storage
  - 1KB = 2^{10} = 1024 Bytes
  - 1MB = 2^{20} = 1 048 576 Bytes
  - 1GB = 2^{30} = 1 073 700 000 Bytes
  - 1TB = 2^{40} = 1 099 5 00 000 000 Bytes
  - Main memory (RAM) is measured in MB (or GB)
  - Disk storage is measured in GB for small systems, TB for large systems.

Data Formats

- Computers
  - Process and store all forms of data in binary format
- Human communication
  - Includes language, images and sounds
- Data formats:
  - Specifications for converting data into computer-usable form
  - Define the different ways human data may be represented, stored and processed by a computer

Sources of Data

- Binary input
  - Begins as discrete input
    - Example: keyboard input such as A 1+2=3 math
      - Keyboard generates a binary number code for each key
- Analog
  - Continuous data such as sound or images
    - Requires hardware to convert data into binary numbers



Common Data Representations

Type of Data	Standard(s)
Alphanumeric	Unicode, ASCII, EDCDIC
Image (bitmapped)	▪GIF (graphical image format) ▪TIF (tagged image file format) ▪PNG (portable network graphics)
Image (object)	PostScript, JPEG, SWF (Macromedia Flash), SVG
Outline graphics and fonts	PostScript, TrueType
Sound	WAV, AVI, MP3, MIDI, WMA
Page description	PDF (Adobe Portable Document Format), HTML, XML
Video	Quicktime, MPEG-2, RealVideo, WMV

Data Representation

- Reflects the
  - Complexity of input source
  - Type of processing required
- Trade-offs
  - Accuracy and resolution
    - Simple photo vs. painting in an art book
  - Compactness (storage and transmission)
    - More data required for improved accuracy and resolution
    - Compression represents data in a more compact form
    - Metadata
- Ease of manipulation:
  - Processing simple audio vs. high-fidelity sound
- Standardization
  - Proprietary formats for storing and processing data (WordPerfect vs. Word)
  - De facto standards: proprietary standards based on general user acceptance (PostScript)

Metadata

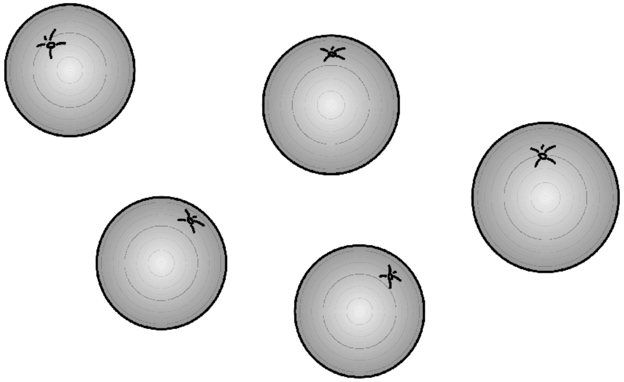
- A set of data that describes and gives information about other data.
  - Descriptive metadata
    - For finding or understanding a resource
  - Administrative metadata
    - Technical metadata
      - For decoding and rendering files
    - Preservation metadata
      - Long-term management of files
    - Rights metadata
      - Intellectual property rights attached to content
  - Structural metadata
    - Relationships of parts of resources to one another
  - Markup languages
    - Integrates metadata and flags for other structural or semantic features within content

What kinds of data do we need to represent?

- Numbers
  - signed, unsigned, integers, floating point, complex, rational, irrational, …
- Text
  - characters, strings, …
- Images
  - pixels, colors, shapes, …
- Sound
- Logical
  - true, false
- Instructions
- …
- Data type:
  - representation and operations within the computer

Numbers: Physical Representation

- Different numerals, same number of oranges
  - Cave dweller:
    - IIIII
  - Roman:
    - V
  - Arabic:
    - 5



- Different bases, same number of oranges
  - 5_{10}
  - 101_2
  - 12_3

Number System

- Roman: position independent
- Modern: based on positional notation (place value)
  - Decimal system:
    - system of positional notation based on powers of 10.
  - Binary system:
    - system of positional notation based on powers of 2.
  - Octal system:
    - system of positional notation based on powers of 8.
  - Hexadecimal system:
    - system of positional notation based on powers of 16.
- Positive radix, positional number systems
- A number with *radix* ***r*** is represented by a string of digits:

  $A_{n-1}A_{n-2} \ldots A_1A_0 \,.\, A_{-1}A_{-2} \ldots A_{-m+1}A_{-m}$

  in which $0 \leq A_i < r$ and **.** is the radix point.
- The string of digits represents the power series:

$$(Number)_r = \left(\sum_{i=0}^{i=n-1} A_i\, r^i + \sum_{j=-m}^{j=-1} A_{ij}\, r^j \right)$$

(Integer Portion) + (Fraction Portion)

Decimal Numbers

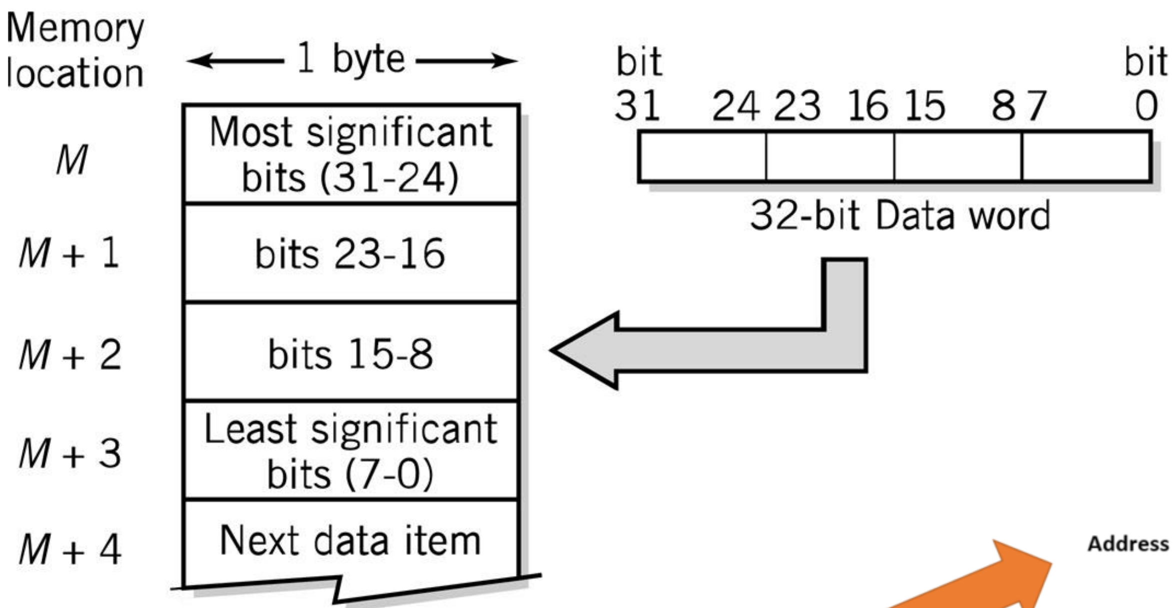
- Decimal means that we have ten digits to use in our representation
  - the symbols 0 through 9
- What is 3546?
  - it is *three* thousands plus *five* hundreds plus *four* tens plus *six* ones.
  - i.e. $3546 = 3\times10^3 + 5\times10^2 + 4\times10^1 + 6\times10^0$
- How about negative numbers?
  - we use two more symbols to distinguish positive and negative:

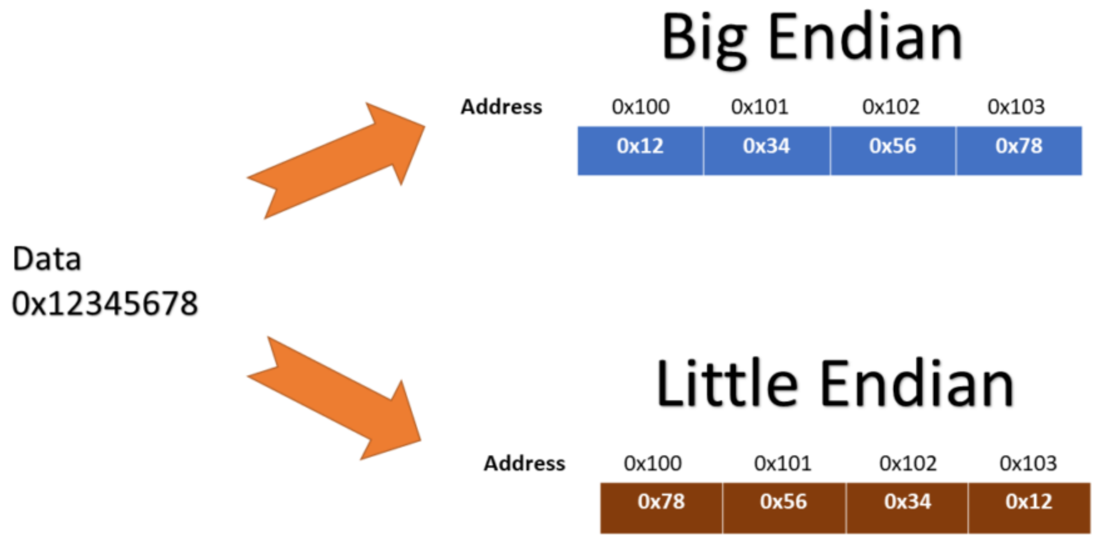
    + and **-**

Internal Computer Data Format

- All data stored as binary numbers
  - Interpreted based on
    - Operations computer can perform
    - Data types supported by programming language used to create application
- Simple Data Types
  - Boolean:
    - 2-valued variables or constants with values of true or false
  - Char:
    - Variable or constant that holds alphanumeric character
  - Enumerated:
    - User-defined data types with possible values listed in definition
      - Type DayOfWeek = Mon, Tues, Wed, Thurs, Fri, Sat, Sun
  - Integer:
    - positive or negative whole numbers
  - Real :
    - Numbers with a decimal point, whose magnitude (large or small) exceeds computer's capability to store as an integer

Representing Numbers - 32-bit Data Word





Unsigned Binary Integers

$$Y = \text{"abc"} = a.2^2 + b.2^1 + c.2^0$$

(where the digits a, b, c can each take on the values of 0 or 1 only)

N = number of bits

Range is:
$0 \leq i \leq (2^N - 1)$

	3-bits	5-bits	8-bits
0	000	00000	00000000
1	001	00001	00000001
2	010	00010	00000010
3	011	00011	00000011
4	100	00100	00000100

Problem:

- How do we represent negative numbers?

Signed Binary Integers - 2s Complement representation

- Transformation
  - To transform a into -a, invert all bits in a and add 1 to the result

Range is:

$$-2^{N-1} \leq i \leq (2^{N-1} - 1)$$

Advantages:

- Operations need not check the sign
- Only one representation for zero
- Efficient use of all the bits

-16	10000
…	…
-3	11101
-2	11110
-1	11111
0	00000
+1	00001
+2	00010
+3	00011
…	…
+15	01111

Signed Binary Integers - 2s Complement representation

- 2's Complement

$$-2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

+3 = 00000011 +2 = 00000010

+1 = 00000001 +0 = 00000000

-1 = 11111111 -2 = 11111110 -3 = 11111101

Range	-2^{n-1} through $2^{n-1} - 1$
Number of Representations of Zero	One
Negation	Take the Boolean complement of each bit of the corresponding positive number, then add 1 to the resulting bit pattern viewed as an unsigned integer.
Expansion of Bit Length	Add additional bit positions to the left and fill in with the value of the original sign bit.
Overflow Rule	If two numbers with the same sign (both positive or both negative) are added, then overflow occurs if and only if the result has the opposite sign.
Subtraction Rule	To subtract *B* from *A*, take the twos complement of *B* and add it to *A*.

Limitations of integer representations

- Most numbers are not integer!
  - Even with integers, there are two other considerations:
- Range:
  - The magnitude of the numbers we can represent is determined by how many bits we use:
    - e.g. with 32 bits the largest number we can represent is about +/- 2 billion, far too small for many purposes.
- Precision:
  - The exactness with which we can specify a number:
    - e.g. a 32 bit number gives us 31 bits of precision, or roughly 9 figure precision in decimal repesentation.
- We need another data type!

Real numbers

- Our decimal system handles non-integer real numbers by adding yet another symbol
  - the decimal point (.) to make a fixed point notation:
    - e.g. $3456.78 = 3.10^3 + 4.10^2 + 5.10^1 + 6.10^0 + 7.10^{-1} + 8.10^{-2}$
- The floating point, or scientific notation allows us to represent very large and very small numbers (integer or real), with as much or as little precision as needed:
  - Unit of electric charge e = $1.602\ 176\ 462 \times 10^{-19}$ Coulomb
  - Volume of universe = 1×10^{85} cm^3
    - the two components of these numbers are called the mantissa and the exponent

Real numbers in binary

- We mimic the decimal floating point notation to create a “hybrid” binary floating point number:
  - We first use a “binary point” to separate whole numbers from fractional numbers to make a fixed point notation:
    - e.g. 00011001.110 = $1.2^4 + 1.10^3 + 1.10^1 + 1.2^{-1} + 1.2^{-2}$ => 25.75

      (2^{-1} = 0.5 and 2^{-2} = 0.25, etc.)
  - We then “float” the binary point:
    - 00011001.110 => 1.1001110 x 2^4

      mantissa = 1.1001110, exponent = 4
  - Now we have to express this without the extra symbols by convention, we divide the available bits into three fields:

    sign, mantissa, exponent

Floating Point Representation (IEEE-754 fp)

	s	biased exp.	fraction
32 bits:	1	8 bits	23 bits

$$N = (-1)^{s} \times 1.\mathbf{fraction} \times 2^{(\text{biased exp.} - 127)}$$

- Sign: 1 bit
- Mantissa: 23 bits
  – We “normalize” the mantissa by dropping the leading 1 and recording only its fractional part
- Exponent: 8 bits
  – In order to handle both +ve and -ve exponents, we add 127 to the actual exponent to create a “biased exponent”:
    - 2^{-127} => biased exponent = 0000 0000 (= 0)
    - 2^{0} => biased exponent = 0111 1111 (= 127)
    - 2^{+127} => biased exponent = 1111 1110 (= 254)

Floating Point Representation (IEEE-754 fp)

- Example:
  - Find the corresponding fp representation of 25.75
    - 25.75 => 00011001.110 => 1.1001110 x 2^4
    - sign bit = 0 (+ve)
    - normalized mantissa (fraction) = 100 1110 0000 0000 0000 0000
    - biased exponent = 4 + 127 = 131 => 1000 0011
    - so 25.75 => 0 1000 0011 100 1110 0000 0000 0000 0000 => x41CE0000
- Values represented by convention:
  - Infinity (+ and -):
    - exponent = 255 (1111 1111) and fraction = 0
  - NaN (not a number):
    - exponent = 255 and fraction $\neq$ 0
  - Zero (0):
    - exponent = 0 and fraction = 0
      - note: exponent = 0 => fraction is *de-normalized,* i.e no hidden 1

Floating Point Representation (IEEE-754 fp)

- Double precision (64 bit) floating point

	s	biased exp.	fraction
64 bits:	1	11 bits	52 bits

$$N = (-1)^{s} \times 1.\textbf{fraction} \times 2^{(\textbf{biased exp.} - 1023)}$$

- Range & Precision:
  - 32 bit:
    - mantissa of 23 bits + 1 => approx. 7 digits decimal
    - $2^{+/-127}$ => approx. $10^{+/-38}$
  - 64 bit:
    - mantissa of 52 bits + 1 => approx. 15 digits decimal
    - $2^{+/-1023}$ => approx. $10^{+/-306}$

Binary Numbers and Binary Coding

- Flexibility of representation
  - Within constraints below, can assign any binary combination (called a code word) to any data as long as data is uniquely encoded.
- Information Types
  - Numeric
    - Must represent range of data needed
    - Very desirable to represent data such that simple, straightforward computation for common arithmetic operations permitted
    - Tight relation to binary numbers
  - Non-numeric
    - Greater flexibility since arithmetic operations not applied.
    - Not tied to binary numbers

Non-numeric Binary Codes

- Given n binary digits (called bits), a binary code is a mapping from a set of represented elements to a subset of the 2^n binary numbers.

- Example:
  - A binary code for the seven colors of the rainbow
    - Code 100 is not used

Color	Binary Number
Red	000
Orange	001
Yellow	010
Green	011
Blue	101
Indigo	110
Violet	111

Number of Bits Required

- Given M elements to be represented by a binary code, the minimum number of bits, n, needed, satisfies the following relationships:

  $2^n > M > 2^{(n-1)}$

  $n = \lceil \log_2 M \rceil$ where $\lceil x \rceil$, called the *ceiling function,* is the integer greater than or equal to x.

- Example:
  - How many bits are required to represent decimal digits with a binary code?
    - 4 bits are required ($n = \lceil \log_2 9 \rceil = 4$)

Number of Elements Represented

- Given n digits in radix r, there are r^n distinct elements that can be represented.
  - But, you can represent m elements,
    - $m < r^n$

- Examples:
  - You can represent 4 elements in radix $r = 2$ with $n = 2$ digits:
    - (00, 01, 10, 11).
  - You can represent 4 elements in radix $r = 2$ with $n = 4$ digits:
    - (0001, 0010, 0100, 1000).

Binary Coded Decimal (BCD)

- In the 8421 Binary Coded Decimal (BCD) representation each decimal digit is converted to its 4-bit pure binary equivalent
- This code is the simplest, most intuitive binary code for decimal digits and uses the same powers of 2 as a binary number,
  - –but only encodes the first ten values from 0 to 9.
    - For example: $(57)_{dec}$ ➔ $(?)_{bcd}$

(5 7) dec

= (0101 0111)bcd

Error-Detection Codes

- Redundancy (e.g. extra information), in the form of extra bits, can be incorporated into binary code words to detect and correct errors.
- A simple form of redundancy is parity, an extra bit appended onto the code word to make the number of 1's odd or even.
  - Parity can detect all single-bit errors and some multiple-bit errors.
  - A code word has even parity if the number of 1's in the code word is even.
  - A code word has odd parity if the number of 1's in the code word is odd.

4-Bit Parity Code Example

- Fill in the even and odd parity bits:

Even Parity Message_Parity	**Odd Parity** Message_Parity
000_	000_
001_	001_
010_	010_
011_	011_
100_	100_
101_	101_
110_	110_
111_	111_

- The codeword "1111" has even parity and the codeword "1110" has odd parity.
  - Both can be used to represent 3-bit data.

Alphanumeric Codes

- Arbitrary choice of bits to represent characters
  - Consistency:
    - input and output device must recognize same code
- Value of binary number representing character corresponds to placement in the alphabet
  - Facilitates sorting and searching
- Representing Characters
  - ASCII:
    - most widely used coding scheme
  - EBCDIC:
    - IBM mainframe (legacy)
  - Unicode:
    - developed for worldwide use

ASCII Character Codes

- American Standard Code for Information Interchange
  - ASCII
- This code is a popular code used to represent information sent as character-based data.
- It uses 7- bits to represent
  - 94 Graphic printing characters
  - 34 Non-printing characters
- Some non-printing characters are used for text format
  - e.g. BS = Backspace, CR = carriage return
- Other non-printing characters are used for record marking and flow control
  - e.g. STX = start text areas, ETX = end text areas.

ASCII Properties

- **ASCII has some interesting properties:**

- Digits 0 to 9 span Hexadecimal values 30_{16} to 39_{16}
- Upper case A-Z span 41_{16} to $5A_{16}$
- Lower case a-z span 61_{16} to $7A_{16}$
  - Lower to upper case translation (and vice versa) occurs by flipping bit 6
- Delete (DEL) is all bits set,
  - a carryover from when punched paper tape was used to store messages

ASCII Reference Table

MSD LSD	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P		p
1	SOH	DC1	!	1	A	Q	a	W
2	STX	DC2	“	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACJ	SYN	&	6	F	V	f	v
7	BEL	ETB	‘	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

74_{16}

111 0100

EBCDIC

- Extended Binary Coded Decimal Interchange Code developed by IBM
  - Restricted mainly to IBM or IBM compatible mainframes
  - Conversion software to/from ASCII available
  - Common in archival data
  - Character codes differ from ASCII

	ASCII	EBCDIC
Space	20_{16}	40_{16}
A	41_{16}	$C1_{16}$
b	62_{16}	82_{16}

UNICODE

- extends ASCII to 65,536 universal characters codes
- Available in many modern applications
- 2 byte (16-bit) code words

- ASCII Latin-I subset of Unicode
  - Values 0 to 255 in Unicode table
- Multilingual: defines codes for
  - Nearly every character-based alphabet
  - Large set of ideographs for Chinese, Japanese and Korean
  - Composite characters for vowels and syllabic clusters required by some languages
- Allows software modifications for local-languages

Unicode Assignment Table

Code range (in hexadecimal)	
0000–	0000–00FF Latin-I (ASCII)
1000–	General character alphabets: Latin, Cyrillic, Greek, Hebrew, Arabic, Thai, etc.
2000–	Symbols and dingbats: punctuation, math, technical, geometric shapes, etc.
3000–	3000–33FF Miscellaneous punctuations, symbols, and phonetics for Chinese, Japanese, and Korean
4000–	Unassigned
5000–	
•	
•	4E00–9FFF Chinese, Japanese, Korean ideographs
•	
A000–	Unassigned
B000–	
C000–	AC00–D7AF Korean Hangui syllables
D000–	
E000–	Space for surrogates
F000–	E000–F8FF Private use
FC00–	FC00–FFFF Various special characters

2 Classes of Codes

- Printing characters
  - Produced on the screen or printer
- Control characters
  - Control position of output on screen or printer
    - VT: vertical tab   LF: Line feed
  - Cause action to occur
    - BEL: bell rings   DEL: delete current character
  - Communicate status between computer and I/O device
    - ESC: provides extensions by changing the meaning of a specified number of contiguous following characters

Control Code Definitions

NUL	(Null) No character; used to fill space
SOH	(Start of Heading) Indicates start of a header used during transmission
STX	(Start of Text) Indicates start of text during transmission
ETX	(End of Text) Similar to above
EOT	(End of Transmission)
ENQ	(Enquiry) A request for response from a remote station; the response is usually an identification
ACK	(Acknowledge) A character sent by a receiving device as an affirmative response to a query by a sender
BEL	(Bell) Rings a bell
BS	(Backspace)
HT	(Horizontal Tab)
LF	(Line Feed)
VT	(Vertical Tab)
FF	(Form Feed) Moves cursor to the starting position of the next page, form, or screen
CR	(Carriage return)
SO	(Shift Out) Shift to an alternative character set until SI is encountered
SI	(Shift In) see above
DLE	(Data Link Escape) Similar to escape, but used to change meaning of data control characters; used to permit sending of data characters with any bit combination
DC1,DC2, DC3, DC4	(Device Controls) Used for the control of devices or special terminal features
NAK	(Negative Acknowledgment) Opposite of ACK
SYN	(Synchronous) Used to synchronize a synchronous transmission system
STB	(End of Transmission Block) Indicates end of a block of transmitted data
CAN	(Cancel) Cancel previous data
EM	(End of Medium) Indicates the physical end of a medium such as tape
SUB	(Substitute) Substitute a character for one sent in error
ESC	(Escape) Provides extensions to the code by changing the meaning of a specified number of contiguous following characters
FS, GS, RS, US	(File, group, record, and united separators) Used in optional way by systems to provide separations within a data set
DEL	(Delete) Delete current character

Keyboard Input

- Scan code
  - Two different scan codes on keyboard
    - One generated when key is struck and another when key is released
  - Converted to Unicode, ASCII or EBCDIC by software in terminal or PC
- Advantage
  - Easily adapted to different languages or keyboard layout
  - Separate scan codes for key press/release for multiple key combinations
    - Examples: shift and control keys

Other Alphanumeric Input

- OCR (optical character reader)
  - Scans text and inputs it as character data
  - Used to read specially encoded characters
    - Example: magnetically printed check numbers
  - General use limited by high error rate
- Bar Code Readers
  - Used in applications that require fast, accurate and repetitive input with minimal employee training
    - Examples: supermarket checkout counters and inventory control
  - Alphanumeric data in bar code read optically using wand
- Magnetic stripe reader:
  - alphanumeric data from credit cards
- Voice
  - Digitized audio recording common but conversion to alphanumeric data difficult
    - Requires knowledge of sound patterns in a language (phonemes) plus rules for pronunciation, grammar, and syntax

Image Data

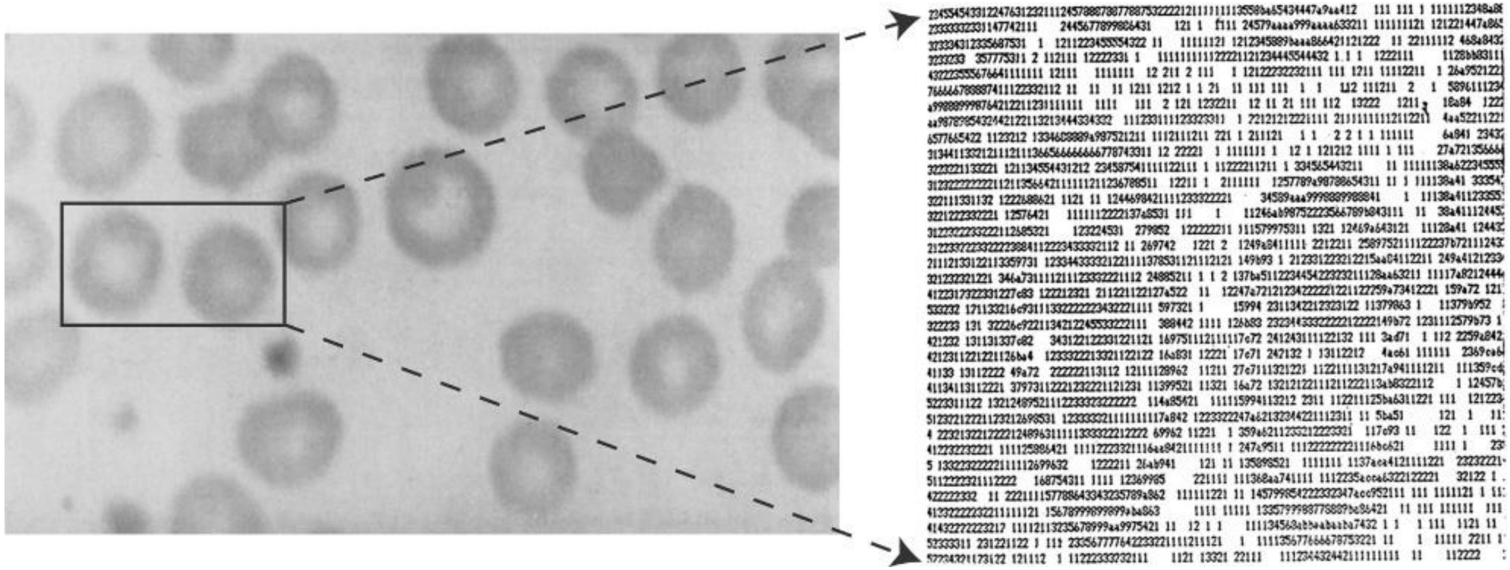
- Photographs, figures, icons, drawings, charts and graphs
- Two approaches:
  - Bitmap or raster images of photos and paintings with continuous variation
  - Object or vector images composed of graphical objects like lines and curves defined geometrically
- Differences include:
  - Quality of the image
  - Storage space required
  - Time to transmit
  - Ease of modification

Bitmap Images

- Used for realistic images with continuous variations in shading, color, shape and texture
  - Examples:
    - Scanned photos
    - Clip art generated by a paint program
- Preferred when image contains large amount of detail and processing requirements are fairly simple
- Input devices:
  - Scanners
  - Digital cameras and video capture devices
  - Graphical input devices like mice and pens
- Managed by photo editing software or paint software
  - Editing tools to make tedious bit by bit process easier

Bitmap Images

- Each individual pixel (pi(x)cture element) in a graphic stored as a binary number
  - –Pixel:
    - A small area with associated coordinate location
  - –Example:
    - each point below represented by a 4-bit code corresponding to 1 of 16 shades of gray



Bitmap Display

- Monochrome:
  - –black or white
    - 1 bit per pixel
- Gray scale:
  - –black, white or 254 shades of gray
    - 1 byte per pixel
- Color graphics:
  - –16 colors, 256 colors, or 24-bit true color (16.7 million colors)
    - 4, 8, and 24 bits respectively

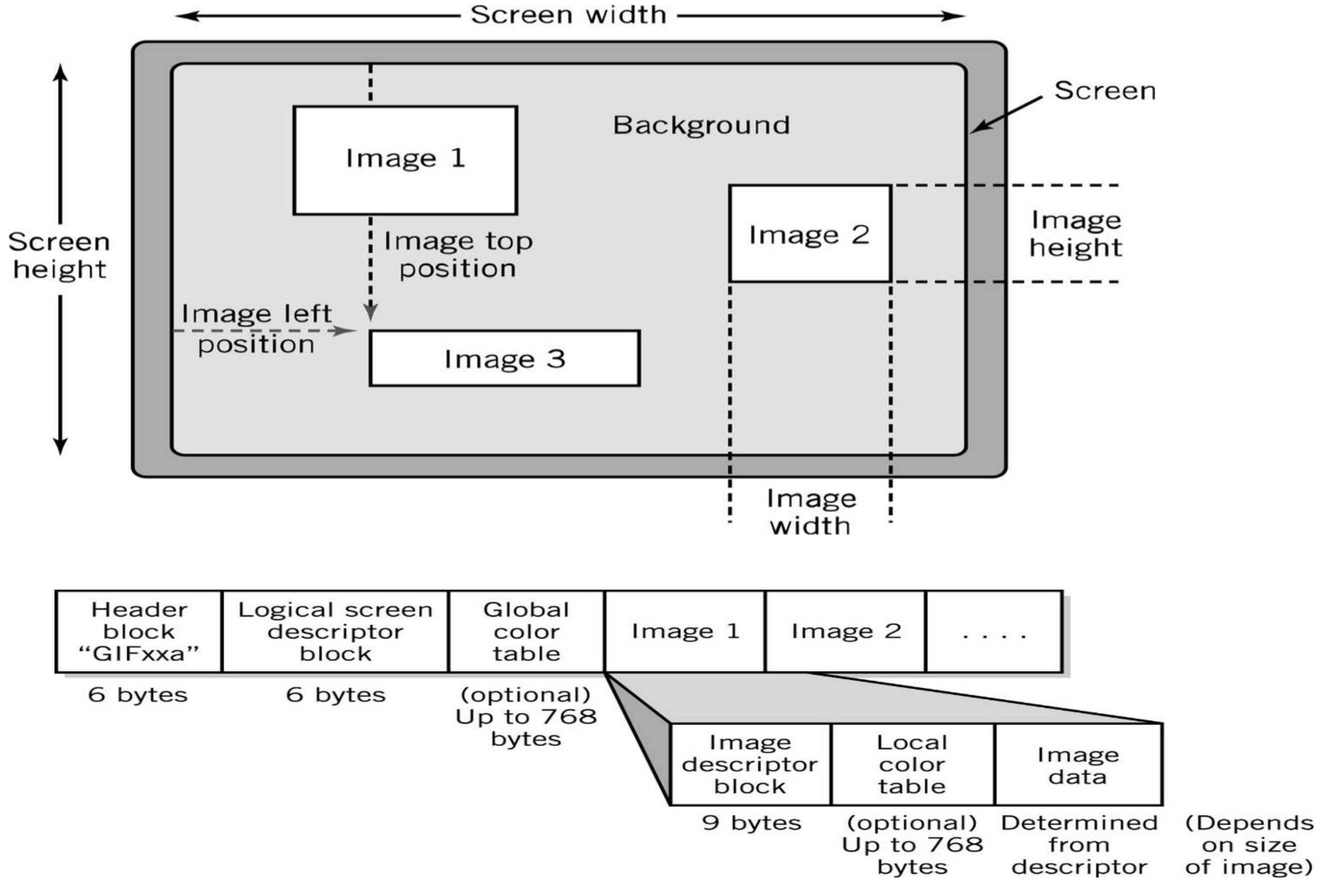
Storing Bitmap Images

- Frequently large files
  - Example:
    - 600 rows of 800 pixels with 1 byte for each of 3 colors ➔ ~1.5 MB file
- File size affected by
  - Resolution:
    - The number of pixels per inch
    - Amount of detail affecting clarity and sharpness of an image
  - Levels:
    - Number of bits for displaying shades of gray or multiple colors
  - Palette:
    - Color translation table that uses a code for each pixel rather than actual color value
- Data compression

GIF (Graphics Interchange Format)

- First developed by CompuServe in 1987
- GIF89a enabled animated images
  - allows images to be displayed sequentially at fixed time sequences
- Color limitation: 256
- Image compressed by LZW (Lempel-Zif-Welch) algorithm
- Preferred for line drawings, clip art and pictures with large blocks of solid color
- Lossless compression

GIF (Graphics Interchange Format)



JPEG (Joint Photographers Expert Group)

- Allows more than 16 million colors
- Suitable for highly detailed photographs and paintings
- Employs lossy compression algorithm that
  - Discards data to decreases file size and transmission speed
  - May reduce image resolution, tends to distort sharp lines

Other Bitmap Formats

- TIFF (Tagged Image File Format): .tif (pronounced tif)
  - –Used in high-quality image processing, particularly in publishing
- BMP (BitMaPped): .bmp (pronounced dot bmp)
  - –Device-independent format for Microsoft Windows environment:
    - pixel colors stored independent of output device
- PCX: .pcx (pronounced dot p c x)
  - –Windows Paintbrush software
- PNG: (Portable Network Graphics): .png (pronounced ping)
  - –Designed to replace GIF and JPEG for Internet applications
  - –Patent-free
  - –Improved lossless compression
  - –No animation support

Object Images

- Created by drawing packages or output from spreadsheet data graphs
- Composed of lines and shapes in various colors
- Computer translates geometric formulas to create the graphic
- Storage space depends on image complexity
  - number of instructions to create lines, shapes, fill patterns
    - Movies Shrek and Toy Story use object images
- Based on mathematical formulas
  - Easy to move, scale and rotate without losing shape and identity as bitmap images may
- Require less storage space than bitmap images
- Cannot represent photos or paintings
- Cannot be displayed or printed directly
  - Must be converted to bitmap since output devices except plotters are bitmap

Popular Object Graphics Software

- Most object image formats are proprietary
  - Files extensions include .wmf, .dxf, .mgx, and .cgm
- Macromedia Flash:
  - low-bandwidth animation
- Micrographx Designer:
  - technical drawings to illustrate products
- CorelDraw:
  - vector illustration, layout, bitmap creation, image-editing, painting and animation software
- Autodesk AutoCAD:
  - for architects, engineers, drafters, and design-related professionals
- W3C SVG (Scalable Vector Graphics) based on XML Web description language
  - Not proprietary

PostScript

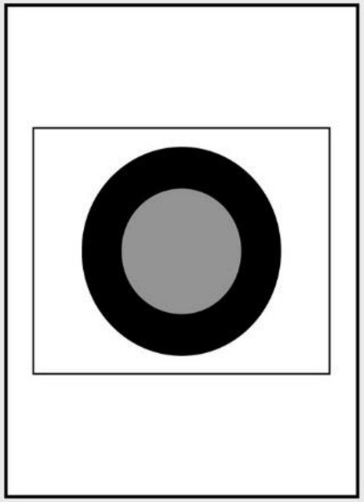
- Page description language:
  - –list of procedures and statements that describe each of the objects to be printed on a page
    - Stored in ASCII or Unicode text file
    - Interpreter program in computer or output device reads PostScript to generate image
  - –Scalable font support
    - Font outline objects specified like other objects

```
288 396 translate      % move origin to center of page
0 0 144 0 360 arc      % define 2" radius black circle
fill

0.5 setgray            % define 1" radius gray circle
0 0 72 0 360 arc
fill

0 setgray              % reset color to black
-216 -180 moveto       % start at lower left corner
0 360 rmoveto          % and define rectangle
432 0 rmoveto          % ...one line at a time
0 -360 rmoveto
closepath              % completes rectangle
stroke                 % draw outline instead of fill

showpage               % produce the image
```



Representing Characters

- Characters stored in format like Unicode or ASCII
  - Text processed and stored primarily for content
- Presentation requirements like font stored with the character
  - Text appearance is primary factor
    - Example: screen fonts in Windows
- Glyphs:
  - Macintosh coding scheme that includes both identification and presentation requirement for characters

Bitmap vs. Object Images

Bitmap (Raster)	Object (Vector)
Pixel map	Geometrically defined shapes
Photographic quality	Complex drawings
Paint software	Drawing software
Larger storage requirements	Higher computational requirements
Enlarging images produces jagged edges	Objects scale smoothly
Resolution of output limited by resolution of image	Resolution of output limited by output device

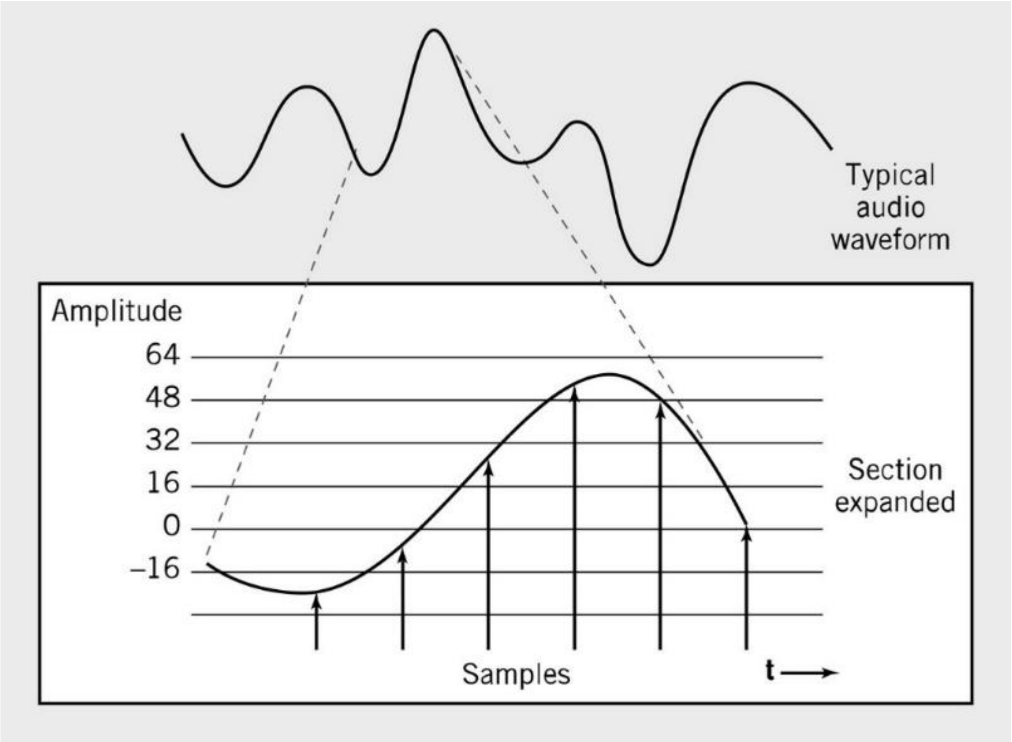
Bitmap vs. Object Images

Video Images

- Require massive amount of data
  - Video camera producing full screen 640x480 pixel true color image at 30 frames/sec ➔ 27.65 MB of data/sec
  - 1-minute film clip ➔ 1.6 GB storage
- Options for reducing file size:
  - decrease size of image,
  - limit number of colors,
  - reduce frame rate
- Method depends on how video delivered to users
  - Streaming video:
    - video displayed as it is downloaded from the Web server
      - Example: video conferencing
  - Local data (file on DVD or downloaded onto system) for higher quality
    - MPEG-2: movie quality images with high compression require substantial processing capability

Audio Data

- Transmission and processing requirements less demanding than those for video
- Waveform audio:
  - –digital representation of sound
    - Audio CD sampling rate = 44.1KHz
    - Height of each sample saved as:
      - –8-bit number for radio-quality recordings
      - –16-bit number for high-fidelity recordings
        - 2 x 16-bits for stereo

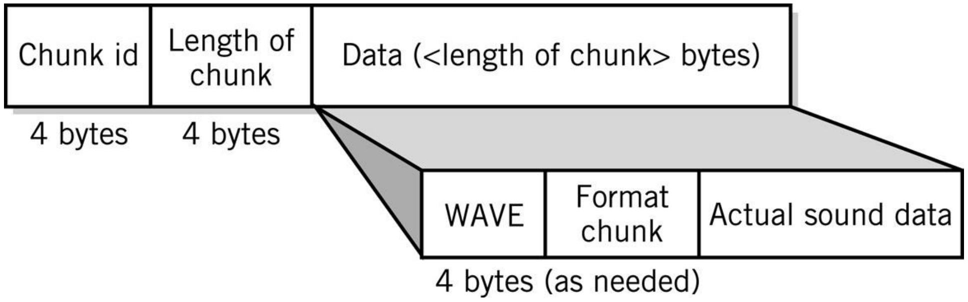


MIDI

- MIDI (Musical Instrument Digital Interface):
  - –instructions to recreate or synthesize sounds
    - Analog sound converted to digital values by A-to-D converter
  - –Music notation system that allows computers to communicate with music synthesizers
  - –Instructions that MIDI instruments and MIDI sound cards use to recreate or synthesize sounds.
    - Do not store or recreate speaking or singing voices
    - More compact than waveform
      - –3 minutes = 10 KB

Audio Formats

- MP3
  - Derivative of MPEG-2 (ISO Moving Picture Experts Group)
  - Uses psychoacoustic compression techniques to reduce storage requirements
  - Discards sounds outside human hearing range: lossy compression
- WAV
  - Developed by Microsoft as part of its multimedia specification
  - General-purpose format for storing and reproducing small snippets of sound



Data Compression

- Compression: recoding data so that it requires fewer bytes of storage space.
- Compression ratio: the amount file is shrunk
- Lossless: inverse algorithm restores data to exact original form
  - Examples: GIF, PCX, TIFF
- Lossy: trades off data degradation for file size and download speed
  - Much higher compression ratios, often 10 to 1
    - Example: JPEG
  - Common in multimedia
- MPEG-2:
  - uses both forms for ratios of 100:1

Compression Algorithms

- Repetition
  - **0** 5 8 7 **0 0 0 0** 3 4 **0 0 0** ➔ **0 1** 5 8 7 0 4 3 4 0 3
  - Example: large blocks of the same color
- Pattern Substitution
  - Scans data for patterns
  - Substitutes new pattern, makes dictionary entry

⌘	Pe	✵	pi	❖	ed
✷	er	◎	ck	⌗	pe
✪	Pi				

- Example: 45 to 30 bytes plus dictionary
  - Peter Piper picked a peck of pickled peppers.
  - ⌘ t ✷ ✪ p ✷ ✵◎ ❖ a ⌗ ◎ of ✵ ◎l ❖ ⌗ **pp** ✷ s.

Warning: Conversion or Coding?

- Do NOT mix up "conversion of a decimal number to a binary number" with "coding a decimal number with a binary code".
- $13_{10} = 1101_2$
  - This is conversion
- 13 $\Leftrightarrow$ 0001 0011_{BCD}
  - This is coding

Another use for bits: Logic

- Beyond numbers
  - *logical variables* can be *true* or *false*, *on* or *off*, etc., and so are readily represented by the binary system.
  - A logical variable A can take the values *false = 0* or *true = 1* only.
  - The manipulation of logical variables is known as Boolean Algebra, and has its own set of operations
    - which are not to be confused with the arithmetical operations.
  - Some basic operations: NOT, AND, OR, XOR

Basic Logic Operations

- Truth Tables of Basic Operations

NOT

A	A'
0	1
1	0

AND

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

- Equivalent Notations
  - not A = A' = $\overline{A}$
  - A and B = A.B = A$\wedge$B = A intersection B
  - A or B = A+B = A$\vee$B = A union B

More Logic Operations

XOR

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

XNOR

A	B	(A⊕B)'
0	0	1
0	1	0
1	0	0
1	1	1

– Exclusive OR (XOR):
  - either A or B is 1, not both

– $A \oplus B = A.B' + A'.B$

Any Questions?