FE Review

Computer Terms

2/2/2011 1

Computers sizes

- Supercomputers
- Mainframe – centralized, large memory, large peripherals
- Mini-computers – Less memory, smaller
- Microcomputers – smaller still few peripherals, slower than all, can be used as standalone devices

2/2/2011 2

Architecture

- Input
- Output
- Memory
- CPU

2/2/2011 3

The Microprocessor

- CPU on a single chip
  - » Control Unit
  - » Logic Unit
  - » Accumulators
  - » Specialized and non-specialized registers

2/2/2011 4

Special registers

- Accumulators: Hold data and instructions while they are being worked on and for future use. The accumulator is a special case register. More is better!
- Program Counter (PC): Special register which holds the address of the NEXT instruction to be executed

2/2/2011 5

Special Registers (continued)

- Stacks: temporary storage of data or address in sequential order
  - FIFO => First in First Out
  - LIFO =>Last in First Out
  - Sometimes called a "Push-down register"
- Flag Register: Holds individual indicators that specific actions have taken place (usually within the accumulators)
- Instruction Register (IR): holds current instruction being executed.

2/2/2011 6

Base 10 number system reviewed

radix pt.

$\underbrace{3}_{10^2}\ \underbrace{4}_{10^1}\ \underbrace{2}_{10^0}\ .\ \underbrace{1}_{10^{-1}}\ \underbrace{5}_{10^{-2}}$

We can represent this number by the polynomial

$$(3*10^2)+(4*10^1)+(2*10^0)+(1*10^{-1})+(5*10^{-2}) = \boxed{342.15_{10}}$$

2/2/2011 7

Base 2 (binary system)

Computers use a number system known as binary which is a base 2 system. Since it is a base 2 system, the highest number in the system is a 1.

$\underbrace{1}_{2^4}\ 0\ 0\ \underbrace{1}_{2^1}\ \underbrace{1}_{2^0}$

$$(1*2^4)+(1*2^1)+(1*2^0)$$

$$16+2+1=19$$

2/2/2011 8

Base 10 to Base 2 Conversion

Convert 42_{10} to Binary (Base 2)

powers of two ⇒ need 42 remainder ⇒

$\overset{32}{\underset{42}{1}}\ \overset{16}{\underset{10}{0}}\ \overset{8}{\underset{10}{1}}\ \overset{4}{\underset{2}{0}}\ \overset{2}{\underset{2}{1}}\ \overset{1}{\underset{0}{0}}{}_2 = 42_{10}$

bit	8	7	6	5	4	3	2	1	0
#	273	17	17	17	17	1	1	1	1
power	256	128	64	32	16	8	4	2	1
273	1	0	0	0	1	0	0	0	1_2

2/2/2011 9

Base 8 (octal system)

Convert 1750_8 to $base_{10}$

1	*	8^3	=	1 * 512		512
7	*	8^2	=	7 * 64		448
5	*	8^1	=	5 * 8		40
0	*	8^0	=	0 * 1		0
						$1,000_{10}$

2/2/2011 10

Base 10 to Octal conversion

800	288	32	0	.
512	64	8	1	.
1	4	4	0	.

2/2/2011 11

Binary to Octal Conversion

Convert 10110111_2 to Octal

Start from the radix point and group into groups of three. Feel free to pad the left most group to make a group of three.

0 1 0 | 1 1 0 | $1\ 1\ 1_2$. ⇒ 267_8

2 | 6 | 7

2/2/2011 12

Octal to Binary Conversion

Convert 52_8 to Binary

$\underbrace{5}_{101}\ \underbrace{2}_{010}{}_8 \Rightarrow 101010_2$

Base 16 (hex system)

Convert $AD85_{16}$ to $base_{10}$

A	*	16^3	=	10	*	4,096	40,960
D	*	16^2	=	13	*	256	3,328
8	*	16^1	=	8	*	16	128
5	*	16^0	=	5	*	1	5
							$44,421_{10}$

Base 10 to Base 16 Conversion

Convert 2748 to Hex.

As with the other systems, we will use the "Sum of Weights" method.

256	16	1
A	B	C
2748	188	12

So the answer is ABC_{16} ⇐

Binary to Base 16 Conversion

Convert 101101_2 to Base 16

$$101101 = \underbrace{10}_{2}\,\underbrace{1101}_{D}{}_2 \Rightarrow 2D_{16}$$

Base 16 to Binary Conversion

Convert $5AC_{16} \Rightarrow$ Base 2

$$\underbrace{5}_{0101}\;\underbrace{A}_{1010}\;\underbrace{C}_{1100}{}_{16} \Rightarrow 10110101100_2$$

Data Sizes

The smallest changeable unit of data is known as the BIT.

The letters stand for Binary digiT.

$$1\;0\;0\;1_2 = 9_{10} \Rightarrow 4 \text{ bits} = 1 \text{ nibble}$$

The next data size up is the Nibble which is 4 bits long.

Data Sizes (continued)

- Byte = 8 Bits
- Word = 16 Bits = 2 Bytes
- Long Word = 32 Bits = 4 bytes
- Double Word = 64 Bits = 4 Words

2/2/2011 19

Data Types

- Character data – ASCII
  - American Standard Code for Information Interchange
- Numerical data – Binary
  - Commonly used by mini- and micro- computers
  - Allows a maximum of 128 different characters.
- EBCDIC
  - Extended Binary Coded Decimal Interchange Code
  - Used extensively by mainframes
  - Allows a maximum of 256 characters

2/2/2011 20

Parity System

This is an extremely simple way of making sure that single errors in data interchange does not occur. There are two basic types of parity:

Even or Odd.

A bit position is reserved somewhere in the data Word (or larger data structure). Then the number of 1's in the word are counted. If Even Parity is desired, then the total number of ones Needs to be EVEN. If it isn't, then the Parity bit is set to a 1 which makes the total even. Otherwise, it is set to a 0. The same process applies to Odd parity.

2/2/2011 21

Memory Size

Memory size uses prefixes such as K, M, and G

1k x 8 of ram is 2^{10} x 8 bits wide of data.
2^{10} x 8 = 1,024 * 8 = 8,192 bits. Note that
The K here did not mean 1000 exactly. Nor
Will a Gig (G) actually mean 10^9 exactly.

2/2/2011 22

Memory Types

- Volatile memory (Lost when power is lost)
  - Random Access Memory (RAM)
- Non-volatile memory
  - Read Only Memory (ROM)
  - Programmable ROM (PROM)
  - Erasable PROM (EPROM)
  - Hard Drives, CD/DVD ROMS, Floppies, etc.

2/2/2011 23

Example Question

Which of the following types of memory
is lost when a computer's power is interrupted?

(A)	RAM
(B)	ROM
(C)	PROM
(D)	EPROM

2/2/2011 24

Bus'

- Address Bus – Carries address locations
- Control Bus – Transfers control and and status information
- Data Bus – Carries the actual data
  - Series or parallel

2/2/2011 25

Example Question

A 256k-word memory uses 16 bit-words. How many parallel data lines are required to pass data to the CPU for processing?

(A)	2
(B)	8
(C)	9
(D)	16

The key word here is "Parallel". All bits in a word are passed in parallel to the CPU. There is one data line per bit in the word.

2/2/2011 26

Command Sets

- Complex Instruction Set (CISC)
- Reduced Instruction Set (RISC)
  - Used to increase speed. Fewer, but more powerful instructions.

2/2/2011 27

Operating System

Sometimes referred to as the OS, this manages everything that the computer does with regards to memory, processor operation scheduling, access to/form peripherals, I/O, resolves any conflicts with resources. The OS is often also referred to as the BIOS or "Basic input/output System".

2/2/2011 28

Multi-Tasking

Main memory is allocated between several users with different applications running at the same time. It is also known as Multi-Programming. If the term USER is literally different people, then it can also be called a Multi-User system.

2/2/2011 29

Time Sharing

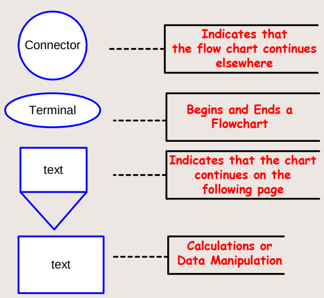
Also known as swapping, (not that type of swapping!!. Lets keep this out of the gutter, OK?!!). This is a technique where each user takes a turn using the OS for a specific period of time, (less than one second). At the end of the users time, the active memory is stored in a private area (assigned to the user), and then the next users private memory area is loaded into active memory.

2/2/2011 30

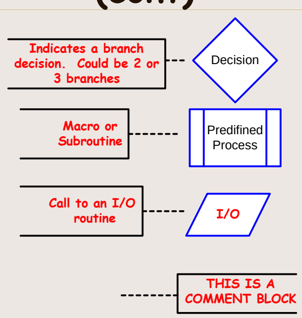
Run Modes

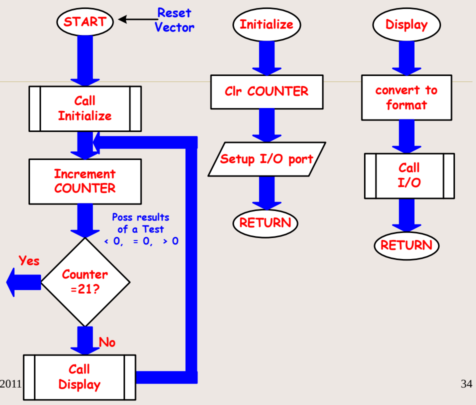
- Batch Mode
- Real-time mode (also known as interactive mode.

Structured Programming



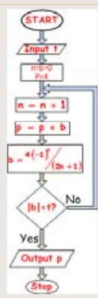
Structured Programming (cont)





2/2/2011 34

Example Question



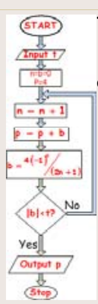
The variable t in the flowchart below has an initial value of 0.5. What is the value of the variable p at the conclusion of the routine?

(A)	2.9
(B)	3.334
(C)	4.0
(D)	4.44

iteration	n	p	b	| b |< t ?
1	1	4.00	$-4/3$	no
2	2	2.67	$4/5$	no
3	3	3.47	$-4/7$	no
4	4	2.9	$4/9$	yes

2/2/2011 35

Example Question



The key here is to come up with some equations and then complete a grid!

$$p = p + b_{n-1}$$

$$b_n = \frac{4(-1)^n}{2n + 1}$$

(A)	2.9
(B)	3.334
(C)	4.0
(D)	4.44

iteration	n	p	b	| b |< t ?
1	1	4.00	$-4/3$	no
2	2	2.67	$4/5$	no
3	3	3.47	$-4/7$	no
4	4	2.9	$4/9$	yes

2/2/2011 36

Algorithms

Language independent

IF-THEN-single statement

Algorithm	IF – condition – THEN
	single statement
	END IF

Fortran	IF (condition) statement

IF-THEN-multiple statements

Algorithm	IF – condition – THEN
	Multiple statements
	END IF

Fortran	IF (not – condition) GO TO 10 statements 10 CONTINUE

IF-THEN-ELSE

Algorithm

```
IF – condition – THEN
  statements
ELSE
  statements
END IF
```

Fortran

```
  IF (not – condition) GO TO 5
     statements
   GO TO 6
5  CONTINUE
     statements
6  CONTINUE
```


DO WHILE

Algorithm

```
DO WHILE – condition
  statements
END DO
```

Fortran

```
5  IF (not – condition) GO TO 6
     statements
   GO TO 5
6  CONTINUE
```


DO UNTIL

Algorithm

```
DO UNTIL – condition
  statements
END DO
```

Fortran

```
5  CONTINUE
     statements
   IF (not – condition) GO TO 5
```

DO FOR

Algorithm

```
DO FOR I ← L TO M BY N
    statements
END DO
```

Fortran

```
      DO 10 I = L, M, N
          statements
   10 CONTINUE
```

(, N can be omitted in which case , 1 is assumed)

2/2/2011 43

Multi-way selection ELSE-IF Algorithm

```
IF condition(1) THEN
    statements
ELSE IF condition(2) THEN
    statements
ELSE IF condition(3) THEN
    statements
ELSE condition(4)
    statements
END IF
```

2/2/2011 44

Multi-way selection ELSE-IF Fortran

```
    IF (not – condition(1)) GO TO 10
       statements
    GO TO 20
10  IF (not – condition(2)) GO TO 11
       statements
    GO TO 20
11  IF (not – condition(3)) GO TO 12
       statements
    GO TO 20
12  CONTINUE
       statements
20  CONTINUE
```

2/2/2011 45

Multi-way selection CASE Algorithm

```
CASE variable OF
1:
    statements
2:
    statements
3:
    statements
ELSE
    statements
END CASE
```

ELSE is optional

Multi-way selection CASE Fortran

```
IF (variable .LT. 1) GO TO 20
IF (variable .GT. 3) GO TO 20
GO TO (11,12,13), variable
11  CONTINUE
       statements
    GO TO 30
12  CONTINUE
       statements
    GO TO 30
13  CONTINUE
       statements
    GO TO 30
30  CONTINUE
```


Function

Algorithm

```
FUNCTION function-name (param-1,...,param-n)
    statements
RETURN expression
END function-name
```

Fortran

```
data type FUNCTION function-name (param-1,...,param-n)
    statements
    function-name = expression
    RETURN
END
```

Subroutine

Algorithm

SUBROUTINE subroutine-name (param-1,...,param-n)

statements

RETURN

END subroutine-name

Fortran

SUBROUTINE subroutine-name (param-1,...,param-n)

statements

RETURN

END

2/2/2011 49

Global Reference

Algorithm

GLOBAL REFERENCE

GLOBAL variable (1), ... , variable (n)

Fortran

COMMON variable (1), ... , variable (n)

2/2/2011 50