UMS
UNIVERSITI MALAYSIA SABAH

KE27203 COMPUTER ARCHITECTURE & MICROPROCESSOR

TUTORIAL 1: COMPUTER ARCHITECTURE

NO MATRIC: BK20110197

STUDENT NAME: FAUZAN AHMAD BIN GOFRAN

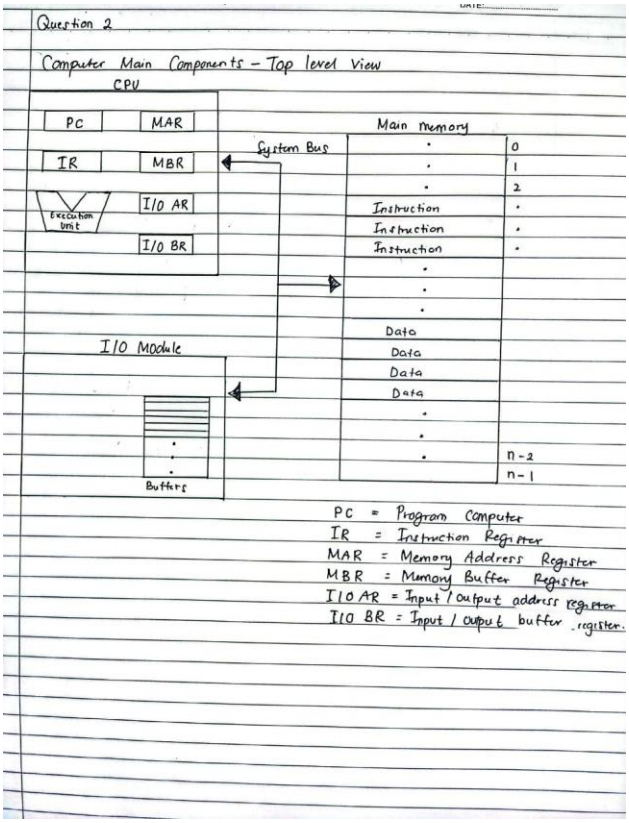
1. Explain what is the stored program concept?

Answer: A computer that stores instructions in its memory to enable it to perform a variety of tasks in sequence or intermittently.

2. Explain the function of main memory, CPU and input/output module for the operation of stored program concept and sketch the top-level view of computer main components.

Answer:

1. **Main memory that stores operating system software, software applications and other information for the central processing unit (CPU) to have fast and direct access when needed to perform tasks**.
2. **CPU or Central Processing Unit is computer hardware that carries out a computer's instructions and controls all the arithmetical, logical, and input/output operations of a computer system.**
3. **An input/output (I/O) device is a hardware device that can accept inputted, outputted or other processed data. It also can acquire respective media data as input sent to a computer or send computer data to storage media as storage output.**



3. By giving an example, demonstrate the steps taken during instruction fetch and instruction execute.
Answer:

Characteristics of a Hypothetical Machine

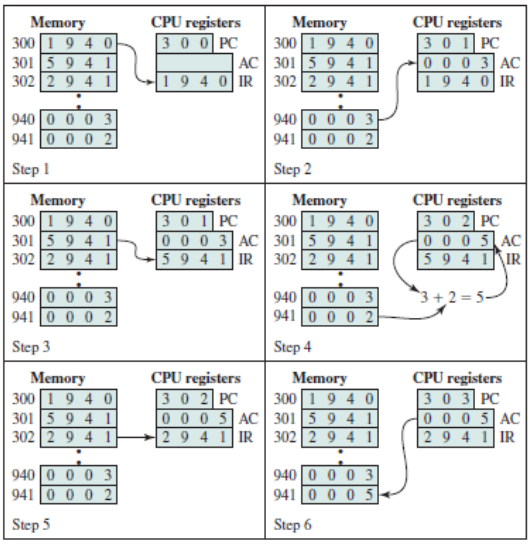


Figure 1 Based on COMPUTER ORGANIZATION AND ARCHITECTURE Design Performance Ninth Edition, William Stallings , Page 71 – 72.

1. The PC contains 300, the address of the first instruction. This instruction (the value 1940 in hexadecimal) is loaded into the instruction register IR, and the PC is incremented. Note that this process involves the use of a memory address register and a memory buffer register. For simplicity, these intermediate

   registers are ignored.
2. The first 4 bits (first hexadecimal digit) in the IR indicate that the AC is to be loaded. The remaining 12 bits (three hexadecimal digits) specify the address (940) from which data are to be loaded.
3. The next instruction (5941) is fetched from location 301, and the PC is incremented.
4. The old contents of the AC and the contents of location 941 are added, and the result is stored in the AC.
5. The next instruction (2941) is fetched from location 302, and the PC is incremented.
6. The contents of the AC are stored in location 941.

4. Sketch and explain the Instruction Cycle State diagram without Interrupts and with Interrupts for the actions taken by each of the state and distinguish the effectiveness between both cycles.
Answer:

1. Instruction Cycle State diagram without Interrupts.

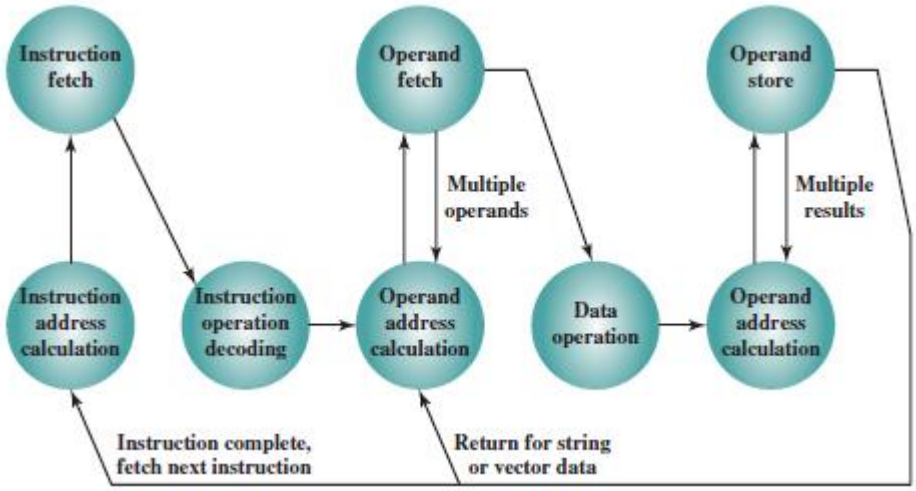


Figure 2 Based on COMPUTER ORGANIZATION AND ARCHITECTURE Design Performance Ninth Edition, William Stallings, Page 73

- **Instruction fetch(if):** Read instruction from its memory location into the processor.
- **Instruction operation decoding (iod):** Analyze instruction to determine type of operation to be performed and operand(s) to be used.
- **Operand address calculation (oac):** If the operation involves reference to an operand in memory or available via I/O, then determine the address of the operand.
- **Operand fetch (of):** Fetch the operand from memory or read it in from I/O.
- **Data operation (do):** Perform the operation indicated in the instruction.
- **Operand store (os):** Write the result into memory or out to I/O.

2. Instruction Cycle State diagram with Interrupts.

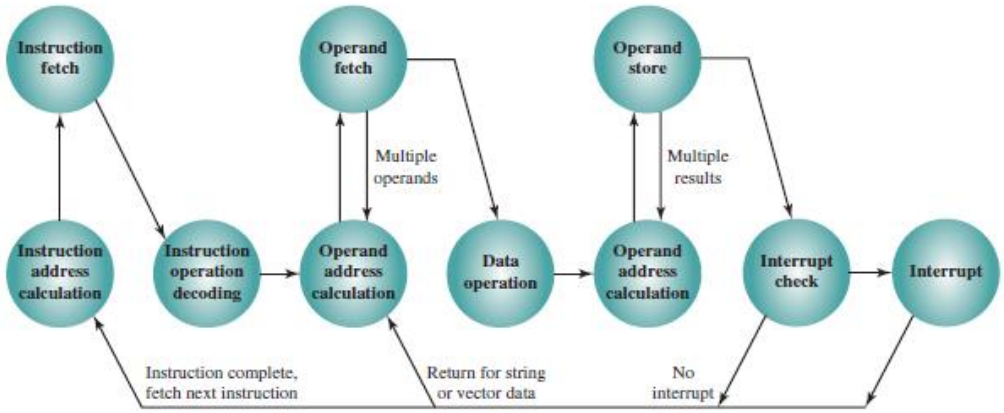


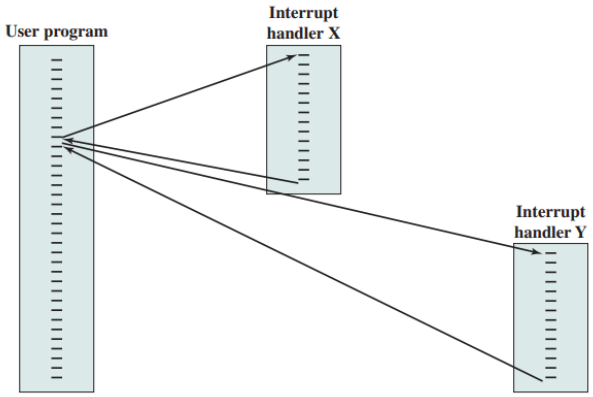
Figure 3Based on COMPUTER ORGANIZATION AND ARCHITECTURE Design Performance Ninth Edition, William Stallings, Page 81

- **Instruction address calculation (iac)**: Determine the address of the next instruction to be executed. Usually, this involves adding a fixed number to the address of the previous instruction. For example, if each instruction is 16 bits long and memory is organized into 16-bit words, then add 1 to the previous address.
- **Instruction fetch (if)**: Read instruction from its memory location into the processor.
- **Instruction operation decoding (iod)**: Analyze instruction to determine type of operation to be performed and operand to be used.
- **(Operand address calculation (oac)**: If the operation involves reference to an operation in memory or available via I/O, then determine the address of the operand.
- **Operand fetch (of)**: Fetch the operand from memory or read it in from I/O.
- **Data operation (dt)**: Perform the operation indicated in the instruction.
- **Operand store(os)**: Write the result into memory or out to I/O.
- **Fetch**: Read the next instruction from memory into the processor.
- **Execute**: Interpret the op-code and perform the indicated operation.
- **Interrupt**: If interrupts are enabled and an interrupt has occurred, save the current process state and service the interrupt.

5. Two ways of dealing with multiple interrupts are by using Sequential and Nested process, explain in detail both process in handling multiple interrupts using an appropriate diagram.

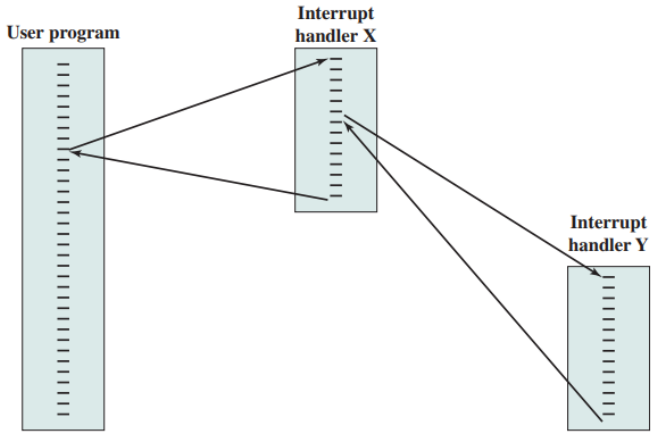
Answer:

- The Sequential Multiple Interrupts system operates by distributing interrupts in a tight order. Unless there is one interrupt that has priority, the interruptions are usually dealt with in the order in which they occurred, from the first to the last. An interrupt is a signal from a device that prompts a computer's main software to pause and figure out what it needs to do next. When an interrupt occurs, the task that we are presently working on is suspended and passed to another handler until the interrupt is resolved. When the handler is finished, the operation that was interrupted will resume with data loss.



(a) Sequential interrupt processing

- In a nested interrupt system, an interrupt is allowed to anytime and anywhere even an ISR is being executed. But only the highest priority ISR will be executed immediately. The second highest priority ISR will be executed after the highest one is completed.



(b) Nested interrupt processing

6. Give a list of an elements for designing any bus for processor and shortly explain each one of the elements in contributing towards bus design.

Answer:

- **Bus Types**

Bus lines can be reported into two generic types are dedicated and multiplexed. A dedicated bus line is permanently authorized either to one function or a physical subgroup of computer components. A multiplexed bus line is assigned too many functions based on some parameters.

- **Method of Arbitration**

In all but the simplest systems, more than one module can require control of the bus. Therefore, only one unit at a time can strongly transfer over the bus, some method of arbitration is required. Various methods can be classified as centralized or distributed. In a centralized scheme, a single hardware device defined as a bus controller or arbiter is important for assigning time on the bus. In a distributed scheme, there is no central controller. Each module includes access control logic and the modules help together to transfer the bus.

- **Timing**

Timing defines how events are integrated on the bus. With synchronous timing, the circumstances of events on the bus are persistent by a clock. The figure shows the timing diagram for a synchronous read operation. With asynchronous timing, the circumstances of one event on a bus follows and are based on the circumstances of a previous event. In this example, the CPU places address and read signals on the bus. After pausing for these signals to maintain, it declares an MSYN (master sync) signal. It is indicating the presence of valid current address and control signals. The memory module responds with data and an SSYN (slave sync) signal, indicating the response.

- **Bus Width**

The width of the data has an impact on system execution. The wider the data bus, the higher the number of bits moved at one time. The width of the address bus has an impact on system capacity, that is, the wider the address bus, the higher the dimension of locations that can be referenced.

- **Data Transfer Type**

A bus provides several data transfer types as shown in the figure. All buses provide both write (master to slave) and read (slave to master) assigns.

7. Explain Moore's Law and it’s important for future computer and processors design?
Answer:

In 1975, Gordon Moore, co-founder of Intel and Fairchild Semiconductor, predicted that the number of transistors in a dense integrated circuit (IC) would double every two years. He predicted that as transistors shrink to the nanoscale, allowing integrated circuits to be integrated and composed of more transistors, allowing for the development of more powerful computers, twice as many transistors would be created every two years, allowing for twice as many transistors to fit onto computer chips. Moore's Law was born as a result. However, advances in engineering and physics have progressed to the point where the bigger the quantity of power, the more the resources required to perform sophisticated activities via computers, putting a burden on the use of smaller transistors and forcing computer systems to approach their limit in transistor capacity and power. Hence, industry leaders are asserting that Moore’s Law has come to an end, and computers will no longer have many more transistors every year. Moore’s Law is simply an estimate predicting the speedy development of additional advanced technologies, and therefore the evolution of transistors. As an estimate, it helped to strategically pave the way for larger enterprises to set up for the implementation of systems that might have the benefit of more sturdy computer systems. The result of Moore’s Law has modified the way end-users and corporations have operated, with the rolling out of more powerful computers, video gaming devices, data centers and workstations leading to raised capacities and even the event of recent systems and apps

8. Explain the three types of computer bus and differentiate the function of each bus?
Answer:

A bus is a high-speed internal connection. Buses are used to send control signals and data between the processor and other components.

Three types of buses are used.

Address bus - carries memory addresses from the processor to other components such as primary storage and input/output devices. The address bus is unidirectional.

Data bus - carries the data between the processor and other components. The data bus is bidirectional.

Control bus - carries control signals from the processor to other components. The control bus also carries the clock's pulses. The control bus is unidirectional

9. Consider a hypothetical 32-bit microprocessor having 32-bit instructions composed of two fields: The first byte contains the opcode and the remainder the immediate operand or an operand address.

a. What is the maximum directly addressable memory capacity (in bytes)
Answer:

$2^{(32-8)} = 2^{24}$ = 16,777, 216 bytes = 16 MB

8 bits = 1 bytes for the opcode

b. Discuss the impact on the system speed if the processor bus has:

i. a 32-bit local address bus and a 16-bit local data bus or
Answer:
A 16-bit local data bus and a 32-bit local address bus Three bus cycles would be required for each instruction and data transmission, one for the address and two for the data. Since the address bus is 32 bits, the entire address can be transmitted to memory at once and decoded there; but, because the data bus is only 16 bits, fetching the 32-bit instruction or operand will take two bus cycles.

ii. a 16-bit local address bus and a 16-bit local data bus

Answer:

A 16-bit local data bus and a 16-bit local address bus Instruction and data transfers would each require four bus cycles, two for addresses and two for data. As a result, the processor will have to send the entire 32-bit address to memory in two transmissions; this will necessitate more complex memory interface control to latch the two halves of the address before performing an access to it. In addition to the two-step address problem, the microprocessor will need two bus cycles to fetch the 32-bit instruction or operand because the data bus is also 16 bits.

c. How many bits are needed for the Program Counter (PC) and the Instruction Register (IR)?
Answer:

Program Counter (PC) needs 24 bits (24-bit addresses), while Instruction Register (IR) needs 32 bits (32-bit addresses).

10. Compute zero-, one- , two- , and three-address machines by writing programs to compute : **X** = (A + B x C) / (D - E x F), for each of the four machines. The instructions available for use are as below:

0 Address	1 Address	2 Address	3 Address
PUSH MPOP M ADD SUB MUL DIV	LOAD M STORE M ADD M SUB M MUL M DIV M	MOVE (X ← Y) ADD (X ← X + Y) SUB (X ← X - Y) MUL (X ← X x Y) DIV (X ← X/Y)	MOVE (X ← Y) ADD (X ← Y + Z) SUB (X ← Y-Z) MUL (X ← Y x Z) DIV (X ← Y/Z)

Answer

Question 10

0 Address Machine

Input	Description	Arithmetic
PUSH A	Insert A	A
PUSH B	Insert B	B, A
PUSH C	Insert C	C, B, A
MUL	Multiply B with C	(C*B), A
ADD	Add (C*B) to A	A+(C*B)
PUSH D	Insert D	D, (A+C*B)
PUSH E	Insert E	E, D, (A+C*B)
PUSH F	Insert F	F, E, D, (A+C*B)
MUL	Multiply E with F	(E*F), D, (A+C*B)
SUB	Subtract (E*F) from D	D-(E*F), (A+C*B)
DIV	Divide (A+C*B) with (D-E*F),	(A+C*B)/ (D-E*F)
POP X	Display X with the answer	X= (A+C*B)/ (D-E*F)

1 Address Machine

Input	Description	Arithmetic
LOAD E	AC equal to E	AC<= E
MUL F	AC equal to previous AC multiply with F	AC<= AC*F (E*F)
STORE T	Store AC into T	T<= AC
LOAD D	AC equal to D	AC<= D
SUB T	AC equal to previous AC subtract with T	AC<= AC-T (D-E*F)
STORE T	Store AC into T	T<= AC
LOAD B	AC equal to B	AC<=B
MUL C	AC equal to previous AC multiply with C	AC<=AC*C (B*C)
ADD A	AC equal to previous AC add with A	AC<=AC+A (A+B*C)
DIV T	AC equal to previous AC divide with T	AC<=AC/T (A+B*C)/ (D-E*F)

STORE X	Store AC into X	X<=AC X=(A+B*C)/ (D-E*F)

2 Address Machine

Input	Description	Arithmetic
MOV R0, E	Store E into R0	R0<=E
MUL RO, F	Multiply R0 with F and store into R0	R0<=R0*F (E*F)
MOV R1, D	Store D into R1	R1<= D
SUB R1, R0	R1 subtract with R0 and store into R1	R1<= R1=R0 (D-E*F)
MOV R0,B	Store B into R0	R0<=B
MUL R0, C	Multiply R0 with C and store into R0	R0<=R0*C (B*C)
ADD R0, A	Add R0 with A and store into R0	R0<= A+ R0 (A+B*C)
DIV R0, R1	Divide R0 with R1 and store into R0	R0<=R0/R1 (A+B*C)/ (D-E*F)
MOV X, R0	Store R0 into X	X=R0 X=(A+B*C)/ (D-E*F)

3 Address Machine

Input	Description	Arithmetic
MUL R0, E, F	Multiply E with F and store into R0	R0<= E*F
SUB R0, D, R0	D subtract with R0 and store into R0	R0<= D-R0 (D-E*F)
MUL R1, B, C	Multiply B with C and store into R1	R1<= B*C
ADD R1, A, R1	Add A with R1 and store into R1	R1<=A+R1 (A+B*C)
DIV X, R1, R0	Divide R1 with R0 and store into X	X<=R1/R0 X=(A+B*C)/ (D-E*F)

11. All Intel based microprocessor manufacture used segmentation memory addressing ratherthan Flat addressing mode:

i. Differentiate the Flat and Segmentation addressing mode

Answer:

A flat addressing mode is a memory system in which there is no segmentation. The address of the first byte in the memory is at 00 0000 0000H and the last location is at FF FFFF FFFFH. The flat addressing mode does not use a segment register to address a location in the memory and the offset address is the actual physical address in 64-bit mode. This form of addressing is much easier to understand but offers little protection to the system. At the same time, this addressing mode consumes more resources compared to segmentation because it contain a lot of meaningless zeros in the 64 bit address. The data paths must be wider, and memories have to be bigger to hold all the redundant zeros

Segmentation addressing mode consist of a combination of a segment address and an offset address to access a memory location in the real mode. The segment address, located within one of the segment registers, defines the beginning address of any 64K-byte memory segment, while the offset address selects any location within the 64K byte memory segment. The offset is the distance above the start of the segment. Segments in the real mode always have a length of 64K bytes. Therefore, once the beginning address is known, the ending address is found by adding FFFFH. Segmented addresses allow a denser and more efficient use of all CPU resources compared to flat addressing mode

ii. In real mode, how many total memory that can be active at any given time

Answer:

In real mode, the total memory that can be active at any given time is only the first 1M byte of memory. The first 1M byte of memory is called the real memory, conventional memory, or DOS memory system. Real mode operation allows application software written for the 8086/8088, which only contains 1M byte of memory, to function in the 80286 and above without changing the software.

iii. Determine the Physical memory address of the following segmentation:

a. 23FB:FAB3H

Answer:

Segment Address: 23FBH

Offset Address: FAB3H

Physical Memory address:

$$\begin{array}{r} 23FB0H \\ +\ FAB3H \\ \hline 33A63H \end{array}$$

Physical Memory address: 33A63H

b. b. 1234: FFB3H

Answer:

Segment Address: 1234H

Offset Address: FFB3H

Physical Memory address:

$$\begin{array}{r} 12340H \\ +\ FFB3H \\ \hline 222F3H \end{array}$$

Physical Memory address: 222F3H

c. c. 23FB:89FAH

Answer:

Segment Address: 23FBH

Offset Address: 89FAH

Physical Memory address:

$$\begin{array}{r} 23FB0H \\ 89FAH \\ \hline 2C9AAH \end{array}$$

Physical Memory address: 2C9AAH

12. Give 3 examples (register name and its abbreviation with available number of bits can be used) of the following available general-purpose and special-purpose registers for the 80386 microprocessor and define the usage of the registers respectively:

i. Data Register

Answer

Used to store temporary data during execution and can be used as destination or source for any data movement. Data registers that are addressable on a byte or 16-bit basis, and four 16-bit pointer and index registers. The data registers can be used as general purpose in some instructions. In others, the registers are used implicitly. For example, a multiply instruction always uses the accumulator.

Data (DL)(8-bits),

Base Index(BL) (8-bits),

Accumulator(AL)(8-bits)

ii. Index and Pointer Register

Answer

Can be used either 16-bit or 32-bit registers and is used to hold the offset of the address of data in memory
Base pointer(EBP)(32-Bits),

Destination Index(DI)(16-bits),

Source Index(SI)(16-bits)

iii. Segment Register

Answer

Each segment registers hold the base memory address of 64-k memory segment.The segment registers of the 80386 give systems software designers the flexibility to choose among various models of memory organization. The 80386 architecture takes advantage of this by providing mechanisms to support direct access to the instructions and data of the current module's environment, with access to additional segments on demand. At any given instant, six segments of memory may be immediately accessible to an executing 80386 program. The segment registers CS, DS, SS, ES, FS, and GS are used to identify these six current segments. Each of these registers specifies a particular kind of segment, as characterized by the associated mnemonics ("code," "data," or "stack").

Stack Segment(SS)(16-bits),

Code Segment(CS)(16-bits),

Data Segment(DS)(16-bits)

iv. Flags Register

Answer

Status Flags change as a result of executing certain instruction.

Control Flags control operations of the 80386.

The flags register is a 32-bit register named EFLAGS. The flags control certain operations and indicate the status of the 80386.The low-order 16 bits of EFLAGS is named FLAGS and can be treated as a unit. This feature is useful when executing 8086 and 80286 code, because this part of EFLAGS is identical to the FLAGS register of the 8086 and the 80286. The flags may be considered in three groups: the status flags, the control flags, and the systems flags.

Carry(C)(32-bits)

Parity(P)(32-bits)

Zero(Z)(32-bits)