COS2621 - Computer Organization

Review Questions 2013

Computer Organization and Architecture - *Designing for Performance*
8th Edition - William Stallings

v1.00 April 2013

Ron Barnard

Chapter 1 - Introduction

Question 1.1

What, in general terms, is the distinction between computer organization and computer architecture ?

Computer architecture - refers to those attributes of a system visible to a programmer, ie: those attributes that have a direct impact on the logical execution of a program. Eg: instruction set, the number of bits used to represent various data types, I/O mechanisms, and techniques for addressing memory.

Computer organization - refers to the operational units and their interconnections that realize the architectural specifications. Eg: control signals, interfaces between computer and peripherals, and memory technology.

Question 1.2

What, in general terms, is the distinction between computer structure and computer function ?

Computer structure - The way in which the components of a computer are interrelated.

Computer function - The operation of each individual component as part of the structure.

Question 1.3

What are the four main functions of a computer ?

- Data processing;
- Data storage;
- Data movement;
- Control.

Question 1.4

List and briefly define the main structural components of a **computer**.

Central processing unit (CPU) - Controls the operation of a computer and performs its data processing functions; referred to as *processor*.

Main memory - Stores data.

I/O - Moves data between the computer and its external environment.

System interconnection - Mechanism that provides for communication among CPU, main memory, and I/O. Eg: System bus, consisting of a number of conducting wires to which all the other components are attached.

Question 1.5

List and briefly define the main structural components of a **processor**.

Control unit - Controls the operation of the CPU and hence the computer.

Arithmetic and logic unit (ALU) - Performs the computers' data processing functions.

Registers - Provide storage internal to the CPU.

CPU interconnection - Mechanism that provides for communication among the control unit, ALU, and registers.

--ooOoo--

Chapter 2 - Computer Evolution & Performance

Question 2.1

What is a stored program computer ?

In a stored program computer programs are represented in a form suitable for storing in memory alongside the data. The computer gets its instructions by reading them from memory, and a program can be set or altered by setting the values of a portion of memory.

Question 2.2

What are the four main components of any general-purpose computer ?

Main memory - stores both data and instructions.

Arithmetic and logic unit (ALU) - capable of operating on binary data.

Control unit - interprets the instructions in *memory* and causes them to be executed.

Input and output system (I/O) - operated by the *control unit*.

Question 2.3

At the integrated circuit level, what are the three principle constituents of a computer system ?

- Gates
- Memory cells
- Interconnections between gates and memory cells.

Question 2.4

Explain Moores' Law.

Moore observed that the number of transistors that could be put on a single chip was doubling every year, and correctly predicted that this pace would continue into the near future.

Question 2.5

List and explain the key characteristics of a computer family.

Similar or identical instruction set - In many cases the same set of machine instructions is supported on all members of the family. So a program that will execute on one machine will also execute on any other.

Similar or identical operating system - The same basic operating system is available for all family members.

Increasing speed - The rate of instruction execution increases from lower to higher family members.

Increasing number of I/O ports - from lower to higher family members.

Increasing memory size - from lower to higher family members.

Increasing cost - from lower to higher family members.

Question 2.6

What is the key distinguishing feature of a microprocessor ?

In a microprocessor, all of the components of the CPU are on a single chip.

--ooOoo--

Chapter 3 - A Top-Level View of Computer Function & Interconnection

Question 3.1

What general categories of functions are specified by computer instructions ?

- **Processor - memory**: Data may be transferred from processor to memory, or from memory to processor;
- **Processor - I/O**: Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module;
- **Data processing**: The processor may perform some arithmetic or logic operation on data;
- **Control**: An instruction may specify that the sequence of execution be altered.

Question 3.2

List and briefly define the possible states that define an instruction execution.

- **Instruction address calculation** (iac): Determine the address of the next instruction to be executed;
- **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.

Question 3.3

List and briefly define two approaches to dealing with multiple interrupts.

- Disable all interrupts while an interrupt is being processed;
- Define priorities for interrupts, and allow an interrupt of higher priority to cause a lower priority interrupt handler to be interrupted.

Question 3.4

What types of transfers must a computers' interconnection structure (eg: bus) support ?

- **Memory to processor**: The processor reads an instruction or a unit of data from memory;
- **Processor to memory**: The processor writes a unit of data to memory;

- **I/O to processor**: The processor reads data from an I/O device via an I/O module;
- **Processor to I/O**: The processor sends data to the I/O device;
- **I/O to or from memory**: For these two cases, an I/O module is allowed to exchange data directly with memory, without going through the processor, using direct memory access (DMA).

Question 3.5

What is the benefit of using a multiple-bus architecture compared to a single-bus architecture ?

- Multiple-bus architecture - fewer devices per bus;
    - Reduces propagation delay, as each bus can be shorter; and
    - Reduces bottleneck effects.

Question 3.6

List and briefly define the functional groups of signal lines for PCI.

n/a

--ooOoo--

Chapter 4 - Cache Memory

Question 4.1

What are the differences among sequential access, direct access, and random access ?

Sequential access - Memory is organized into units of data, called records. Access must be made in a specific linear sequence.

Direct access - Individual blocks or records have a unique address based on a physical location. Access is accomplished by direct access to reach a general vicinity plus sequential searching, counting, or waiting to reach the final location.

Random access - Each addressable location in memory has a unique, physically wired-in addressing mechanism. The time to access a given location is independent of the sequence of prior accesses and is constant.

Question 4.2

What is the general relationship among access time, memory cost, and capacity ?

- Faster access time - greater cost per bit;
- Greater capacity - lower cost per bit;
- Greater capacity - slower access time.

Question 4.3

How does the *principle of locality* relate to the use of multiple memory levels ?

It is possible to organize data across a memory hierarchy such that the percentage of accesses to each successively lower level is substantially less than that of the level above. Because memory references tend to cluster, the data in the higher level memory need not change very often to satisfy memory access requests.

Question 4.4

What are the differences among direct mapping, associative mapping, and set-associative mapping ?

In a cache system -

Direct mapping - maps each block of main memory into only one possible cache line.

Associative mapping - permits each main memory block to be loaded into any line of the cache.

Set-associative mapping - the cache is divided into a number of sets of cache lines; each main memory block can be mapped into any line in a particular set.

Question 4.5

For a direct-mapped cache, a main memory address is viewed as consisting of three fields. List and define the three fields.

n/a

Question 4.6

For an associative cache, a main memory address is viewed as consisting of two fields. List and define the two fields.

n/a

Question 4.7

For a set-associative cache, a main memory address is viewed as consisting of three fields. List and define the three fields.

n/a

Question 4.8

What is the distinction between spatial locality and temporal locality ?

Spatial locality - refers to the tendency of execution to involve a number of memory locations that are clustered.

Temporal locality - refers to the tendency for a processor to access memory locations that have been used recently.

Question 4.9

In general, what are the strategies for exploiting spatial locality and temporal locality ?

Spatial locality - is generally exploited by using larger cache blocks, and by incorporating prefetching mechanisms (fetching items of anticipated use) into the cache control logic;

Temporal locality - is exploited by keeping recently used instruction and data values in cache memory and by exploiting a cache hierarchy.

--ooOoo--

Chapter 5 - Internal Memory Technology

Question 5.1

What are the key properties of semiconductor memory ?

- They exhibit two stable (or semistable) states, which can be used to represent binary 1 or 0;
- they are capable of being written into (at least once), to set the state;
- they are capable of being read to sense the state.

Question 5.2

What are two senses in which the term *random-access memory* is used ?

- A memory in which individual words of memory are directly accessed through wired-in addressing logic;

- Semiconductor main memory in which it is possible both to read data from the memory, and to write new data into the memory easily and rapidly.

Question 5.3

What is the difference between DRAM and SRAM in terms of application ?

- **SRAM** - is used for cache memory (both on and off chip);

- **DRAM** - is used for main memory.

Question 5.4

What is the difference between DRAM and SRAM in terms of characteristics such as speed, size, cost ?

- **DRAM** - are less expensive and smaller than SRAM's;

- **SRAM** - generally have faster access times than DRAM's.

Question 5.5

Explain why one type of RAM is considered to be analogue and the other digital.

- **DRAM** - is essentially an analogue device using a capacitor, which can store any charge value within a range. A threshhold value determines whether the charge is interpreted as 1 or 0;

- **SRAM** - is a digital device, in which binary values are stored using traditional flip-flop logic-gate configurations.

Question 5.6

What are some applications for ROM ?

- Microprogrammed control unit memory;
- Library sub-routines for frequently wanted functions;
- System programs;
- Function tables.

Question 5.7

What are the differences among EPROM, EEPROM, and flash memory ?

- **EPROM** -
  - is read and written electrically;
  - before a write operation, all the storage cells must be erased to the same initial state by exposure of the packaged chip to ultraviolet radiation;

- **EEPROM** -
  - is a read-mostly memory that can be written into at any time without erasing prior contents;
  - only the byte, or bytes, addresses are updated;

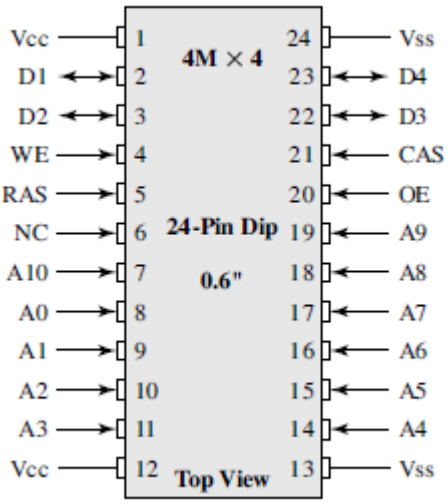
- **FLASH memory** -
  - is intermediate between EPROM and EEPROM in both cost and functionality;
  - Like EEPROM, flash memory uses an electrical erasing technology. Erasing is much faster than EPROM. It is possible to erase blocks of memory rather than an entire chip, but does not provide byte-level erasure;
  - Like EPROM, flash memory uses only one transistor per bit, so achieves the high density of EPROM (compared to EEPROM).

Question 5.8

Explain the function of each pin in Figure 5.4 (b).

1. Vcc — Power supply to chip.
2. D1 - D4 — Data pins - input/output, 4 lines.
4. WE — Write enable - indicates write operation.
5. RAS — Row Address Select
6. NC — No connect - to make even number of pins.
7. A0-A10 — Address of word being accessed.
   11 address pins reqd for 4M x 4.
24. Vss — Ground pin.
21. CAS — Column Address Select.
20. OE — Output enable - indicates read operation.

Fig 5.4 (b) 16 Mbit DRAM



Question 5.9

What is a parity bit ?

A bit appended to an array of binary digits to make the sum of all the binary digits, including the parity bit, always odd (odd parity), or always even (even parity).

Question 5.10

How is the syndrome for the Hamming code interpreted ?

n/a

Question 5.11

How does SDRAM differ from ordinary DRAM ?

- Traditional DRAM is asynchronous;
- SDRAM exchanges data with the processor synchronized to an external clock signal, and running at the full speed of the processor / memory bus without imposing wait states.

--ooOoo--

Chapter 6 - External Memory

Question 6.1

What are the advantages of using a glass substrate for a magnetic disk ?

- Improvement in the uniformity of the magnetic film surface to increase disk reliability;
- A significant reduction in overall surface defects to help reduce read / write errors;
- Ability to support lower fly heights;
- Better stiffness to reduce disk dynamics;
- Greater ability to withstand shock and damage.

Question 6.2

How are data written onto a magnetic disk ?

- Write mechanism is based on the fact that electricity flowing through a coil produces a magnetic field;
- Pulses are sent to the write head, and magnetic patterns are recorded on the surface below, with different patterns for positive and negative currents;
- An electric current in the wire induces a magnetic field across the gap, which in turn magnetizes a small area of the recording medium;
- Reversing the direction of the current reverses the direction of the magnetization on the recording medium.

Question 6.3

How are data read from a magnetic disk ?

- The read head consists of a partially shielded magnetoresistive (MR) sensor;
- The MR material has an electrical resistance that depends on the direction of the magnetization of the medium moving under it;
- By passing a current through the MR sensor, resistance changes are detected as voltage signals.

Question 6.4

Explain the difference between a simple CAV system and a multiple zoned recording system.

- **Constant angular velocity** (CAV) system the number of bits per track is constant;

- An increase in density is achieved with **multiple zoned recording**, in which the surface is divided into a number of zones, with zones further from the centre containing more bits than zones closer to the centre.

Question 6.5

Define the terms *track*, *cylinder*, and *sector*.

Track - On a magnetic disk, data is organized on the platter in concentric sets of rings, called tracks.

Cylinder - On a disk with multiple platters, the set of all tracks in the same relative position on the platter is referred to as a cylinder.

Sector - Data are transferred to and from the disk in sectors.

Question 6.6

What is the typical disk sector size ?

512 bytes.

Question 6.7

Define the terms *seek time*, *rotational delay*, *access time*, and *transfer time*.

On a movable-head system -

Seek time - Time taken to position the head at the track.

Rotational delay - Once the track is selected, the disk controller waits until the appropriate sector rotates to line up with the head. The time it takes for the beginning of the sector to reach the head is known as the *rotational delay*.

Access time - The sum of the *seek time*, if any, plus the *rotational delay*. The time it takes to get into position to read or write.

Transfer time - Time taken for data transfer. Once the head is in position, the read or write operation is performed as the sector moves under the head - data transfer portion of the operation.

Question 6.8

What common characteristics are shared by all RAID levels ?

- RAID is a set of physical disk drives viewed by the operating system as a single logical drive;
- Data are distributed across the physical drives of an array;
- Redundant disk capacity is used to store parity information, which guarantees data recoverability in case of disk failure. (Except RAID 0, and RAID 1 which mirrors disks.)

Question 6.9

Briefly define the seven RAID levels.

- RAID 0 - Non-redundant.
- RAID 1 - Mirrored, every disk has a mirror disk containing the same data.
- RAID 2 - Redundant via Hamming code; an error-correcting code is calculated across corresponding bits on each data disk, and the bits of the code are stored in the corresponding bit positions on multiple parity disks.
- RAID 3 - Bit-interleaved parity;
- RAID 4 - Block-interleaved parity;
- RAID 5 - Block-interleaved distributed parity;
- RAID 6 - Block-interleaved dual distributed parity;

Question 6.10

Explain the term *striped data*.

The disk is divided into strips, which may be physical blocks, sectors, or some other unit. The strips are mapped round robin to consecutive array members. A set of logically consecutive strips that maps exactly one strip to each array member is referred to as a stripe.

Question 6.11

How is redundancy achieved in a RAID system ?

For RAID Level 1 redundancy is achieved by having two identical copies of all data. For higher levels, redundancy is achieved by the use of error-correcting codes.

Question 6.12

In the context of RAID, what is the distinction between parallel access and independent access ?

Parallel access - All member disks participate in the execution of every I/O request. Typically, the spindles of the individual drives are synchronized so that each disk head is in the same position on each disk at any given time.

Independent access - Each member disk operates independently, so that separate I/O requests can be satisfied in parallel.

Question 6.13

What is the difference between CAV and CLV ?

Constant angular velocity (CAV) - Number of bits per track is constant, disk rotates at constant velocity.

Constant linear velocity (CLV) - Disk rotates more slowly for access near the outer edge than near the centre. The capacity (density) of a track and the rotational delay both increase for positions nearer the outer edge of a disk.

Question 6.14

What differences between a CD and a DVD account for the larger capacity of the latter ?

- Bits are packed more closely on a DVD, which results in approx seven-fold increase in capacity;
- DVD has a second layer of pits and lands on top of the first layer. Almost doubles capacity of DVD;
- DVD-ROM can be double sided, whereas CD can only be recorded on one side.

Question 6.15

Explain serpentine recording.

When tape data are being recorded, first set of bits is recorded in a track along the whole length of the tape. At the end of the tape the heads are repositioned to record a new track, and the tape is again recorded on its whole length, in the opposite direction. Process continues until tape is full.

--OOo--

Chapter 7 - Input / Output

Question 7.1

List three broad classifications of external, or peripheral, devices.

- **Human readable** - Suitable for communicating with the computer user;
- **Machine readable** - Suitable for communicating with equipment;
- **Communication** - Suitable for communicating with remote devices.

Question 7.2

What is the International Reference Alphabet ?

IRA also known as ASCII.

Most commonly used text code - each character is represented by a unique 7-bit binary code, so 128 different characters can be represented.

Question 7.3

What are the major functions of an I/O module ?

- Control and timing;
- Processor communication;
- Device communication;
- Data buffering;
- Error detection.

Question 7.4

List and briefly define three techniques for performing I/O.

- **Programmed I/O** - The processor issues an I/O command, on behalf of a process, to an I/O module; that process then busy-waits for the operation to be completed before proceeding.
- **Interrupt-driven I/**O - The processor issues an I/O command on behalf of a process, continues to execute subsequent instructions, and is interrupted by the I/O module when the latter has completed its work.
- **Direct Memory Access** (DMA) - A DMA module controls the exchange of data between main memory and an I/O module. The processor sends a request for the transfer of a block of data to the DMA module, and is interrupted only after the entire block has been transferred.

Question 7.5

What is the difference between memory-mapped I/O, and isolated I/O ?

- **Memory-mapped I/O** - there is a single address space for memory locations and I/O devices. The processor treats the status and data registers of I/O modules as memory locations and uses the same machine instructions to access both memory and I/O devices.

- **Isolated I/O** - a command specifies whether the address refers to a memory location or an I/O device. The full range of addresses may be available for both.

Question 7.6

When a device interrupt occurs, how does the processor determine which device issued the interrupt ?

Four categories of techniques are in common use -

- multiple interrupt lines;
- software poll;
- daisy chain (hardware poll, vectored);
- bus arbitration (vectored).

Question 7.7

When a DMA module takes control of a bus, and while it retains control of the bus, what does the processor do ?

The processor pauses for each bus cycle stolen by the DMA module.

--ooOoo--

Chapter 9 - Computer Arithmetic

Question 9.1

Briefly explain the following representations: sign magnitude, twos complement, biased.

Sign-magnitude representation - The leftmost bit is the sign (0: positive, 1:negative), remaining bits are the magnitude of the number.

Twos' complement representation - A positive integer is represented as in sign-magnitude. A negative number is represented by taking the twos' complement of the number.

Biased representation - A fixed value, called the bias, is added to the integer.

Question 9.2

Explain how to determine if a number is negative in the following representations: sign magnitude, twos complement, biased.

Sign-magnitude - the left-most bit is a sign bit.

Twos'complement - the left-most bit is a sign bit.

Biased - a number is negative if the value of the representation is less than the bias.

Question 9.3

What is the sign-extension rule for twos complement numbers ?

Add additional bit positions to the left, and fill in with the value of the original sign bit.

Question 9.4

How can you form the negation of an integer in twos complement representation ?

Take the ones' complement of the number, then add 1 to the resulting bit pattern, viewed as an unsigned integer.

Or, copy each bit from the rhs up to and including the first 1, then reverse all the other bits.

Question 9.5

In general terms, when does the twos complement operation on an n-bit integer produce the same integer ?

When performed on the n-bit integer -2^{n-1} (One followed by n-1 zeroes).

Question 9.6

What is the difference between the twos complement representation of a number and the twos complement of a number ?

Twos' complement representation - of a number is the bit pattern used to represent an integer.

Twos' complement - of a number is the operation that computes the negation of a number in twos' complement representation.

Question 9.7

If we treat 2 twos complement numbers as unsigned integers for purposes of addition, the result is correct if interpreted as a twos complement number. This is not true for multiplication. Why ?

n/a

Question 9.8

What are the four essential elements of a number in floating-point representation ?

Sign, significand, exponent, base.

Question 9.9

What is the benefit of using biased representation for the exponent portion of a floating-point number ?

An advantage of biased representation is that non-negative floating-point numbers can be treated as integers for comparison purposes.

Question 9.10

What are the differences among positive overflow, exponent overflow, and significand overflow ?

n/a

Question 9.11

What are the basic elements of floating-point addition and subtraction ?

n/a

Question 9.12

Give a reason for the use of guard bits.

n/a

Question 9.13

List four alternative methods of rounding the result of a floating-point operation.

n/a

--ooOoo--

Chapter 10 - Instruction Sets

Question 10.1

What are the typical elements of a machine instruction ?

Opcode -	Specifies the operation to be performed.
Source and destination operand references -	Specify the input and output locations for the operation.
Next instruction reference -	Usually implicit.

Question 10.2

What types of locations can hold source and destination operands ?

Registers and memory.

Question 10.3

If an instruction contains four addresses, what might be the purpose of each address ?

Two operands, one result, and the address of the next instruction.

Question 10.4

List and briefly explain five important instruction set design issues.

- **Operation repertoire** - How many and which operations to provide, and how complex operations should be.
- **Data types** - The various types of data upon which operations are performed.
- **Instruction format** - Instruction length (in bits), number of addresses, size of various fields etc.
- **Registers** - Number of CPU registers that can be referenced by instructions, and their use.
- **Addressing** - The mode, or modes, by which the address of an operand is specified.

Question 10.5

What types of operands are typical in machine instruction sets ?

Addresses, numbers, characters, logical data.

Question 10.6

What is the relationship between the IRA character code and the packed decimal representation ?

The digits 0 - 9 are represented by their binary equivalents, 0000 to 1001, in the 4 right-most bits.

Question 10.7

What is the difference between an arithmetic shift and a logical shift ?

Logical shift - the bits of a word are shifted left or right. On one end, the bit shifted out is lost. On the other end, a zero is shifted in.

Arithmetic shift - operation treats the data as a signed integer and does not shift the sign bit. On a right arithmetic shift, the sign bit is replicated into the bit position to its right. On a left arithmetic shift, a left logical shift is performed on all bits except the sign bit, which is retained.

Question 10.8

Why are transfer of control instructions needed ?

- Repetition - It may be necessary to execute some instructions multiple times;
- Branching - process different instructions depending on the value of a condition;
- Modular program - mechanism for breaking up a large or complex task into smaller tasks.

Question 10.9

List and briefly explain two common ways of generating the condition to be tested in a conditional branch instruction.

- Condition code is set as the result of an operation;
- Perform a comparison, and specify a branch dependent on the outcome of the comparison.

Question 10.10

What is meant by the term *nesting of procedures* ?

Occurs when a procedure is called from within another procedure.

Question 10.11

List three possible places for storing the return address for a procedure return.

Register, start of procedure, stack.

Question 10.12

What is a reentrant procedure ?

Procedure that can have several concurrent calls to it open at the same time.

Question 10.13

What is reverse Polish notation ?

The operator follows the two operands, as opposed to being in between them. (Postfix)

Question 10.14

What is the difference between big endian and little endian ?

Big-endian: A multibyte numerical value stored with the most significant byte in the lowest numerical address.

Little-endian: A multibyte numerical value stored with the most significant byte in the highest numerical address.

--ooOoo--

Chapter 11 - Instruction Sets: Addressing Modes and Formats

Question 11.1

Briefly define immediate addressing.

The value of the operand is in the instruction.

Question 11.2

Briefly define direct addressing.

The address field contains the effective address of the operand.

Question 11.3

Briefly define indirect addressing.

The address field refers to the address of a word in memory, which in turn contains the effective address of the operand.

Question 11.4

Briefly define register addressing.

The address field refers to a register that contains the operand.

Question 11.5

Briefly define register indirect addressing.

The address field refers to a register, which in turn contains the effective address of the operand.

Question 11.6

Briefly define displacement addressing.

The instruction has two address fields, at least one of which is explicit. The value contained in one address field (value = A) is used directly. The other address field refers to a register whose contents are added to A to produce the effective address.

Question 11.7

Briefly define relative addressing.

The implicitly referenced register is the program counter (PC). That is, the current instruction address is

added to the address field to produce the EA.

Question 11.8

What is the advantage of autoindexing ?

Typically, there is a need to increment or decrement the index register after each reference to it. Because this is such a common operation, some systems will automatically do this as part of the same instruction cycle, using auto indexing.

Question 11.9

What is the difference between post indexing and pre indexing ?

There are two forms of addressing, both of which involve indirect addressing and indexing -

- **pre indexing** - the indexing is performed before the indirection;
- **post indexing** - the indexing is performed after the indirection.

Question 11.10

What facts go into determining the use of the addressing bits of an instruction ?

n/a

Question 11.11

What are the advantages and disadvantages of using a variable-length instruction format ?

- Advantages -
  - It is easy to provide a large repertoire of opcodes, with different opcode lengths;
  - Addressing can be more flexible, with various combinations of register and memory references plus addressing modes;

- Disadvantages -
  - an increase in the complexity of the CPU.

--ooOoo--

Chapter 12 - Processor Structure and Function

Question 12.1

What general roles are performed by processor registers ?

User-visible registers - These enable the machine- or assembly language programmer to minimize main-memory references by optimizing use of registers.

Control and status registers - These are used by the control unit to control the operation of the CPU, and by privileged operating system programs to control the execution of programs.

Question 12.2

What categories of data are commonly supported by user-visible registers ?

General purpose, Data, Address, Condition codes.

Question 12.3

What is the function of condition codes ?

Condition codes are bits set by the CPU hardware as the result of operations. For example, an arithmetic operation may produce a positive, negative, zero, or overflow result. In addition to the result itself being stored in a register or memory, a condition code is also set. The code may subsequently be tested as part of a conditional branch operation.

Question 12.4

What is a program status word ?

All CPU designs include a register, or set of registers, often known as the *program status word* (PSW) that contains status information. The PSW typically contains condition codes plus other status information.

Question 12.5

Why is a two-stage instruction pipeline unlikely to cut the instruction cycle time in half, compared with the use of no pipeline ?

n/a

Question 12.6

List and briefly explain various ways in which an instruction pipeline can deal with conditional branch instructions.

n/a

Question 12.7

How are history bits used for branch prediction ?

One or more bits that reflect the recent history of the instruction can be associated with each conditional branch instruction. These bits are referred to as a taken / not taken switch that directs the processor to make a particular decision the next time the instruction is encountered.

--ooOoo--

Chapter 13 - Reduced Instruction Set Computers (RISC)

Question 13.1

What are some typical distinguishing characteristics of RISC organization ?

- a limited instruction set with a fixed format;
- a large number of registers or the use of a compiler that optimizes register usage; and
- an emphasis on optimizing the instruction pipeline.

Question 13.2

Briefly explain the two basic approaches used to minimize register-memory operations on RISC machines.

Software approach - is to rely on the compiler to maximize register usage. The compiler will attempt to allocate registers to those variables that will be used the most in a given time period. This approach requires the use of sophisticated program-analysis algorithms.

Hardware approach - is simply to use more registers so that more variables can be held in registers for longer periods of time.

Question 13.3

If a circular register buffer is used to handle local variables for nested procedures, describe two approaches for handling global variables.

- Variables declared as global in an HLL can be assigned memory locations by the compiler, and all machine instructions that reference these variables will use memory-reference operands.

- Incorporate a set of global registers in the processor. These registers would be fixed in number and available to all procedures.

Question 13.4

What are some typical characteristics of a RISC instruction set architecture ?

- One instruction per cycle;
- Register-to-register operations;
- Simple addressing modes;
- Simple instruction formats.

Question 13.5

What is a delayed branch ?

A way of increasing the efficiency of the pipeline, makes use of a branch that does not take effect until after execution of the following instruction.

--ooOoo--

Appendix B - Assembly Language

Question B.1

List some reasons why it is worthwhile to study assembly language programming.

- It clarifies the execution of instructions;
- It shows how data is represented in memory;
- It shows how a program interacts with the operating system, processor, and the I/O system;
- It clarifies how a program accesses external devices;
- Understanding assembly language programming makes students better high-level language (HLL) programmers, by giving them a better idea of the target language that the HLL must be translated to.

Question B.2

What is an assembly language ?

Assembly language is a programming language that is one step away from machine language. Assembly language includes symbolic names for locations. It also includes directives and macros.

Question B.3

List some disadvantages of assembly language compared to high-level languages.

- Development time - Writing code in assembly language takes much longer than in a high level language.
- Reliability and security - It is easy to make errors in assembly code. There is no error checking.
- Debugging and verifying - Assembly code is more difficult to debug and verify because there are more possibilities for errors than in high-level code.
- Maintainability - Assembly code is more difficult to modify and maintain because the language allows unstructured spaghetti code, and all kinds of dirty tricks that are difficult for others to understand.
- Portability - Assembly code is very platform-specific. Porting to a different platform is difficult.

Question B.4

List some advantages of assembly language compared to high-level languages.

- Debugging and verifying - Looking at compiler-generated assembly code or the disassembly window in a debugger is useful for finding errors and for checking how well a compiler optimizes a particular piece of code.
- Making compilers - Understanding assembly coding techniques is necessary for making compilers,

debuggers, and other development tools.

- Embedded systems - Small embedded systems have fewer resources than PC's and mainframes. Assembly programming can be necessary for optimizing code for speed or size in small embedded systems.

- Hardware drivers and system code - Accessing hardware, system control registers etc may sometimes be difficult or impossible with high level code.

- Accessing instructions that are not accessible from high-level language. Certain assembly instructions have no high-level language equivalent.

- Self-modifying code - is generally not profitable because it interferes with efficient code caching. It may, however, be advantageous for example to include a small compiler in math programs where a user-defined function has to be calculated many times.

- Optimizing code for size - Storage space and memory is so cheap nowadays that it is not worth the effort to use assembly language for reducing code size. However, cache size is still such a critical resource that it may be useful in some cases to optimize a critical piece of code for size in order to make it fit into the code cache.

- Optimizing code for speed - Compilers generally optimize code quite well, but there are still cases where compilers perform poorly, and where dramatic increases in speed can be achieved by careful assembly programming.

- Function libraries - The total benefit of optimizing code is higher in function libraries that are used by many programmers.

- Making function libraries compatible with multiple compilers and operating systems - requires assembly programming.

Question B.5

What are the typical elements of an assembly language statement ?

Label, Mnemonic, Operand/s, Comment

Question B.6

List and briefly define four different kinds of assembly language statements.

- **Instruction** - symbolic representation of machine language instructions.

- **Directive** - instruction to the assembler to perform a specified action during the assembly process.

- **Macro definition** - is a section of code that the programmer writes once, and then can use many times. The assembler replaces the macro call with the macro itself.

- **Comment** - A statement consisting entirely of a comment.

Question B.7

What is the difference between a one-pass assembler and a two-pass assembler ?

Two-pass assembler - takes a first pass through the assembly program to construct a symbol table that contains a list of all labels and their associated location counter values. It then takes a second pass to translate the assembly program into object code.

One-pass assembler - combines both operations in a single pass, and resolves forward references on the fly.

--ooOoo--