Tutorial-3: Real-Time Computer Systems and Architecture

A 1. The essential elements of a computer instruction are the opcode, which specifies the operation to be performed, the source and destination operand references, which specify the input and output locations for the operation, and a next instruction reference, which is usually implicit.

A 2. **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, and so on.

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.

A 3. For the IRA bit pattern 011XXXX, the digits 0 through 9 are represented by their binary equivalents, 0000 through 1001, in the right-most 4 bits. This is the same code as packed decimal.

A 4. With a 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 0 is shifted in. The 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 logical left shift is performed on all bits but the sign bit, which is retained.

A 5.

1. In the practical use of computers, it is essential to be able to execute each instruction more than once and perhaps many thousands of times. It may require thousands or perhaps millions of instructions to implement an application. This would be unthinkable if each instruction had to be written out separately. If a table or a list of items is to be processed, a program loop is needed. One sequence of instructions is executed repeatedly to process all the data.
2. Virtually all programs involve some decision-making. We would like the computer to do one thing if one condition holds, and another thing if another condition holds.
3. 3. To compose correctly a large or even medium-size computer program is an exceedingly difficult task. It helps if there are mechanisms for breaking the task up into smaller pieces that can be worked on one at a time.

A 6. A multibyte numerical value stored with the most significant byte in the lowest numerical address is stored in big-endian fashion. A multibyte numerical value stored with the most significant byte in the highest numerical address is stored in little-endian fashion.

A 7. These are two forms of addressing, both of which involve indirect addressing and indexing. With **preindexing**, the indexing is performed before the indirection. With **postindexing**, the indexing is performed after the indirection.

A 8. **Number of addressing modes:** Sometimes an addressing mode can be indicated implicitly. In other cases, the addressing modes must be explicit, and one or more mode bits will be needed.

Number of operands: Typical instructions on today's machines provide for two operands. Each operand address in the instruction might require its own mode indicator, or the use of a mode indicator could be limited to just one of the address fields.

Register versus memory: The more that registers can be used for operand references, the fewer bits are needed.

Number of register sets: One advantage of using multiple register sets is that, for a fixed number of registers, a functional split requires fewer bits to be used in the instruction.

Address range: For addresses that reference memory, the range of addresses that can be referenced is related to the number of address bits. Because this imposes a severe limitation, direct addressing is rarely used. With displacement addressing, the range is opened up to the length of the address register.

Address granularity: In a system with 16- or 32-bit words, an address can reference a word or a byte at the designer's choice. Byte addressing is convenient for character manipulation but requires, for a fixed-size memory, more address bits.

A 9. **Advantages:** It 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.

A 10.
a) the address field
b) memory location 14
c) the memory location whose address is in memory location 14
d) register 14
e) the memory location whose address is in register 14

A 11. (PC + 1) + Relative Address = Effective Address
Relative Address = –621 + 530 = –91
Converting to twos-complement representation, we have: 1110100101.

A 12.
a) 3 times: fetch instruction; fetch operand reference; fetch operand.
b) 2 times: fetch instruction; fetch operand reference and load into PC.

A 13. Load the address into a register. Then use displacement addressing with a displacement of 0.

A 14. The 32-bit instruction length yields incremental improvements. The 16-bit length can already include the most useful operations and addressing modes. Thus, relatively speaking, we don't have twice as much "utility".

A 15. Let X be the number of one-address instructions. The feasibility of having *K* two-address, X one-address, and L zero-address instructions, all in a 16-bit instruction word, requires that:

$$(K \times 2^6 \times 2^6) + (X \times 2^6) + L = 2^{16}$$

Solving for X:

$$X = (2^{16} - (K \times 2^6 \times 2^6) - L)/2^6$$

To verify this result, consider the case of no zero-address and no two-address instructions; that is, L = K = 0. In this case, we have

$$X = 2^{16}/2^6 = 2^{10}$$

This is what it should be when 10 bits are used for opcodes and 6 bits for addresses.

A 16. **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.

A 17. 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.

A 18.
1) The execution time will generally be longer than the fetch time. Execution will involve reading and storing operands and the performance of some operation. Thus, the fetch stage may have to wait for some time before it can empty its buffer.
2) A conditional branch instruction makes the address of the next instruction to be fetched unknown. Thus, the fetch stage must wait until it receives the next instruction address from the execute stage. The execute stage may then have to wait while the next instruction is fetched.

A 19. **Multiple streams:** A brute-force approach is to replicate the initial portions of the pipeline and allow the pipeline to fetch both instructions, making use of two streams.

Prefetch branch target: When a conditional branch is recognized, the target of the branch is prefetched, in addition to the instruction following the branch. This target is then saved until the branch instruction is executed. If the branch is taken, the target has already been prefetched.

Loop buffer: A loop buffer is a small, very-high-speed memory maintained by the instruction fetch stage of the pipeline and containing the n most recently fetched instructions, in sequence. If a branch is to be taken, the hardware first checks whether the branch target is within the buffer. If so, the next instruction is fetched from the buffer.

Branch prediction: A prediction is made whether a conditional branch will be taken when executed, and subsequent instructions are fetched accordingly.

Delayed branch: It is possible to improve pipeline performance by automatically rearranging instructions within a program, so that branch instructions occur later than actually desired.

A 20. a) 0.2 ns
b) 0.6 ns