Tutorial-1: Real-Time Computer Systems and Architecture

A1. **Computer architecture** refers to those attributes of a system visible to a programmer or, put another way, those attributes that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent various data types (e.g., numbers, characters), I/O
mechanisms, and techniques for addressing memory.

Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Organizational attributes include those hardware details transparent to the
programmer, such as control signals; interfaces between the computer and peripherals; and the memory technology used.

A2. Computer structure refers to the way in which the components of a computer are interrelated. Computer function refers to the operation of each individual component as part of the structure.

A3. **Central processing unit (CPU):** Controls the operation of the computer and performs its data processing functions; often simply referred to as processor.

Main memory: Stores data.

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

System interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O. A common example of system interconnection is by means of a system bus, consisting of a number of conducting wires to which all the other components attach.

A4 **Control unit:** Controls the operation of the CPU and hence the Computer.

Arithmetic and logic unit (ALU): Performs the computer's data processing functions.

Registers: Provides storage internal to the CPU.

CPU interconnection: Some mechanism that provides for communication among the control unit, ALU, and registers.

A5. A device with tightly coupled computing hardware and software that are designed to perform a dedicated function.

- Usually encapsulated by the device it controls.
- Often hidden from view – without the awareness of their users.
- Usually as a sub-system within a larger system – embedding system.
- Multiple embedded systems can coexist in an mbedding system.

Washing machine, iPhone, network router, TV, wireless phone etc..

A6. Main requirements are:

- Environmental – size, power (heat), weight, and radiation-hardened
- Performance –responsive, predictability
- Economic – cost, time-to-market
- Consequential – safety, reliability, security

A7.

a) No. These programs are never considered to be embedded because they are not an integral component of a larger system.
b) Yes, regardless of what the disk drive is used for. The software (firmware, actually) within the disk drive controls the HDA (head disk assembly) hardware and is hard real-time as well.
c) No, because that computer may be a general-purpose computer that is not part of a larger system.
d) No. People often say that PDAs are embedded because they are very small and constrained and because PDA OS and application software is kept in non-volatile memory, but PDAs parallel the desktop systems used to run office productivity applications, and no special hardware is being controlled.
e) Yes. The firmware in the cell phone is controlling the radio hardware.
f) Yes. These computers were generally some of the most powerful computers available when the system was built, are located in a large computer room occupying almost one whole floor of a building, and may be hundreds of meters away from the radar hardware. However, the software running in these computers controls the radar hardware; therefore, the computers are an integral component of a larger system.
g) If the FMS is not connected to the avionics and is used only for logistics computations, a function readily performed on a laptop, then the FMS is clearly not embedded.
h) Yes, both in the simulator, and in the thing being tested in the HIL simulator. Hardware is being controlled on both sides.
i) Yes. In this case the “system” is the combination of the pacemaker and the person's heart.
j) Yes. It is part of a larger system, the engine, and it is directly monitoring and controlling the engine through special hardware

A8. Real-time system is a system that responds within specified times. (Not necessarily fast – must meet timing deadline).

A9. Hard Real-Time System:

- when some of the deadlines are hard.
- Late results (after its deadline) may cause fatal flaw or catastrophic consequences.
- Many are safety critical.
- Requires a means for validating that deadlines are always met.

Soft Teal-Time:

- Timely completion desire.
- Late results can be useless or still useful although causing a performance degradation

Examples:

- Sensory data acquisition; data filtering and prediction; detection of critical conditions; data fusion and image processing; actuator servoing; low-level control of critical system components and action planning for systems that tightly interact with the environment.
- Video playing; audio/video encoding and decoding; on-line image processing; sensory data transmission in distributed systems.
- The command interpreter of the user interface; handling input data from the keyboard; displaying messages on the screen; representation of system state variables

A10. Purely cyclic

Every task executes periodically
Demands in resources (computing, communication, and storage) do not vary significantly from period to period
Example: most digital controllers and real-time monitors

Mostly cyclic

Most tasks execute periodically
The system must also respond to some external events (fault recovery and external commands) asynchronously
Example: modern avionics and process control systems

Asynchronous: mostly predictable

Most tasks are not periodic
The time between consecutive executions of a task may vary considerably, or the variations in resource utilization in different periods may be large
These variations have either bounded ranges or known statistics

Asynchronous: unpredictable

Applications that react to asynchronous events and have tasks with high run-time complexity
Example: intelligent real-time control systems

A11. **Pipelining:** The execution of an instruction involves multiple stages of operation, including fetching the instruction, decoding the opcode, fetching operands, performing a calculation, and so on.

Branch prediction: The processor looks ahead in the instruction code fetched from memory and predicts which branches, or groups of instructions, are likely to be processed next.

Superscalar execution: This is the ability to issue more than one instruction in every processor clock cycle.

Data flow analysis: The processor analyzes which instructions are dependent on each other's results, or data, to create an optimized schedule of instructions.

Speculative execution: Using branch prediction and data flow analysis, some processors speculatively execute instructions ahead of their actual appearance in the program execution, holding the results in temporary locations.

A12. **Multicore:** the use of multiple processing units, called cores, on a single chip.

Many integrated core (MIC): a chip containing a large number (50 or more) cores.

General-purpose computing on GPUs (GPGPU): A GPU designed to support a broad range of applications.

A13.
1. It is written in a high-level language, making it portable across different machines.
2. It is representative of a particular kind of programming domain or paradigm, such as systems programming, numerical programming, or commercial programming.
3. It can be measured easily
4. It has wide distribution.

A14.

We know that each computer executes the same number of instructions for the program; let's call this number I. First, find the number of processor clock cycles for each computer:

$$\text{CPU clock cycles}_A = I \times 2.0$$
$$\text{CPU clock cycles}_B = I \times 1.2$$

Now we can compute the CPU time for each computer:

$$\begin{aligned}\text{CPU time}_A &= \text{CPU clock cycles}_A \times \text{Clock cycle time} \\ &= I \times 2.0 \times 250 \text{ ps} = 500 \times I \text{ ps}\end{aligned}$$

Likewise, for B:

$$\text{CPU time}_B = I \times 1.2 \times 500 \text{ ps} = 600 \times I \text{ ps}$$

Clearly, computer A is faster. The amount faster is given by the ratio of the execution times:

$$\frac{\text{CPU performance}_A}{\text{CPU performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{600 \times I \text{ ps}}{500 \times I \text{ ps}} = 1.2$$

We can conclude that computer A is 1.2 times as fast as computer B for this program.

A15.

Sequence 1 executes 2 + 1 + 2 = 5 instructions. Sequence 2 executes 4 + 1 + 1 = 6 instructions. Therefore, sequence 1 executes fewer instructions.

We can use the equation for CPU clock cycles based on instruction count and CPI to find the total number of clock cycles for each sequence:

$$\text{CPU clock cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{C}_i)$$

This yields

$$\text{CPU clock cycles}_1 = (2 \times 1) + (1 \times 2) + (2 \times 3) = 2 + 2 + 6 = 10 \text{ cycles}$$

$$\text{CPU clock cycles}_2 = (4 \times 1) + (1 \times 2) + (1 \times 3) = 4 + 2 + 3 = 9 \text{ cycles}$$

So code sequence 2 is faster, even though it executes one extra instruction. Since code sequence 2 takes fewer overall clock cycles but has more instructions, it must have a lower CPI. The CPI values can be computed by

$$\text{CPI} = \frac{\text{CPU clock cycles}}{\text{Instruction count}}$$

$$\text{CPI}_1 = \frac{\text{CPU clock cycles}_1}{\text{Instruction count}_1} = \frac{10}{5} = 2.0$$

$$\text{CPI}_2 = \frac{\text{CPU clock cycles}_2}{\text{Instruction count}_2} = \frac{9}{6} = 1.5$$

A16. 0000 0010 0000 0000 (512 in two's complement)
1111 1110 0000 0000 (-512 – in two's Complement) (wrong answer - take note two's complement is used to represent both positive and negative value)

A17. 1111 1100 0000 0001 (-1023 in two's complement)
0000 0011 1111 1111 (1023 in two's complement) (wrong answer)

A18. Hex number = 37 (0011 0111)
Decimal value = 55 ($2^5+2^4+2^2+2^1+2^0$ or 32 + 16 + 4 + 2 + 1)

A19. 32 bit-data means 4 bytes wide. 16-bit address bus can address to 64 K words – therefore the largest size of the memory = 64K x 4 = 256 Kbytes.

A20. ARM - A (41_{16}) R (52_{16}) M ($4D_{16}$)

A21. a) Signed Integer –> -2^{31} to $+2^{31}-1$
b) Unsigned Integer -> 0 to + $2^{32}-1$

A22. a) $Cx16^7 + 5x16^6 + 7x16^5 = 3312451584_{10}$
b) $-(4x16^7 + 5x16^6 + 7x16^5) = -1164967936_{10}$