EMBEDDED SYSTEM

Introduction to Embedded System

- Embedded means something that is attached to another thing.
- An embedded system can be thought of as a computer hardware system having software embedded in it.
- An embedded system can be an independent system, or it can be a part of a large system.
- An embedded system is a microcontroller or microprocessor-based system which is designed to perform a specific task.
- An embedded system is a combination of computer hardware and software, either fixed in capability or programmable, designed for a specific function or functions within a larger system.
- **For example**: a fire alarm is an embedded system; it will sense only smoke.

Characteristics of Embedded Systems

Unlike general computer systems, embedded systems work only for a particular function in a time-bound manner. For instance, a washing machine can not multitask like a laptop. In this regard, here are some unique characteristics of an embedded system.

Sophisticated Functionality

The functionality of no two embedded system applications is bound to be the same. The functionality of a washing machine is different from that of a microwave. However, the functionality of a laptop and a desktop are almost the same.

Real-Time Operation

It doesn’t mean live operation. It means the software programs hardware to operate in a time-bound fashion. It could also have two modes: Hard and Soft. The former mode indicates the task has to be completed within the allotted time (ex: clock), but in the soft mode, the system could use additional time over the allotted time (ex: microwave).

Low Manufacturing Costs

As an embedded system design aims for any particular application, it involves less manufacturing cost as compared to a versatile general computing system. As a result, embedded systems also require less power to perform operations.

Characteristics of Embedded Systems Contd…

Processor and Memory
Depending on the type, processor and memory requirements may vary. For instance, small embedded systems would require less memory, but sophisticated systems demand more memory and run on multi-core processors.

Tight Design Constraints
There are many design constraints to consider around the cost, performance, size, and power of an embedded system to realize its absolute performance. These design factors are kept to a minimum to justify their simple function.

RISC & CISC

Introduction:
CISC

- It stands for complex instruction set computer.
- The main idea is that a single instruction will do all loading, evaluating, and storing operations just like a multiplication command will do stuff like loading data, evaluating, and storing it, hence it's complex.
- The CISC approach attempts to minimize the number of instructions per program but at the cost of an increase in the number of cycles per instruction.
- There is a huge cost of implementing a large number of instructions into the microprocessor.
- In a CISC such as 8051 microcontroller, the instruction are of 1, 2, 3 bytes.
- Instruction may take more than a single clock cycle to get executed.
- In a CISC processors there is one set of busses for data and one set of busses for address.

CISC Attributes

CISC instructions sets have some common characteristics:

- A 2-operand format, where instructions have a source and a destination. Register to register, register to memory, and memory to register commands.
- Variable length instructions where the length often varies according to the addressing mode
- Instructions which require multiple clock cycles to execute.

E.g. Pentium is considered a modern CISC processor

Most CISC hardware architectures have several characteristics in common:

- Complex instruction-decoding logic, driven by the need for a single instruction to support multiple addressing modes.
- A small number of general purpose registers. This is the direct result of having instructions which can operate directly on memory and the limited amount of chip space not dedicated to instruction decoding, execution, and microcode storage.
- Several special purpose registers. Many CISC designs set special registers for the stack pointer, interrupt handling, and so on.
- A 'Condition code" register which is set as a side-effect of most instructions. This register reflects whether the result of the last operation is less than, equal to, or greater than zero and records if certain error conditions occur.

At the time of their initial development, CISC machines used available technologies to optimize computer performance.

- Microprogramniing is as easy as assembly language to implement, and much less expensive than hardwiring a control unit.
- The ease of microcoding new instructions allowed designers to make CISC machines upwardly compatible: a new computer could run the same programs as earlier computers because the new computer would contain a superset of the instructions of the earlier computers.
- As each instruction became more capable, fewer instructions could be used to implement a given task. This made more efficient use of the relatively slow main memory.
- Because microprogram instruction sets can be written to match the constructs of high-level languages, the compiler does not have to be as complicated.

Complex Instruction Set Computer (CISC) Characteristics

- Major characteristics of a CISC architecture
  - »1) A large number of instructions - typically from 100 to 250 instruction
  - »2) Some instructions that perform specialized tasks and are used infrequently
  - »3) A large variety of addressing modes - typically from 5 to 20 different modes
  - »4) Variable-length instruction formats
  - »5) Instructions that manipulate operands in memory (RISC in register)

What is RISC?

- It stands for **reduced instruction set computer**.
- The main idea behind this is to make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating, and storing operations just like a load command will load data, a store command will store the data.
- RISC processor have ADD, SUB, MUL, LOAD, STORE, AND, OR, EOR, CALL, JUMP, etc instructions.
- Reduce the cycles per instruction at the cost of the number of instructions per program. More than 95% instructions are executed with only one clock cycle.
- RISC processor have a fixed instruction size.
- More general-purpose registers. All RISC architecture have at least 32 registers, so there is no need to store parameters on stack.
- Simpler instruction, hence simple instruction decoding.
- RICS processor have separate busses for data and code.

Reduced Instruction Set Computer (RISC)

- Major characteristics of a RISC architecture
  - »1) Relatively few instructions
  - »2) Relatively few addressing modes
  - »3) Memory access limited to load and store instruction
  - »4) All operations done within the registers of the CPU
  - »5) Fixed-length, easily decoded instruction format
  - »6) Single-cycle instruction execution
  - »7) Hardwired rather than microprogrammed control

– RISC Instruction
  - Only use **LOAD** and **STORE** instruction when communicating between memory and CPU
  - All other instructions are executed within the registers of the CPU without referring to memory

Program to evaluate *X* = *(A + B)* * *(C + D)*

LOAD	**R1, A**	$R1 \leftarrow M[A]$
LOAD	**R2, B**	$R2 \leftarrow M[B]$
LOAD	**R3, C**	$R3 \leftarrow M[C]$
LOAD	**R4, D**	$R4 \leftarrow M[D]$
ADD	**R1, R1, R2**	$R1 \leftarrow R1 + R2$
ADD	**R3, R3, R4**	$R3 \leftarrow R3 + R4$
MUL	**R1, R1, R3**	$R1 \leftarrow R1 * R3$
STORE	**X, R1**	$M[X] \leftarrow R1$

•Load instruction transfers the operand from memory to CPU Register.

•Add and Multiply operations are executed with data in the registers without accessing the memory.

•Result is then stored in the memory with store information.

- Other characteristics of a RISC architecture
  - 1) A relatively large number of registers in the processor unit
  - 2) Use of overlapped register windows to speed-up procedure call and return
  - 3) Efficient instruction pipeline
  - 4) Compiler support for efficient translation of high-level language programs into machine language programs

CISC versus RISC

CISC	RISC
Emphasis on hardware	Emphasis on software
Includes multi-clock complex instructions	Single-clock, reduced instruction only
Memory-to-memory: "LOAD" and "STORE" incorporated in instructions	Register to register: "LOAD" and "STORE" are independent instructions
Small code sizes, high cycles per second	Low cycles per second, large code sizes
Transistors used for storing complex instructions	Spends more transistors on memory registers