How CPU works

~~In~~

CPU - central processing unit

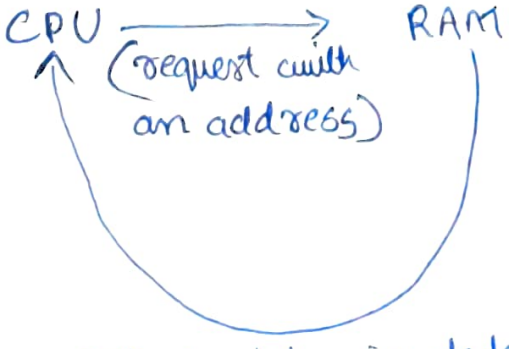
* brain of the computer.
* Consist of register, control unit, arithmetic logic unit (ALU).
* Address bus for ~~transmitting~~ carrying address.
  Data bus for data.

~~CPU is synchronized~~

* Every action in CPU synced by a clock.
* Clock produce a clock signal in range (MHz - GHz)
* Each cycle performs simple operation.

* CPU access data from RAM.
* RAM - Random Access Memory.
* Normal data can be accessed only in a particular order from a memory. But in RAM it can be accessed ~~in any~~ randomly.
* CPU takes data ~~from~~ for current operation from RAM.

* ~~CP~~ How CPU take data from RAM.



Note:

only if enable wire is activated.

Data in RAM: numbers,
addresses,
instruction for CPU,
encoded letters.

* Stored as '1's and '0's.

Instruction Sets for CPU.

1) Load
2) Add
3) Compare
4) control Jumps

} most common.

~~and~~

* These instructions decide on how a CPU perform various actions.

Working of CU and ALU.

* when the data is fetched, it is processed by the ALU based on the commands or ~~in~~ instructions from CU.

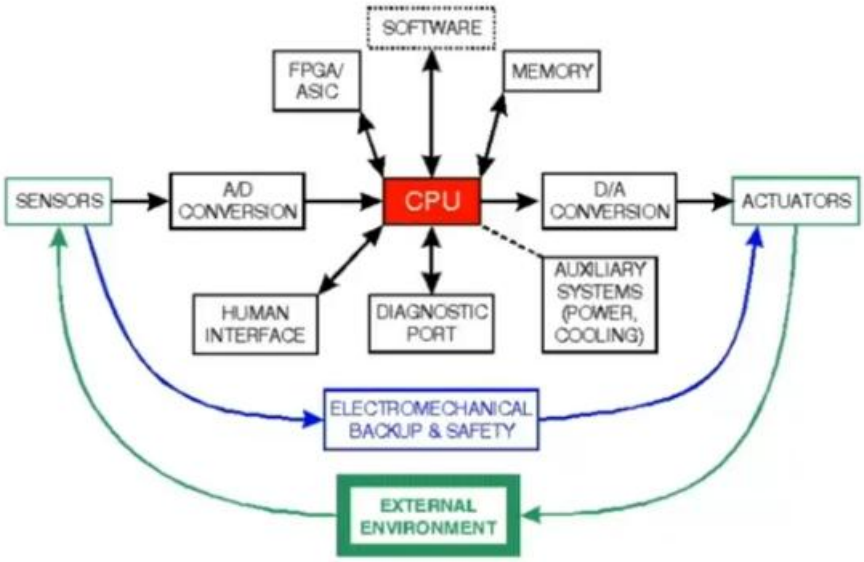
* There are also Flags.

These flags are just wires that become low or high based on some condition. They do not transmit any output but to show any current state.

~~The~~

Hard drives

* When the power is turned off the entire data in ~~RAM~~ RAM is lost since it is an volatile memory.
* ~~But~~ So a hard drive is used for this.
* The movement of the arm ~~in~~ above the magnetic disk is very fast but not as fast as computation done in CPU.
* So first data is loaded to ~~RAM~~ RAM from there CPU access it.



****

To develop an embedded product for an autonomous car that detects objects and takes corrective actions while driving, the requirements must cover the hardware, software, and environmental constraints. Below is a comprehensive list of requirements to guide the selection of a microcontroller for this project.

1. Functional Requirements

1. **Object Detection**
   - Support for interfacing sensors such as LIDAR, ultrasonic, RADAR, and cameras.
   - Real-time image processing capability for object recognition and classification.
2. **Corrective Action**
   - Ability to control actuators for steering, braking, and acceleration.
   - High-speed decision-making to avoid collisions or maintain safe distances.
3. **Communication**
   - Support for CAN, LIN, and Ethernet for communication with other car systems.
   - Ability to interface with GPS and IMU sensors for positioning and orientation.
4. **Fail-Safe Mechanisms**
   - Redundant systems to ensure reliability in case of hardware or software failure.
   - Automatic transition to manual driving in case of system failure.

2. Performance Requirements

1. **Processing Power**
   - High-performance ARM Cortex-M or Cortex-A cores capable of handling complex AI/ML algorithms.
   - Minimum clock speed: 200 MHz.
   - Support for hardware accelerators (e.g., DSP or AI inference engines).
2. **Memory**
   - Flash memory: ≥ 2 MB for program storage.
   - RAM: ≥ 512 KB for real-time processing.
3. **Real-Time Operation**
   - Must support real-time operating systems (RTOS) for deterministic behavior.
   - Low latency for sensor input to action output (≤ 50 ms).
4. **Power Consumption**
   - Optimized power consumption for automotive environments, with sleep modes and low-power states.

3. Hardware Requirements

1. **Interfaces**
   - Multiple UART, SPI, I2C, and GPIOs for connecting peripherals.
   - Support for high-speed data interfaces like USB or PCIe.
2. **Robustness**

    - Temperature range: -40°C to 125°C.
    - Vibration and shock resistance per automotive standards.
3. **Safety Standards Compliance**
    - Compliance with ISO 26262 for functional safety (ASIL-B or higher recommended).
4. **Analog and Digital Input/Output**
    - Support for ADCs and DACs for sensor inputs and control outputs.

4. Software Requirements

1. **Development Tools**
    - Support for standard toolchains like GCC, Keil, IAR, or vendor-specific IDEs.
    - Debugging capabilities with JTAG or SWD.
2. **Connectivity**
    - Support for wireless communication standards like Wi-Fi, Bluetooth, or 5G for over-the-air updates.
3. **AI and Machine Learning**
    - Compatibility with AI frameworks like TensorFlow Lite or ONNX for embedded systems.
    - Hardware or software-based ML acceleration.
4. **Firmware Update**
    - Secure bootloader for over-the-air firmware updates (OTA).
    - Encryption and authentication for updates.

5. Environmental Constraints

1. **Automotive Standards**
    - Must comply with AEC-Q100 for automotive-grade microcontrollers.
    - Electromagnetic compatibility (EMC) and susceptibility (EMS) compliance.
2. **Power Supply**
    - Operate within 12V DC automotive systems with tolerance for voltage spikes.

6. Cost and Scalability

1. **Cost Constraints**
    - Affordable while meeting all performance and safety requirements.
2. **Scalability**
    - Easily scalable for integration into different vehicle models.