Assignment

→ How does CPU works.

The CPU is the primary component of a computer responsible for executing instructions, performing calculations and controlling other s/m processes. Here it uses a 6502 processor as an example and explains how memory plays a crucial role in its operation.

- CPU functions as the brain of computer.
- It executes instructions in a series of cycles controlled by a clock. The clock rate determines how many cycles the CPU can execute per second, with modern CPU's reaching speeds measured in gigahertz.
- The 6502 CPU, a microprocessor from the early 1980's provides a simplified example of CPU operations, offering a basic model.
- The CPU communicates with memory, ie, RAM (Random Access Memory), to store and retrieve data. Memory is organised in cells that can be accessed by addresses.

2024 February

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29		

2024 March

S	M	T	W	T	F	S
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

- The CPU retrieves data from RAM, processes it, and writes result back into memory. In modern systems memory is

connected to the CPU via a bus, a s/m of pathways for data transfer. The bus ensures that data moves efficiently b/w memory and the CPU
- The CPU executes various types of instructions, such as Load, Add, store and compare.
- These instructions allows the CPU to manipulate data stored in memory.
- The ALU (Arithmetic Logic Unit) handles arithmetic & logical operations, while the control unit directs the flow of data and coordinates the execution of instructions.
- Inside the CPU, registers temporarily hold data for fast access during operations. For eg:- when addition is done, the ALU uses registers to store intermediate ~~users~~ results.
- The bus is the s/m of pathways that transfers data b/w the CPU, memory and i/p / o/p devices.
- Buses are essential for efficient data flow & communication across the computer.
- In a guessing game, CPU loads the correct answer into memory, waits for user i/p, compares the guess to correct answer and o/p a result
- when comparing 2 values, the ALU sets flags to indicate the outcome. These flags represent the result of comparison.
- Conditional jump instruction such as Jump if Equal, use these flags to determine whether to change the flow of the prgm.
- After executing an instruction the CPU needs to fetch the next one
- It uses an instruction address

2024						April
S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

2024						May
S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

register to track the location of the next instruction in memory.
- The cpu then fetches the instruction and processes it, continuing the cycle until program finishes.

- The CPU is responsible for executing instructions and processing data within a computer s/m. It relies on control unit, ALU, registers and buses to perform tasks efficiently.
- The interaction b/w the cpu and memory components especially RAM is crucial for the overall performance of the computer.

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

Object Detection

Support for interfacing sensors such as LIDAR, ultrasonic, RADAR, and cameras.

Real-time image processing capability for object recognition and classification.

Corrective Action

Ability to control actuators for steering, braking, and acceleration.

High-speed decision-making to avoid collisions or maintain safe distances.

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.

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

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).

Memory

Flash memory: ≥ 2 MB for program storage.

RAM: ≥ 512 KB for real-time processing.

Real-Time Operation

Must support real-time operating systems (RTOS) for deterministic behavior.

Low latency for sensor input to action output (≤ 50 ms).

Power Consumption

Optimized power consumption for automotive environments, with sleep modes and low-power states.

3. Hardware Requirements

Interfaces

Multiple UART, SPI, I2C, and GPIOs for connecting peripherals.

Support for high-speed data interfaces like USB or PCIe.

Robustness

Temperature range: -40°C to 125°C.

Vibration and shock resistance per automotive standards.

Safety Standards Compliance

Compliance with ISO 26262 for functional safety (ASIL-B or higher recommended).

Analog and Digital Input/Output

Support for ADCs and DACs for sensor inputs and control outputs.

4. Software Requirements

Development Tools

Support for standard toolchains like GCC, Keil, IAR, or vendor-specific IDEs.

Debugging capabilities with JTAG or SWD.

Connectivity

Support for wireless communication standards like Wi-Fi, Bluetooth, or 5G for over-the-air updates.

AI and Machine Learning

Compatibility with AI frameworks like TensorFlow Lite or ONNX for embedded systems.

Hardware or software-based ML acceleration.

Firmware Update

Secure bootloader for over-the-air firmware updates (OTA).

Encryption and authentication for updates.

5. Environmental Constraints

Automotive Standards

Must comply with AEC-Q100 for automotive-grade microcontrollers.

Electromagnetic compatibility (EMC) and susceptibility (EMS) compliance.

Power Supply

Operate within 12V DC automotive systems with tolerance for voltage spikes.

6. Cost and Scalability

Cost Constraints

Affordable while meeting all performance and safety requirements.

Scalability

Easily scalable for integration into different vehicle models.

Ans:RENESAS R-CAR V3H

Advanced Processing Capability:

It supports 7.2 TOPS for AI-based tasks like convolutional neural networks (CNNs) and computer vision, essential for real-time object detection and classification.

Dual Cortex-R7 cores ensure robust real-time operation for safety-critical decision-making.

Sensor and Actuator Support:

Interfaces like MIPI CSI2 allow seamless integration with cameras for object detection.

Dedicated accelerators for optical flow and object detection reduce the computational load on general-purpose cores.

Automotive Communication Interfaces:

Built-in support for CAN FD, FlexRay, and Ethernet AVB facilitates communication with other vehicle systems, meeting requirements for reliable system integration.

Real-Time Performance:

Designed for low-latency operation, which is critical for translating sensor data to corrective actions within ≤50 ms.

Safety and Reliability:

Complies with ASIL-C safety requirements, ensuring functional safety for critical operations.

Robust hardware designed to handle automotive-grade environmental conditions (temperature, vibration, and power spikes).

Development and Scalability:

Supported by a rich development ecosystem, including Linux BSP, hardware debugging tools, and development platforms.

Scalable architecture supports expansion to more complex systems or higher levels of autonomy.