Name: Anandini Sahu
Section: S-101
ID: 2400031855

ECU

CO3 - HA

2 marks

1. Illustrate the key components of the AUTOSAR layered architecture.

AUTOSAR has 4 main layers:
Application layer
Runtime Environment (RTE)
Basic Software (BSW)
Microcontroller (Hardware)
The architecture seperates application logic from hardware for modularity and reusability.

2. Define ECU abstraction and its impoltance.
ECU abstraction layer hides hardware details of the ECU from upper layer hides hardware details of the ECU from up. Enabling hardware independent software and easier reuse across different ECUs.

3. Outline the functional role of the microcontroller abstraction layer (MCAL)
This provides drivers for microcontroller peripherals (ADC, PWM, CAN, etc.) and shields higher layers for hardware specific details.

4. Describe the role of complex drivers (CDD)
Handles proprieatary or complex hardware not supported by standard AUTOSAR modules, ensuring integration without breaking architecture rules.

5. Define Basic Software and its role in AUTOSAR

BSW is the foundational layer providing hardware independent services like comm, memory and OS support to applications.

6. Explain the purpose of separating the application and hardware layers like AUTOSAR.

The separation ensures hardware independence allows software reuse and simplifies integration and maintenance.

7. State two services handled by the communication stack in an ECU

PDU router - routes messages between layers

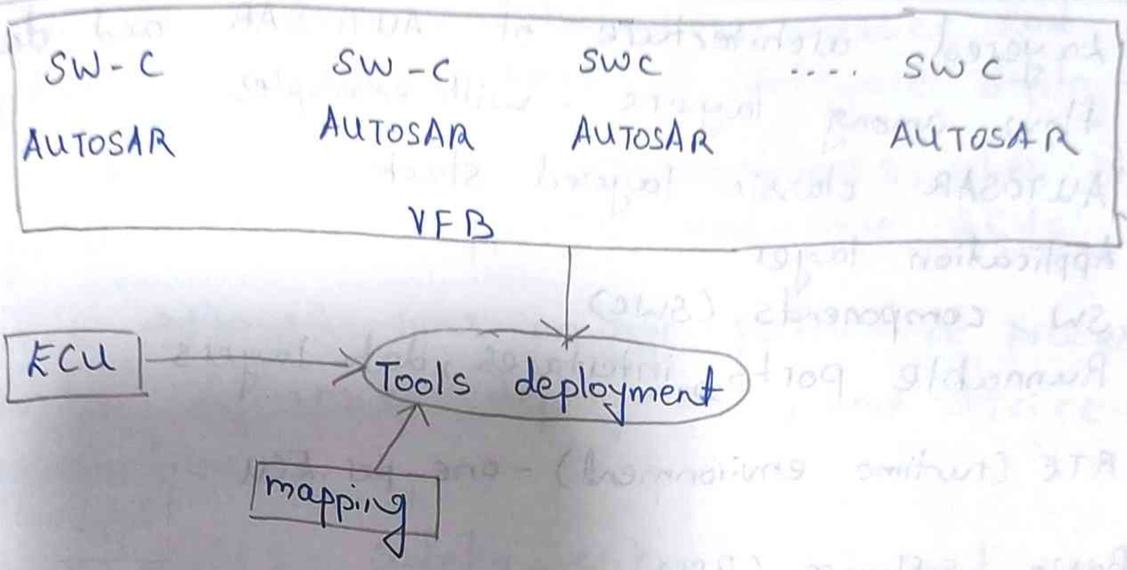
CAN/LIN/FlexRay - manage data transmission and reception.

4 marks

1. Elaborate the virtual function Bus: abstraction & communication between distributed components

VFB - this is a conceptual abstraction in AUTOSAR classic that allows application software components (SW-C's) to communicate with each other and with infrastructure in a hardware and network independence way.

VFB defines logical parts and connects between them, without the application components needing to know where or how the communication is mapped physically.



2. Structure functionality, communication. How among Application layer, RTE and BSW

Application layers contain AUTOSAR software components (SWC) - the functional application logic

RTE : Generated code that implements the VFB on an ECU. Also handles inter-runnable data exchange, timing.

BSW : ECU abstration, microcontroller abstraction, communication stacks.

Functionality :

- SWC
- Define ports (sender-receiver, client-server)
- Contain runnables
- Use port APIs that RTE provides

RTE

- Execute port-level API for SWCs
- Handles scheduling hoots for runnable

BSW :

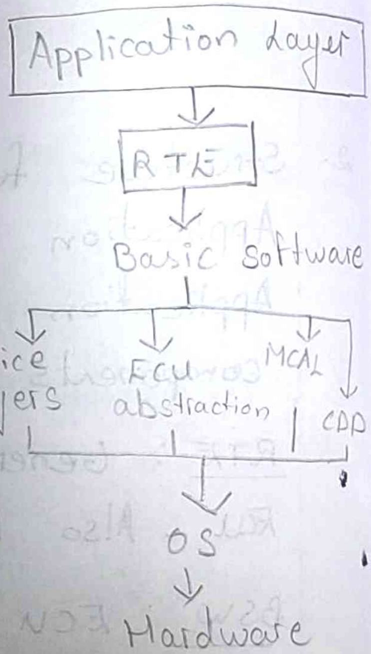
- OST prevent communication system serves. It hides hardware details from the application.

3. Layered architecture of AUTOSAR and data flow among layers - with examples

AUTOSAR classic layered stack

1. Application layer
SW components (SWC)
Runnable ports, interfaces, data layers

2. RTE (runtime environment) - one per ECU



3. Basic Software (BSW)
- Services layers
- ECU abstraction
- MCAL
- CDD

4. Operating System
5. Hardware

eg. Brake pressure sensor value to ABS control over CAN

4. When is a complex device driver (CDD) necessary?
Characterization
CDD is a top of BSW driver implemented when typical MCAL + ECU abstraction cannot meet special functional.
Conditions that make CDD necessary:
1. Time determinism:
when you must implement control loops of processing that require tighter timing.

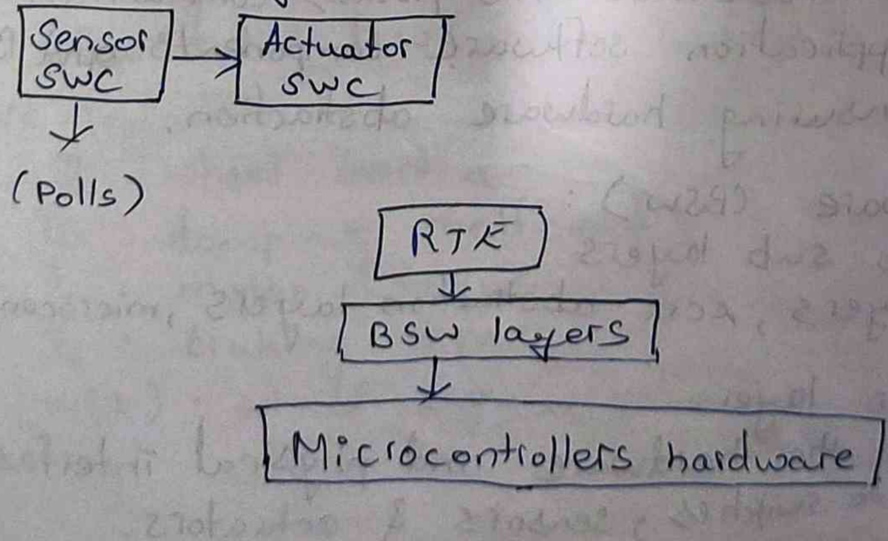
5. Explain the differences between classic and adaptive AUTOSAR in terms of software deployment.

- Classic AUTOSAR targets microcontrollers, uses static configuration, and supports real-time ECUs.
- Adaptive AUTOSAR runs on high-performance processors, supports dynamic deployment, and service-oriented communication.
- Classic suits safety-critical embedded systems; Adaptive suits flexible, updateable systems like autonomous driving.

6. Demonstrate data exchange through RTE in an AUTOSAR ECU.

In AUTOSAR, the RTE acts as a middleware that connects application software with
- Basic software
- other ECUs

Layers involved in data exchange
Application layer



Example:

1. The sensor SWC periodically measures the temperature and write the value to its o/p port using RTE.

2. The RTE is responsible for transferring the data from the sender's output port to the receiver's input port.

7. Apply AUTOSAR layered Architecture principles to design a software structure for a Body control ECU.

Applying AUTOSAR architecture to design a Body control ECU (BCE) includes organizing software into standardized layers to ensure modularity scalability and reusability.

1. Application Layer:
   - contains application software components for body control functions such as central locking, interior lighting
   - each component communicates via AUTOSAR interfaces.

2. Runtime Environment (RTE):
   Acts as a middleware that provides communication between application software components and BSW modules, ensuring hardware abstraction.

3. Basic Software (BSW):
   Divided into sub layers
   service layers, ECU abstraction layers, microcontroller

4. Microcontroller layer:
   Consists of the hardware and physical interfaces connected to switches, sensors & actuators.

11 marks

1.A. A purpose an AUTOSAR -compliant solution for integrating a new radar sensor into an ECU.

B. Apply a model-based Content strategy for an electric braking system using PID and adaptive control

Dataflow.

A. Radar sensor -> physical bus -> transceiver /PHY -> MCAL driver -> ECU

Abstraction -> PDU router /communication services ->

RTE -> SWC

perception SWC, -> application logic

Decide communication transport & top level approach. If radar use CAN/CAN-FD: use CAN interface. If radar uses ethernet.

B. An electronic braking systems converts the kinetic energy of an electric vehicle into electrical energy

system model :

The braking systems dynamics can be modeled as

$$J\omega(t) + B\omega(t) = T_m(t) - T_b(t)$$

where :

J : wheel inertia

B : damping coefficient

T_m : motor torque

T_b : braking torque

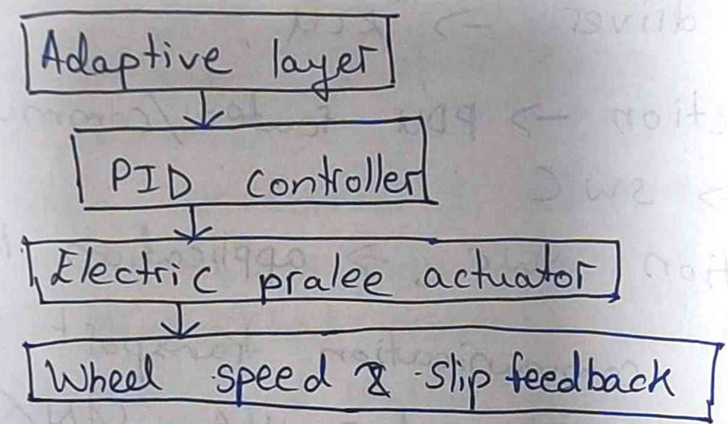
$\omega(t)$: wheel angular velocity

PID controller : proportional - integral - derivative controller provides baseline control for torque regulation.

$$T_b(t) = k_p e(t) + k_i \int e(t)\,dt + k_d \frac{de(t)}{dt}$$

proportional (k_p): reacts to skip error instantly.

Control structure :



2.

A. Develop a conceptual ECU architecture using AUTOSAR principles for an EV battery management system.

B. Illustrate major software layer instructions emphasizing data exchange through RTE and BSW.

A. A battery management (BMS) ECU in an electric vehicle monitors and controls battery parameters such as voltage, current, temperature and state of charge (SOC)

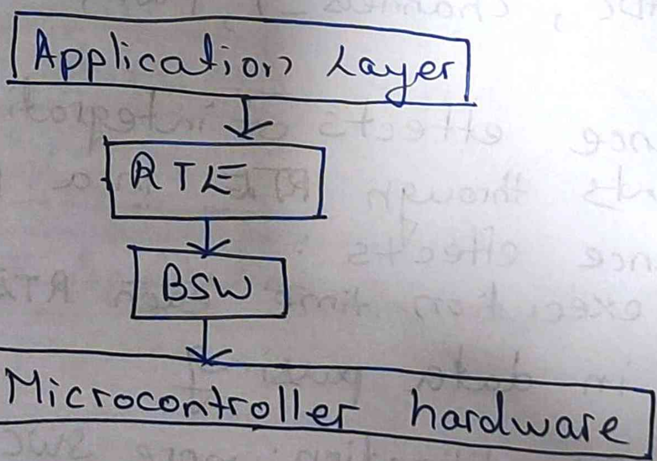
Using AUTOSAR architecture the software is divided into well defined layers that separate application services and hardware.

B. Sensor input
1. Temp. sensor sends analog signal to ADC
MCAL converts it into a digital value.

2. ECU abstraction + service layer (BSW)
ECU abstraction layer passes the data to higher layer using standardized APIs

3. RTE (Runtime environment)
RTE transfers this temperature data to the thermal management SWC in the application layer.

4. Application layer:
sends control command back through RTE -> BSW -> MCAL -> actuator



3 A. Demonstrate Parameter monitoring between application software components and BSW module through configuration files.

In AUTOSAR, application software components (SWC) don't directly access hardware of low-level devices. Instead communication b/w SWCs and BSW happens via the RTE.

Parameters:

→ Application parameters
Defined inside each software component eg. voltage, speed

→ BSW module parameters
Defined for hardware related modules such as ADC, PWM, CAN etc
eg. ADC, Channels_1, PWM

B. Performance effects of integrating multiple software components through RTE in a powertrain ECU.

Performance effects:

- Longer execution time: each RTE call adds small delays in data passing
- Higher CPU utilization: more SWCs = more scheduling and communication events.
- Reduced determinism: Task timing becomes less predictable.

4.A. Illustrate the signal flow through the RTE showing interaction between application and Basic software modules.

B. Apply AUTOSAR design to construct a software structure for a vehicle body control ECU.

A. A signal flow through the RTE (application <-> Basic software)

For a sensor -> application -> network message and a received network frame -> Basic software -> application (Rx).

The RTE is the middleware that virtualizes communication for SWC and provides access to BSW services.

Sensor HW

↓

MCAL / ECU abstraction

↓

Basic software

↓

RTE <-> Application SWC(s)

↕

COM

B. A practical sofware design for a body control ECU (BCU) - typical functions : lighting, doors, windows, central locking, wipers, comfort features, keyless entry, interior lighting and network gatewaying

- BC lighting
  ports : light cmd in (SR), lightstatus (SR)
  (CS for dim/flash)
- Runnables : Blink control
- BC - Door control
  ports : Door switch, look cmd_in, Door status_out
  runnables : Door statemachine
- BC - windowControl
  ports : window Switch_in, windowMotor_out (actuator)
  runnables : antipincheck, moviewindow
- BC - centralLock
  ports : Lock service, Lock state_out

5. A Develop a fault reporting mechanism within an AUTOSAR ECU for a critical sensor failure.

- Modeling: Define DemEventParameter (eg. EventId_SensorCritical) mapped to a DTC (eg. 0x123456) with debounding, healing event memory and NvM persistence.
- Runnables: In a SensorMonitor SWC, cyclic runnable reads via IoHwAb (or CDD) and validates (range, rate-of-change, Plausibility, timeout).

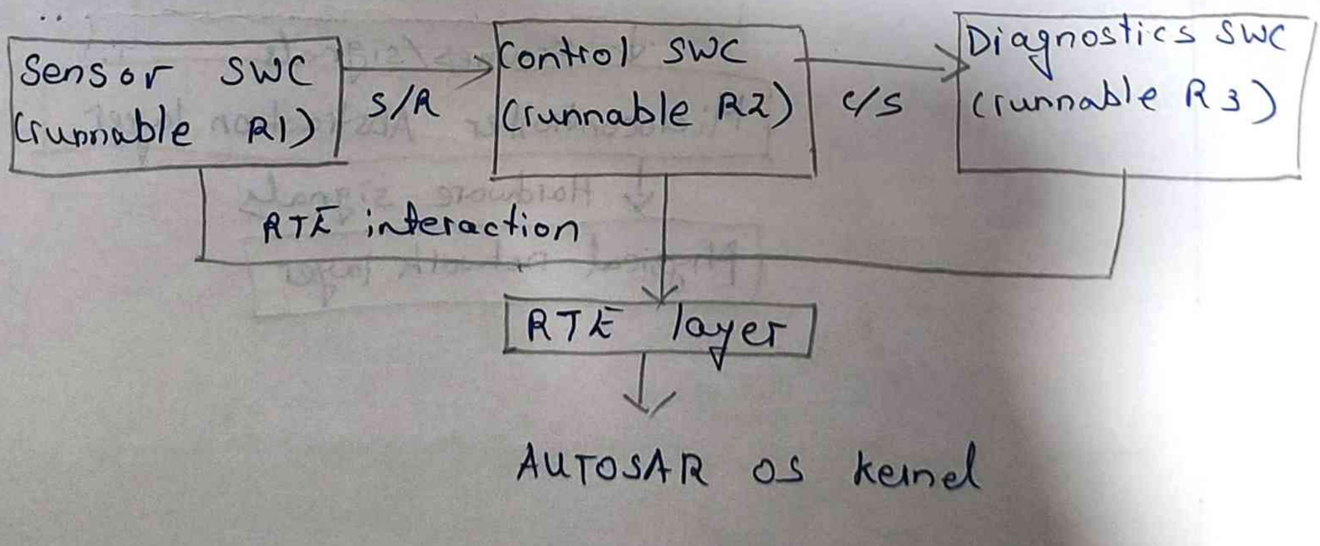
- Report : On fault, call Dem_ReportErrorStatus (EventId_SensorCritical, DEM_EVENT_STATUS_FAILED); on recovery, report PASSED.

- System reaction : BswM rule reacts to Dem event (enter degraded / safe mode, clamp outputs, switch to fallback sensor, raise waring lamp.

8. Demonstrate perforance effects of integrating multiple components through the RTE.

- Call sementics overhead :
client - server (sync) adds RTE marshaling + OS context switch; caller blocks → latency and potential priority inversion if callee's task is lower priority.

- Activation & scheduling : More runnables / events → more tasks activations, context switches and jitter, WCET budgets must include RTE glue + COM proxies.

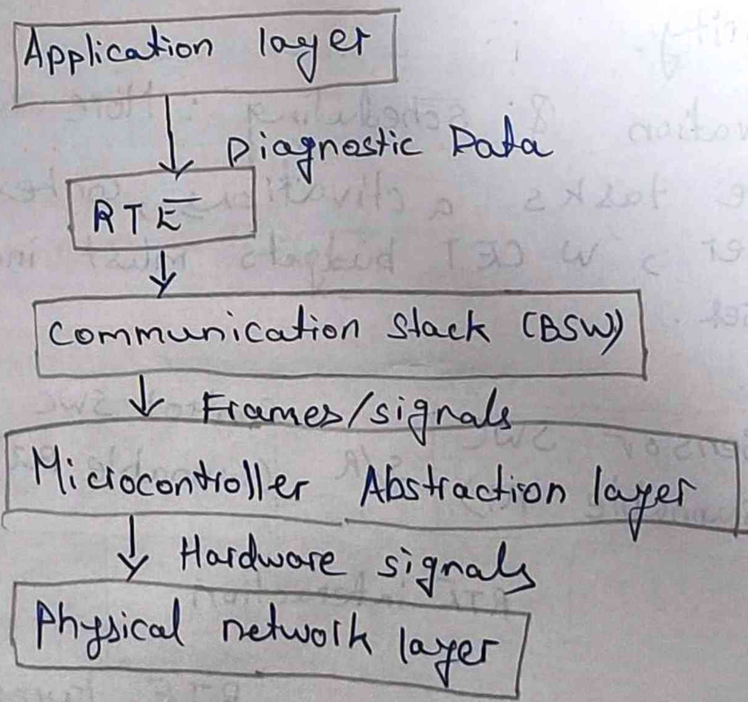


6. A. Describe Diagonstic Message Handling via AUTOSAR communication stack.

B. Explain integration testing methods for AUTOSAR-based ECUs.

A. Diagnostic messages in AUTOSAR are used for fault detection, reporting and troubleshooting. They are managed through the AUTOSAR communication stack, which includes several figures that handle message flow between the ECU and diagnostic tools like OBD.

Flow of Diagnostic Message Handling:
Diagnostic Tool → ECU



B. Integration testing involves that all AUTOSAR software layers (application, RTE and BSW) work together correctly on the ECU.

1. component integration testing:
Tests how application software components (SWCs) interact via the RTE.
eg. checking that a sensor SWC correctly sends data to an actuator SWC.

2. RTE integration testing:
verifies the RTE connections b/w SWCs and BSW modules work properly
Focus on signal flow, timing and data exchange.

Integration Testing Levels

- Unit Testing (SWC modules)
- Component Test (RTE & BSW)
- Integration of ECUs
- System Testing (Vehicle level)

7. A Demonstrating how the event would be detected, debounced, stored by DEM and made accessible to an internal diagonstic tester via DCM.

B. Demonstrate the performance effects of integrating multiple software components through the RTE in a power train ECU and propose methods to improve its real-time execution efficiency.

A.

Sensor (ABS wheel)

↓

Sensor interfaces (MCAL)

↓

CDD / BSW module

↓

DEM (Event manager)

↓

DCM (diagnostic comm)

↓

External tester

To detect when a critical software fails, record the fault and allow a diagnostic tool.

Step-by step flow:

Sensor monitoring

Debouncing

Reporting to DEM

Access by diagnostic tester

Clearing the fault

B.

Application SW Cs (Engine, Transmission)

↓

RTE

↓

BSW

↓

Hardware / Sensors / CAN

Effects on performance:

- Increased CPU load :- each RTE message takes processing time.
- Delay in signal transfer :- Data goes through RTE layers, adding small delays.
- Higher memory usage :- RTE stores buffer for signals & communication
- Scheduling overhead : — More tasks → more content switching and timing issues.

9/10