MODULE 2: Characteristics and quality attributes of embedded systems: Characteristics, Operational and non-operational quality attributes, application specific embedded system - washing machine, domain specific – automotive

Characteristics of Embedded Systems

Embedded systems possess certain specific characteristics. These characteristics are unique to each embedded system. Some of the important characteristics of an embedded system are:

1. Application and domain specific
2. Reactive and Real Time
3. Operates in harsh environments
4. Distributed
5. Small size and weight
6. Power concerns

1. Application and domain specific

Each embedded system has certain functions to perform and they are developed in such a manner to do the intended functions only. They cannot be used for any other purpose. For example, the embedded control unit of a microwave oven cannot be replaced with an air conditioner's embedded control unit, because the embedded control units of microwave oven and air conditioner are specifically designed to perform certain specific tasks.

Similarly, an embedded control unit developed for a particular domain, say telecom, cannot be replaced with another control unit designed to serve another domain like consumer electronics.

2. Reactive and Real Time

Embedded systems are in constant interaction with the real world through sensors and user defined input devices. Any changes happening in the Real world (which is called an Event) are captured by the sensors or input devices in Real Time and the control algorithm running inside the unit reacts in a designed manner to bring the controlled output variables to the desired level.

Embedded systems produce changes in output in response to the changes in the input. So they are generally referred as Reactive Systems.

Real Time System operation means the timing behaviour of the system should be deterministic. The system should respond to requests or tasks in a known amount of time.

A Real Time system should not miss any deadlines for tasks or operations.

Embedded applications or systems which are mission critical, like flight control systems, Antilock Brake Systems (ABS), etc. are examples of Real Time systems.

3. Operates in harsh environments

The environment in which the embedded system deployed may be a dusty one or a high temperature zone or an area subject to vibrations and shock. Systems placed in such areas should be capable to withstand all these adverse operating conditions. The design should take care of the operating conditions of the area where the system is going to implement.

For example, if the system needs to be deployed in a high temperature zone, then all the components used in the system should be of high temperature grade. Power supply fluctuations, corrosion and component aging, etc. are the other factors that need to be taken into consideration for embedded systems to work in harsh environments.

4. Distributed

“Distributed” means that embedded systems may be a part of larger systems. Many numbers of such distributed embedded systems form a single large embedded control unit. For example, a car has many embedded systems controlled to its dash board. Each one is an independent embedded system yet the entire car can be said to function properly only if all the systems work together.

Another example is Automatic Teller Machine (ATM). It contains a card reader embedded unit, responsible for reading and validating the user's ATM card, transaction unit for performing transactions, a currency counter for dispatching/vending currency to the authorized person and a printer unit for printing the transaction details.

5. Small size and weight

An embedded system that is compact in size and has light weight will be desirable or more popular than one that is bulky and heavy. For example: Cell phones. The cell phones that have the maximum features are popular but also their size and weight is an important characteristic.

People believe in the phrase "Small is beautiful". Moreover, it is convenient to handle a compact device than a bulky product. In embedded domain, compactness is a significant deciding factor. Most of the application demands small sized and low weight products.

6. Power concerns

Power management is another important factor that needs to be considered in designing embedded systems. Embedded systems should be designed in such a way as to minimize the heat dissipation by the system. The production of high amount of heat demands cooling requirements like cooling fans which in turn occupies additional space and make the system bulky. Also power management is a critical constraint in battery operated application. The more the power consumption the less is the battery life.

Quality Attributes of Embedded Systems

Quality attributes are the non-functional requirements that need to be documented properly in any system design. If the quality attributes are more concrete and measurable it will give a positive impact on the system development process and the end product. The quality attributes in any embedded system development are broadly classified into two:

- Operational Quality Attributes
- Non-Operational Quality Attributes

Operational Quality Attributes

The operational quality attributes represent the relevant quality attributes related to the embedded system when it is in the operational mode or 'online' mode. The important operational quality attributes are:

1. Response
2. Throughput
3. Reliability
4. Maintainability
5. Security
6. Safety

1. Response

Response is a measure of quickness of the system. It gives you an idea about how fast the system is tracking the changes in input variables. Most of the embedded systems demand fast response which should be almost Real Time. For example, an embedded system deployed in flight control application should respond in a Real Time manner. Any response delay, may cause potential damages to the safety of the flight as well as the passengers.

It is not necessary that all embedded systems should be Real Time in response. For example, the response time requirement for an electronic toy is not at all time-critical. There is no specific deadline that this system should respond within this particular timeline.

2. Throughput

Throughput deals with the efficiency of a system. Throughput can be defined as the rate of production or operation of a defined process over a stated period of time. The rates can be expressed in terms of units of products, batches produced, or any other meaningful measurements. In the case of a Card Reader, throughput means how many transactions the Reader can perform in a minute or in an hour or in a day.

Throughput is generally measured in terms of 'Benchmark'. A 'Benchmark' is a reference point by which something can be measured. Benchmark can be a set of performance criteria that a product is expected to meet or a standard product that can be used for comparing other products of the same product line.

3. Reliability

Reliability is a measure of how much percentage you can rely upon the proper functioning of the system or what is the percentage susceptibility of the system to failures. System reliability is defined using two terms:

- Mean Time Between Failures (MTBF): It gives the frequency of failures in hours /weeks /months.
- Mean Time To Repair (MTTR): Specifies how long the system is allowed to be out of order following a failure. For an embedded system with critical application need, it should be of the order of minutes.

4. Maintainability

Maintainability deals with support and maintenance to the end user or client in case of technical issues and product failures or on the basis of a routine system check-up. Reliability and maintainability are considered as two complementary disciplines. A more reliable system means a system with less corrective maintainability requirements and vice versa.

Maintainability can be broadly classified into two categories:

- Scheduled or Periodic Maintenance (preventive maintenance): For example, replacing the cartridge of a printer after each 'n' number of printouts to get quality prints.
- Maintenance to unexpected failures (corrective maintenance): For example, repairing the printer if the paper feeding part fails.

Maintainability is also an indication of the availability of the product for use. In any embedded system design, the ideal value for availability is expressed as

$Ai = MTBF / (MTBF + MTTR)$, where

Ai = Availability in the ideal condition, MTBF = Mean Time Between Failures & MTTR = Mean Time To Repair.

5. Security

Three major measures of information security are: Confidentiality, Integrity & Availability. Confidentiality deals with the protection of data and application from unauthorized disclosure. Integrity deals with the protection of data and application from unauthorized modification. Availability deals with protection of data and application from unauthorized users.

Example of the 'Security' aspect in an embedded product is a Personal Digital Assistant (PDA). If it is a shared one there should be some mechanism in the form of a user name and password to access into a particular person's profile-this is an example of 'Availability'. Also, all data and applications present in the PDA need not be accessible to all users. Some of them are specifically accessible to administrators only-an example of Confidentiality. Some data present in the PDA may be visible to all users but there may not be necessary permissions to alter the data by the users-an example of Integrity.

Non-Operational Quality Attributes

The quality attributes that needs to be addressed for the product 'not' on the basis of operational aspects are grouped under this category. The important non-operational quality attributes are:

1. Testability & Debugability
2. Evolvability
3. Portability
4. Time-to-prototype and market
5. Per unit and total cost

1. Testability & Debugability

Testability deals with how easily one can test his/her design, application and by which means he/she can test it. Testability is applicable to both the embedded hardware and firmware. Embedded hardware testing ensures that the peripherals and the total hardware functions in the desired manner. Firmware testing ensures that the firmware is functioning in the expected way.

Debug-ability is a means of debugging the product as such for figuring out the

probable sources that create unexpected behaviour in the total system. Debug-ability has two aspects in the embedded system development context:
Hardware level debugging and firmware level debugging.

Hardware debugging is used for figuring out the issues created by hardware problems. Firmware debugging is employed to figure out the probable errors that appear as a result of flaws in the firmware.

2. Evolvability

Evolvability is a term which is closely related to Biology. *Evolvability* is referred as the non-heritable variation. For an embedded system, the quality attribute 'Evolvability' refers to the ease with which the embedded product (including firmware and hardware) can be modified to take advantage of new firmware or hardware technologies.

3. Portability

Portability is measure of “system Independence”. An embedded product is said to be portable if the product is capable of functioning 'as such' in various environments, target processors/ controllers and embedded operating systems. A standard embedded product should always be flexible and portable. In embedded products, 'porting' represents the migration of the embedded firmware written for one target processor (i.e., Intel x86) to a different target processor (say ARM Cortex M3 processor). If the firmware is written in a high level language like ‘C’, it is very easy to port the firmware. If the firmware is written in Assembly Language for a particular family of processor (say x86 family), then the portability is poor.

4. Time-to-prototype and market

Time to market is the time elapsed between the conceptualization of a product and the time at which the product is ready for selling (for commercial product) or use (for non-commercial products). The commercial embedded product market is highly competitive and time-to-market the product is a critical factor in the success of a commercial embedded product. If a new design takes long time to develop and market it, then competitor might release their product before you do and/or technology used might have superseded with a new technology.

Product prototyping helps a lot in reducing time-to-market. Prototyping is an informal kind of rapid product development in which the important features of the product under consideration are developed. The time to prototype is also another critical factor. If the prototype is developed faster, the actual estimated development time can be brought down significantly. In order to shorten the time to prototype, make use of all possible options like the use of off-the-shelf components, re-usable assets, etc.

5. Per unit and total cost

Cost is a factor which is closely monitored by both end user (who buy the product) and product manufacturer (who build the product). Cost is a highly sensitive factor for commercial products. Any failure to position the cost of a commercial product at a nominal rate, may lead to the failure of the product in the market. Proper market study and cost benefit analysis should be carried out before taking a decision on the per-unit cost of the embedded product. From a designer/ product development company perspective the ultimate aim of a product is to generate marginal profit. So the budget and total system cost should be properly balanced to provide a marginal profit.

The Product Life Cycle (PLC)

The product life cycle of every embedded product has different phases:

Design and Development Phase: The product idea generation, prototyping, Roadmap definition, actual product design and development are the activities carried out during this phase. There is only investment and no returns.

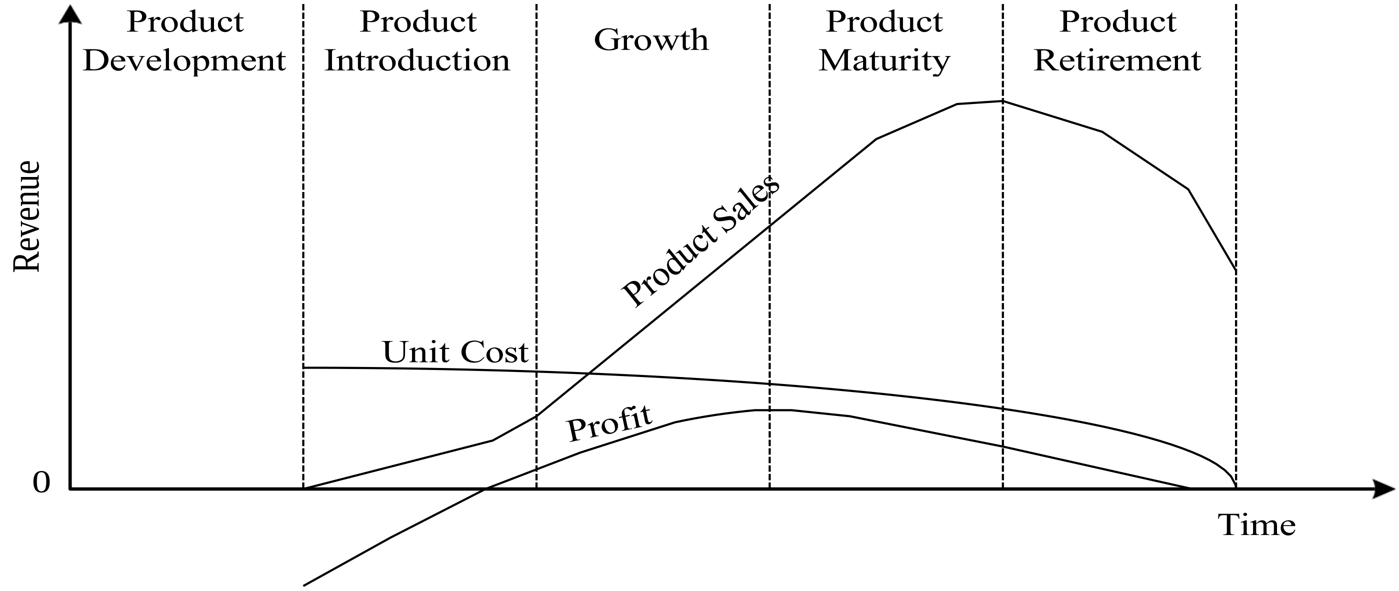
Product Introduction Phase: Once the product is ready to sell, it is introduced to the market. During the initial period the sales and revenue will be low. There won't be much competition and the product sales and revenue increases with time.

Growth Phase: The product grabs high market share.

Maturity Phase: Growth & sales will be steady and the revenue reaches at its peak.

Product Retirement/Decline Phase: Drop in sales volume, market share and revenue. The decline happens due to various reasons like competition from similar product with enhanced features or technology changes, etc. At some point of the decline stage, the manufacturer announces discontinuing of the product.

The different stages of the embedded products life cycle are represented in the below Product Life-cycle graph. From the graph, the total revenue increases from the product introduction stage to the product maturity stage. The revenue peaks at the maturity stage and starts falling in the decline/retirement Stage. The unit cost is very high during the introductory stage. The profit increases with increase in sales and attains a steady value and then falls with a dip in sales.



Application-Specific Embedded System Washing Machine

An embedded system contains sensors, actuators, control unit and application - specific user interfaces like keyboards, display units, etc. All these components can be seen in a washing machine. Some of them are visible and some of them may be invisible.

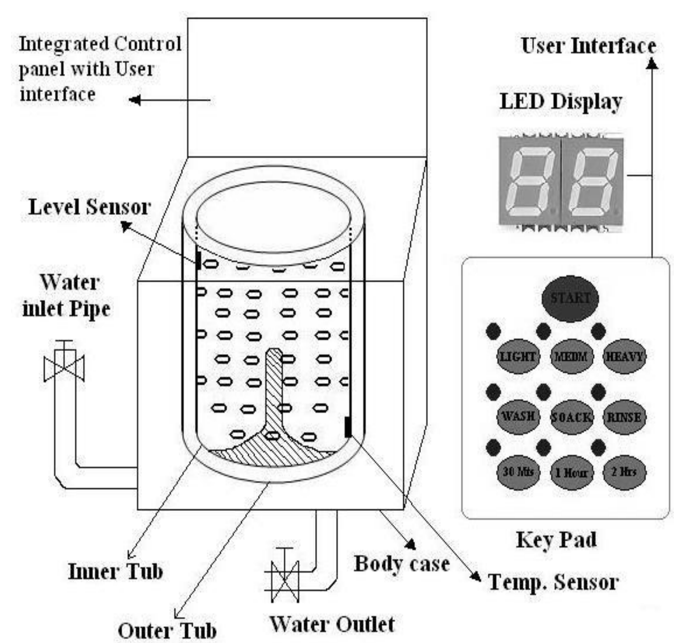
The major Parts of Washing Machine are:

Actuator part: consists of a motorized agitator, tumble tub, water drawing pump and inlet valve to control the flow of water into the unit.

Sensor part: consists of the water temperature sensor, level sensor, etc. The sensor data is fed back to the control unit and the control unit generates the necessary actuator outputs.

Control part: contains a microprocessor/controller based board with interfaces to the sensors and actuators. The control unit also provides connectivity to user interfaces like keypad for setting the washing time, selecting the type of material to be washed like light, medium, heavy duty, etc.

User feedback is reflected through the display unit and LEDs connected to the control board.



Washing machine comes in two models: Top loading & Front loading machines.

First phase of washing: The agitator of the machine twists back and forth and pulls the cloth down to the bottom of the tub. On reaching the bottom of the tub the clothes work their way back up to the top of the tub where the agitator grabs them again and repeats the mechanism.

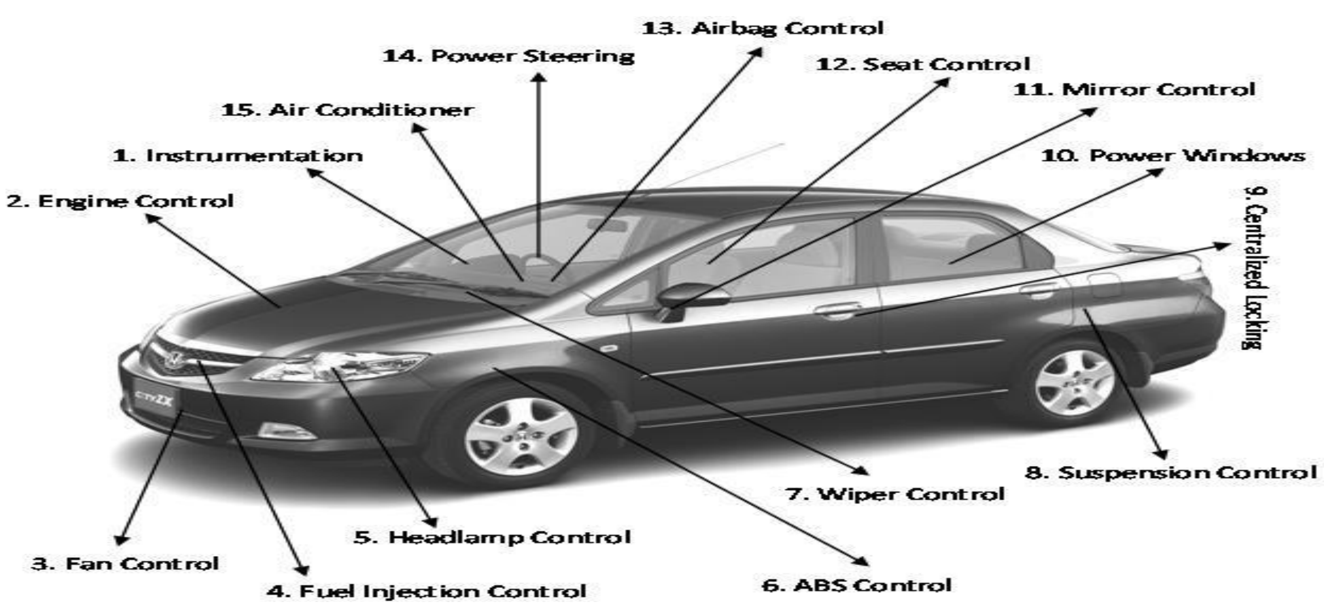
Second phase of washing: Water is pumped out from the tub and the inner tub uses centrifugal force to wring out more water from the clothes by spinning at several hundred Rotations Per Minute (RPM). This is called a 'Spin Phase'.

The inner tub of the machine contains a number of holes and during the spin cycle the inner tub spins, and forces the water out through these holes to the stationary outer tub from which it is drained off through the outlet pipe.

The basic controls consist of a timer, cycle selector mechanism, water temperature selector, load size selector and start button. The integrated control panel consists of a microprocessor / controller based board with I/O interfaces and a control algorithm running in it.

Domain Specific Embedded System- Automotive

Automotive industry is one of the major application domains of embedded systems. Automotive embedded systems are the one where electronics take control over the mechanical system. Ex. Simple viper control. Presence of automotive embedded system in a vehicle varies from simple mirror and wiper controls to complex air bag controller and antilock brake systems (ABS).



Automotive embedded systems are normally built around microcontrollers or DSPs or a hybrid of the two and are generally known as Electronic Control Units (ECUs). The number of embedded controllers in an ordinary vehicle varies from 20 to 40 whereas a luxury vehicle like Mercedes S and BMW 7 may contain 75 to 100 numbers of embedded controllers.

The electronic control units (ECUs) used in the automotive embedded industry can be broadly classified into two:

High-speed Electronic Control Units (HECUs): These are deployed in critical control units requiring fast response. They include fuel injection systems, antilock brake systems, engine control, electronic throttle, steering controls, transmission control and central control unit.

Low-speed Electronic Control Units (LECUs): These are deployed in applications where response time is not so critical. They generally are built around low cost microprocessors /microcontrollers and digital signal processors. Audio controllers, passenger and driver door locks, door glass controls (power windows), wiper control, mirror control, seat control systems, head lamp controls, sun roof control unit etc. are examples of LECUs.

Serial Interface Buses in Automotive Applications

Automotive applications make use of serial buses for communication, which greatly reduces the amount of wiring required inside a vehicle. Different types of serial interface buses are:

- Controller Area Network (CAN) Bus
- Local Interconnect Network (LIN) Bus
- Media-Oriented System Transport (MOST) Bus

Controller Area Network (CAN) Bus: CAN Bus was originally proposed by Robert Bosch. It supports medium speed (ISO11519-class B with data rates up to 125 Kbps) and high speed (IS011898 class C with data rates up to 1 Mbps) data transfer. CAN is an event-driven protocol interface with support for error handling in data transmission. It is generally employed in safety system like airbag control; power train systems like engine control and Antilock Brake System (ABS); and navigation systems like GPS.

Local Interconnect Network (LIN) Bus:

LIN bus is a single master multiple slave (up to 16 independent slave nodes) communication interface. LIN is a low speed, single wire communication interface with support for data rates up to 20 Kbps and is used for sensor/actuator interfacing. LIN bus follows the master communication triggering technique to eliminate the possible bus arbitration problem that can occur by the simultaneous talking of different slave nodes connected to a single interface bus. LIN bus is employed in applications like mirror controls, fan controls, seat positioning controls, window controls, and position controls where response time is not a critical issue.

Media-Oriented System Transport (MOST) Bus: MOST is targeted for automotive audio/ video equipment interfacing, used primarily in European cars. MOST bus is a multimedia fiber-optic, point-to-point network implemented in a star, ring or daisy-chained topology over optical fiber cables. MOST bus specifications define the physical (electrical and optical parameters) layer as well as the application layer, network layer, and media access control. MOST bus is an optical fiber cable connected between the Electrical Optical Converter (EOC) and Optical Electrical Converter (OEC), which would translate into the optical cable MOST bus.

Key Players of the Automotive Embedded Market

The key players of the automotive embedded market can be visualized in three verticals namely: Silicon providers, Tools & Platform providers and Solution providers.

Silicon Providers: They are responsible for providing the necessary chips which are used in the control application development. Chip may be a standard product like microcontroller or DSP or ADC/DAC chips. Some applications may require specific chips and they are manufactured as Application Specific Integrated Chip (ASIC).

The leading silicon providers in the automotive industry are Analog Devices, Xilinx, Atmel, Maxim/Dallas, NXP Semiconductors, Texas Instruments, Fujitsu, NEC, etc.

Tools and Platform Providers: They are manufacturers and suppliers of various kinds of development tools and Real Time Embedded Operating Systems for developing and debugging different control unit related applications.

Some of the leading suppliers of tools and platforms in automotive embedded applications are ENEA, The MathWorks, MATLAB, Keil Software, ARTiSAN, Microsoft, etc.

Solution Providers: They supply Original Equipment Manufacturer (OEM) and complete solution for automotive applications making use of the chips, platforms and different development tools.

The major players of this domain Bosch Automotive, DENSO Automotive, Infosys Technologies, Delphi, etc.