**Embedded systems**

Embedded systems are the electronically controlled devices where software and hardware are tightly coupled. Embedded systems may contain a variety of computing devices. These are PCs incorporated in other devices to operate application-specific functions. The end user usually is not even aware of their existence.

**Embedded Testing**

**EMBEDDED TESTING** is checking the functional and non-functional attributes of both software and hardware in an embedded system. The purpose of Embedded test is to verify and validate the Embedded software as well as hardware against client requirement.

Embedded Software testing checks and ensure the concerned software is of good quality and complies with all the requirements it should meet. Embedded software testing is an excellent approach to guarantee security in critical applications like medical equipment, railways, aviation, vehicle industry, etc. Strict and careful testing is crucial to grant software certification.

**Embedded Software Testing**

* To find bugs in software
* Helps to reduce risk to both users and the company
* Cut down development and maintenance costs
* To improve performance

In Embedded Testing, the following activities are performed:

1. The software is provided with some inputs.

2. A Piece of the software is executed.

3. The software state is observed, and the outputs are checked for expected properties like whether the output matches the expected outcome, conformance to the requirements and absence of system crashes.

**Embedded Software Testing Types**

Fundamentally, there are five levels of testing that can be applied to embedded software

**Software Unit Testing**

The unit module is either a function or class. Unit Testing is performed by the development team, primarily the developer and is usually carried out in a peer-review model. Based on the specification of the module test cases are developed.

**Integration Testing**

Integration testing can be classified into two segments:

1. Software integration testing
2. Software/hardware integration testing.

In the end, the interaction of the hardware domain and software components are tested. This can incorporate examining the interaction between built-in peripheral devices and software.

Embedded software development has a unique characteristic which focuses on the actual environment, in which the software is run, is generally created in parallel with the software. This causes inconvenience for testing since comprehensive testing cannot be performed in a simulated condition.

**System Unit Testing**

Now the module to be tested is a full framework that consists of complete software code additionally all real-time operating system (RTOS) and platform-related pieces such as interrupts, tasking mechanisms, communications and so on. The Point of Control protocol is not anymore a call to a function or a method invocation, but rather a message sent/got utilizing the RTOS message queues.

System resources are observed to evaluate the system's ability to support embedded system execution. For this aspect, gray-box testing is the favored testing method. Depending on the organization, system unit testing is either the duty of the developer or a dedicated system integration team.

**System Integration Testing**

The module to be tested begins from a set of components within a single node. The Points of Control and Observations (PCOs) are a mix of network related communication protocols and RTOS, such as network messages and RTOS events. Additionally to a component, a Virtual Tester can likewise play the role of a node.

**System Validation Testing**

The module to be tested is a subsystem with a complete implementation or the complete embedded system. The objective of this final test is to meet external entity functional requirements. Note that an external entity either be a person, or a device in a telecom network, or both.

**Difference: Embedded testing and Software Testing**

|  |  |
| --- | --- |
| **Software Testing** | **Embedded Testing** |
| Software testing is related to software only. | Embedded testing is related to both software as well as hardware. |
| On average 90% testing done in the world is purely manual black box testing. | Embedded testing is done on embedded systems or chips it can be a black box or white box testing. |
| Primary areas of testing are GUI checks, functionality, validation and some level of database testing. | Primary areas of testing are the behavior of the hardware for the no. of inputs given to it. |
| Software testing is majorly performed on client-server, web and mobile based applications. | Embedded testing generally performed on the Hardware. |
| e.g., Google Mail, Yahoo Mail, Android applications. | e.g., Machines of healthcare domain, Microcontrollers used in computers. |

**Challenges: Embedded Software Testing**

Some of the challenges that one can face during Embedded software testing:

**Hardware Dependency**

Hardware dependency is among the main difficulties faced during embedded software testing because of limited access to hardware. However, Emulators and Simulators may not precisely represent the behavior of the actual device and could give a wrong sense of system performance and application's usability.

**Open Source Software**

The majority of the embedded software components are open source in nature, not created in-house and absence of complete test available for it. There is a wide range of test combinations and resulting scenarios.

**Software vs. Hardware Defects**

Another aspect is when software is being developed for a freshly created hardware, during this process high ratio of hardware defects can be identified. The found defect is just not limited to software. It may be related to hardware also.

**Reproducible Defects**

Defects are harder to reproduce/recreate in the case of the embedded system. That enforces the embedded testing procedure to value every defect occurrence substantially higher than in a standard case, other than to gather as much data as could sensibly be required to alter the system to find the foundation of the defect.

**Continuous Software Updates**

Embedded systems require regular software updates like the kernel upgrade, security fixes, different device drivers, etc. Constraints identified with the software updates influence makes bug identification difficult. Additionally, it increases the significance of build and deployment procedure.