# Senior System Design

CpE 191 and EEE 193B

Week 2

- According to [O'CONNOR , IEEE Spectrum, 2001]: "In developing products and systems***, testing, and not design***, is usually the more expensive, time-consuming, and difficult activity."

- Products and systems undergo testing throughout their development and before they are delivered.

- The system should be tested to ensure it meets the **engineering requirements and specifications**.

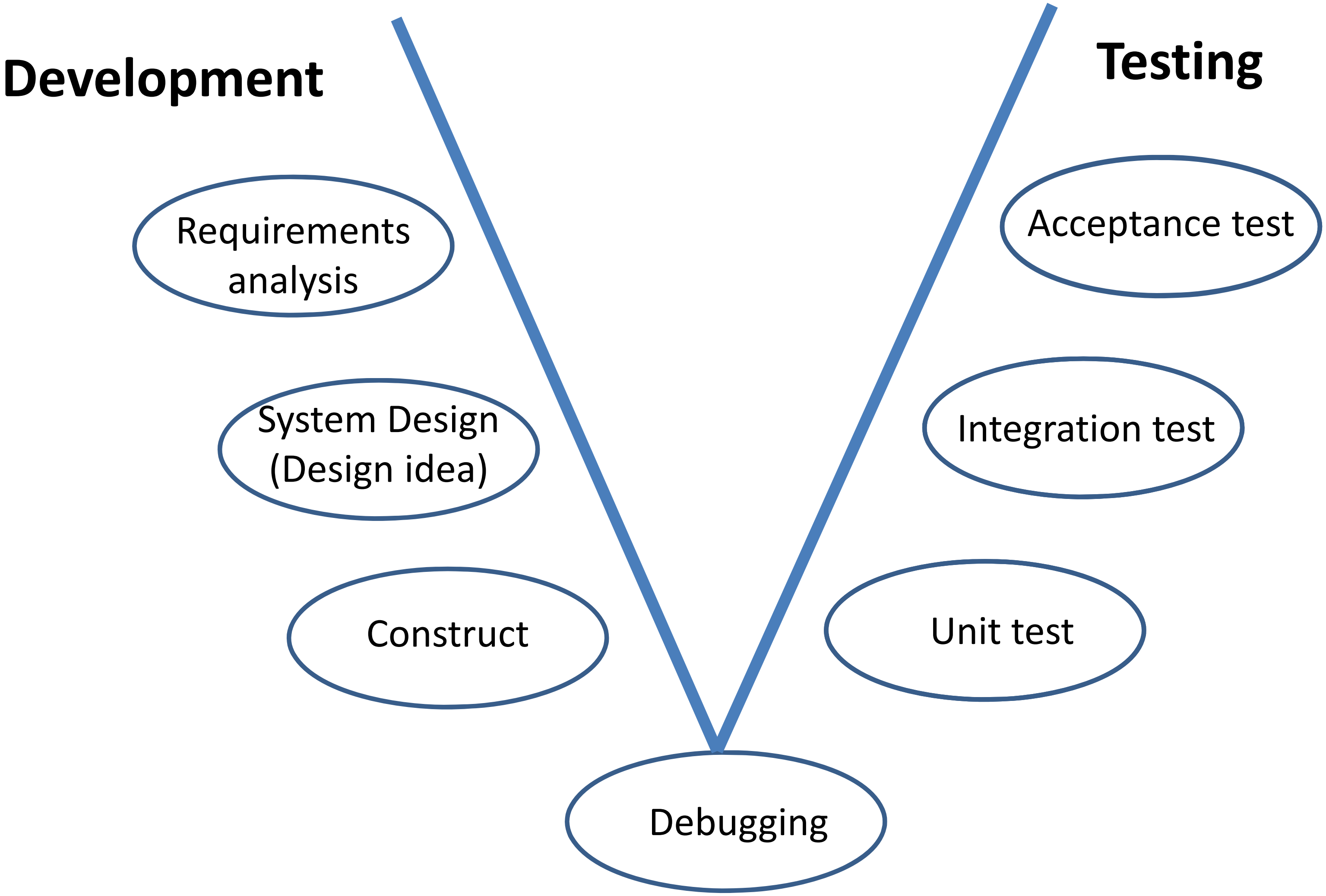
- One of the desirable properties of an engineering requirements is that it be verifiable, i.e., testable

# Testing your device

- Testing should be considered throughout system development
- Start testing early in the development cycle: it is better to correct errors early, rather than wait until their become much larger problem later.
- The cost to correct errors increases exponentially with project lifetime.
- Two important concepts in testing: observability and controllability

# Integrated testing and development

Testing can be a part of the development process: testing activities are performed at every stage of the product development lifecycle as depicted by the V-model below.



**Testing the device**

- Ideally, testing will exercise all system components in all possible ways.
- This is not possible since there are costs associated with testing, so different criteria are used to judge how well our testing strategy “covers” the system.

**Device test categories:**

- Life tests (reliability)
- Functionality tests

**Device test approaches:**

There exist two approaches for testing:

- Black box testing and,
- White box testing

# Black box and Black box testing

- White-box testing takes into account the internal mechanism of a system or component (IEEE, 1990). White-box testing is also known as clear box testing, or glass box testing (Beizer, 1995).

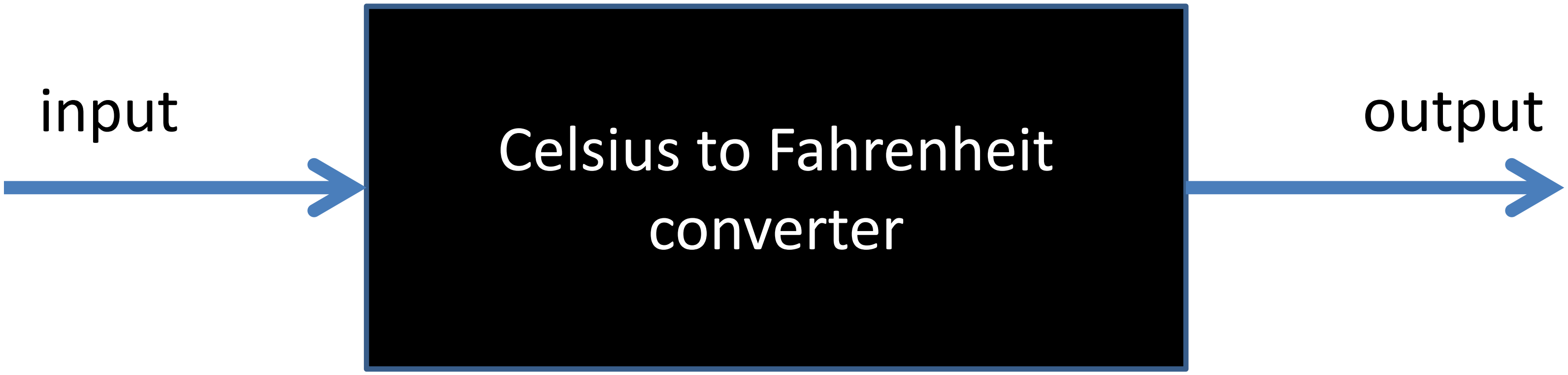
- Black box testing: examines the functionality of a product without peering into its internal units. Black box tests are performed without any knowledge of the system internal organization.

- Black box testing is typically conducted by changing the inputs and comparing the outputs to their expected values.

- Unlike white box testing, black box testing tends to be applied later in the development process

- The input and output values can be classified as:
  - Typical
  - Boundary
  - Extreme
  - Invalid

# Black box testing



input

Celsius to Fahrenheit converter

output

Tests cannot be accomplished by enumerating all possible values. You need to have candidate inputs that represent and capture the behavior of the system. The input and output values can be classified as:

- Typical inputs: normal operation temperature (example: room temperature)
- Boundary: 0C
- Extreme: Absolute zero
- Invalid: -500C

# White box testing

- White box tests are conducted with the knowledge of the internal workings of the system. This test targets specific internal nodes of the system.

- The test should be to handle typical, boundary, extreme and invalid situations

- When you design a system, you want to increase its ***testability***, why?

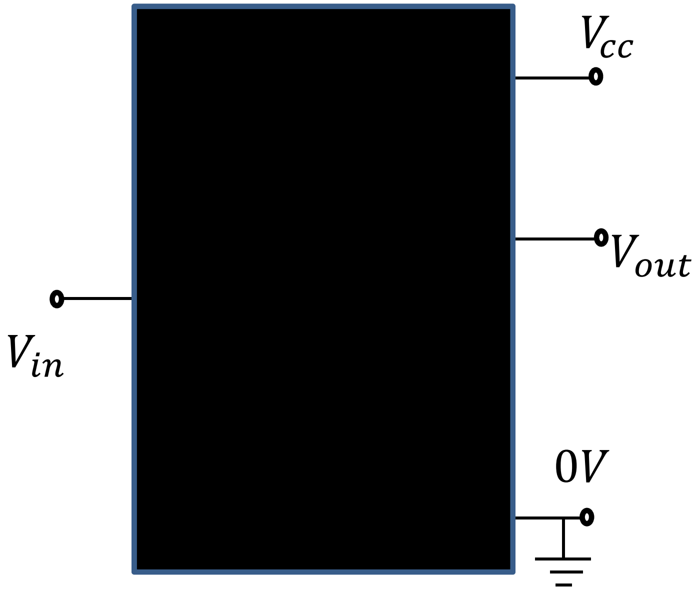
- A failure of a component can be quickly located. A testable design is easier to debug and service in the field.

- One way to increase testability is to increase controllability and observability
  - Controllability: The ability to set any node in the system to a prescribed value
  - Observability: the ability to observe any node of a system

# Example: Transistor amplifier design

- Common emitter amplifier: amplify the input signal to produce a linearly proportional output.

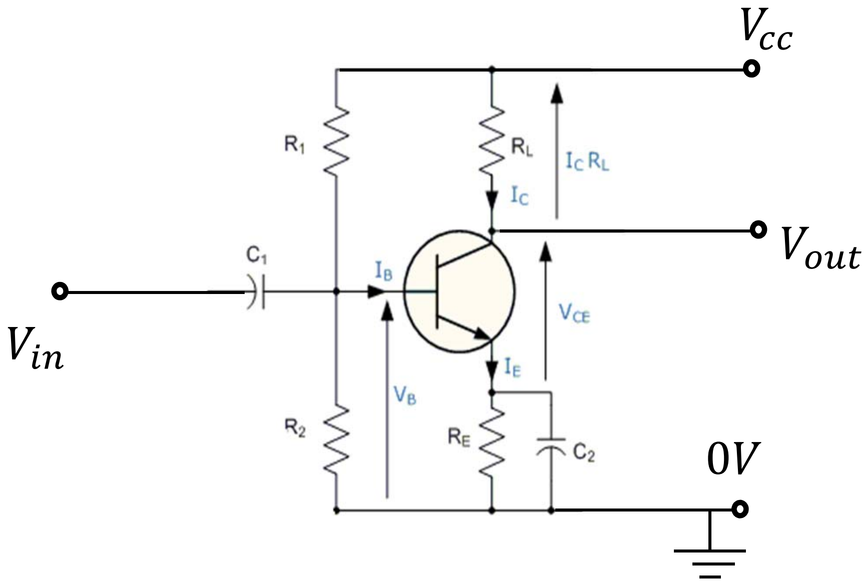
- Black box testing:
  - Check supply and ground voltages
  - Vary the input signal and observe the output



# Example: Transistor amplifier design

- Common emitter amplifier: amplify the input signal to produce a linearly proportional output.

- White box testing: We use knowledge of the internal workings of the design
  - Check supply and ground voltages
  - Check node voltages: $V_B, V_C, V_E$ to see if they meet the expected design values.



# Test case properties

- Runnels [1] defines the following attributes for effective tests
  - Accurate
  - Economical: uniform time to test
  - Limited in complexity
  - Repeatable: test can be performed and repeated by another person, i.e., wide range of testers
  - Traceable: the test should verify a specific requirement
  - Self cleaning: the system should return to pretest state after test is completed.

[1] How to write better test cases, Dianne Runnels, 1999, available online.

# Debugging

- At the implementation level, tasks such as constructing circuits, wiring integrated circuits, writing codes are carries out. Inevitably, there will be problems where a component does not function as expected. This is commonly referred to as a bug.
- Bugs fell into one of two camps: Bohrbugs and Heisenbugs.
- The debugging process is the same as the scientific method
  - Observe
  - Hypothesize
  - Experiment
  - Repeat
- Debugging is an iterative process.

# Unit testing

- A unit test is a complete test of a module’s functionality
- The extent to which the test cases cover all possibilities is called the **test coverage**
- High test coverage: test every possibility, but this is not practical. What is the solution?
- Solution: use representative values: typical, boundary, extreme, invalid.

Temperature conversion

```
if $20< input< 30$
    output=lookup table 1
else
    output =lookup table 2
```

We need to have at least two test cases: one for the *if* clause and the other one is for the *else* clause.

# Summary of common tests

| Test type | General scope | Opacity | Who generally does it? |
|---|---|---|---|
| Unit | Small unit, module | White box | The developer |
| Functional | Whole product | Black box | Independent tester |
| System | Whole product in representative environments | Black box | Independent tester |
| Acceptance | Whole product in customer's environments | Black box | Customer |

Summary of common tests [Williams, 2006]
Testing Overview and Black-Box Testing Techniques by Laurie Williams, available online

# Format of the test case [Williams, 2006]

The format of the test case is important. A suggested format [Williams, 2006] is shown in the table below.

| Test ID | Description | Expected results | Actual results | Pass/Fail |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

- Every test has a unique identifier
- Description: specifically describe the set of steps / input for the particular condition you want to test.
- Expected result: write down expected results **before** test.
- Actual result: record actual results **after** test.

# What goes in the description? Anatomy of a test case

- Statement of purpose, what is being tested
  - Method, how it will be tested
  - Setup, environment, data
- Steps - actions and their order

# System testing

- “Integration test is testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them” [Williams, 2006].

- After individual units have been tested, they are integrated together to make the entire system. Hence, integration testing checks that the overall system operates correctly.

- Allows to detect missing functions or missing interface between functions.

- The integration test must be traceable to the requirements stated in the initial design idea.

- Validation of marketing claims:
  - ‘’This model is 3 times faster than the prior version.’’

# Smoke test

- Turn on the system and see what happens
  - There is smoke, something is burning up: test fails
  - No smoke, I am a genius: test passes
- The smoke tests might be run before deciding to proceed with further testing
- The purpose of smoke tests is to demonstrate stability
- Usually not a systematic way to perform testing

# Validation and verification

Important concepts in software and industrial engineering.

- **Validation:** are we building the right product?
- **Verification:** are we building the product right?

- Example 1: Team 13  signs a multi million contract with the police department to build a quad copter to monitor the American river. However, to save time and money, they decided to build a boat instead.

- Example 2: Team 13 took millions of dollars from the police department to design an autonomous quad copter, but the copter cannot fly higher than 1 meter.

# Verification and validation

- Verification:

  "Are we building the product right”.

  “The determination by objective, repeatable methods that an item satisfies its initial requirements.”

- Validation:

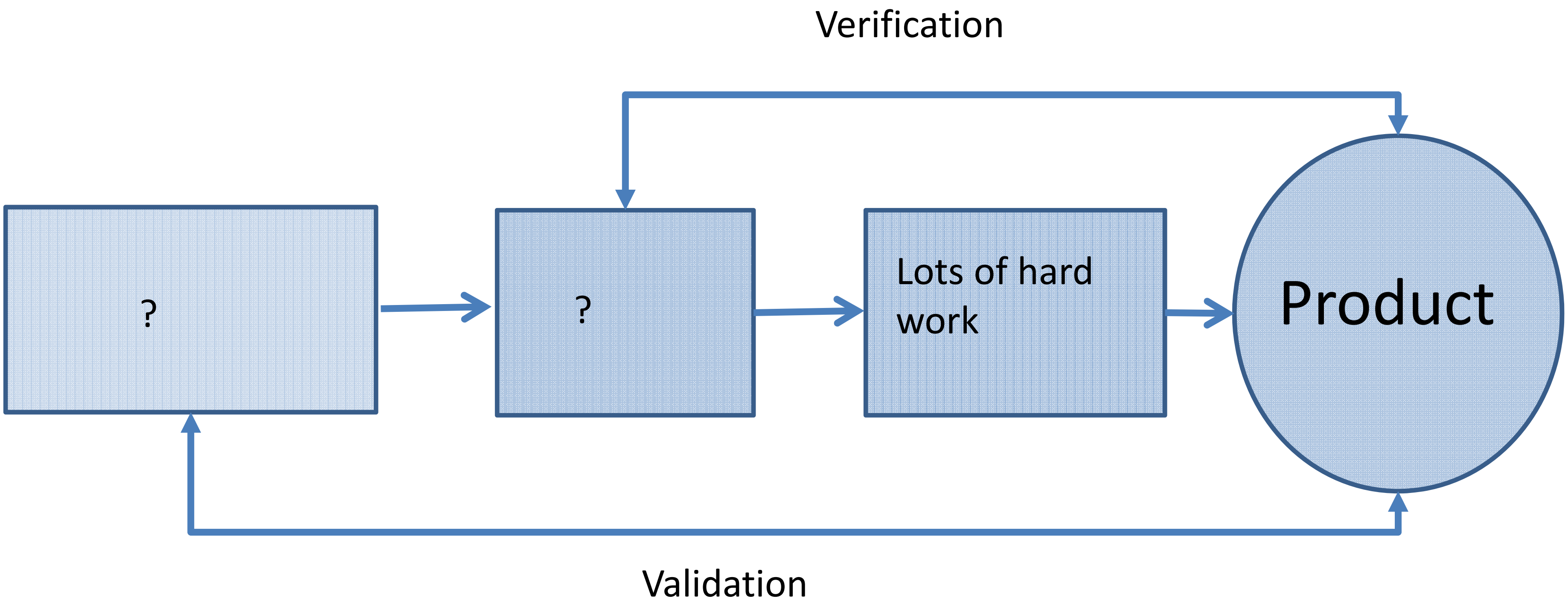
  "Are we building the right product”.

  “The determination by objective, repeatable methods that an item can be used for specific purpose.”

- Testing is critically important for quality software:

Industry averages:

- 30-85 errors per 1000 lines of code;
- 0.5-3 errors per 1000 lines of code not detected before delivery.


Verification
?
?
Lots of hard work
Product
Validation

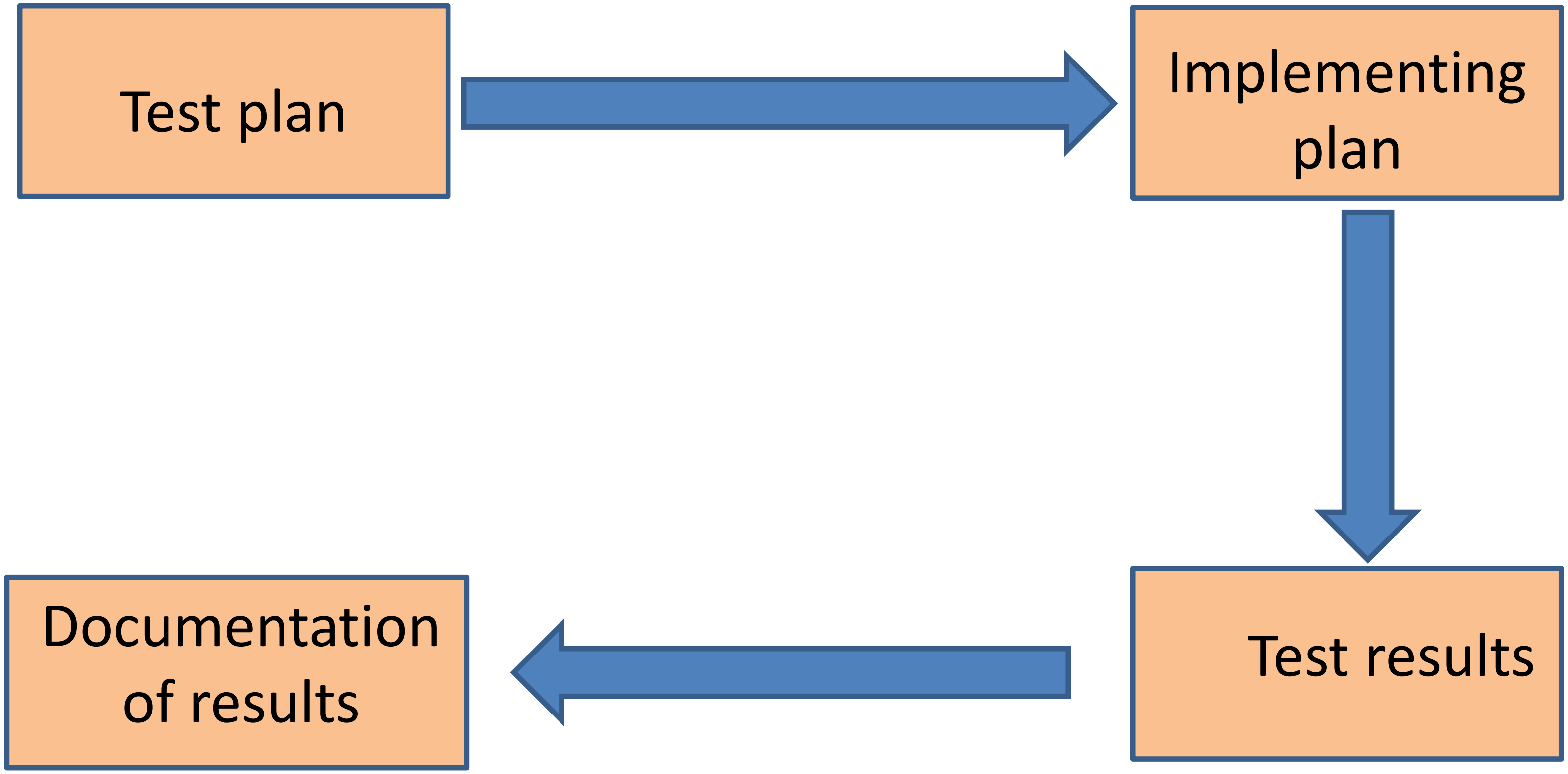
Examples of hardware tests:

- In-circuit test (ICT)
- Burn-in test

- In-circuit test (ICT): is used for printed circuit board test, checking for shorts, opens, resistance, capacitance, etc. There are many specialized platforms for performing in-circuit test; such as Aeroflex 4220, Agilent 3070, etc.

- Burn-in test: is the process by which components of a system are exercised prior to being placed in service to avoid early infant mortality failure.

# The roadmap



Test plan

Implementing plan

Test results

Documentation of results

# Benefits

The benefits of performing testing :

- Improve product quality and customer satisfaction
- Improve reliability
- Reduce cost of maintenance
- Validation of marketing claims. ‘’This model is 3 times faster than the prior version.’’

# Device test plan

- Industry can have stunningly complex test plans. Some companies have a separate testing department where testing is performed by specialized personnel (test engineers).

  - Read about why they feel the need for these complex plans.
    - Keywords: testing, black box, white box, integration test, verification, validation.
  - Devise a simple version for your product.
  - Be creative.

- To test your device, you need to think like someone who is going to use the device or the product on a daily basis.

# Senior System Design

CpE 191 and EEE 193B

Week 2