Politehnica University of Timisoara
Mobile Computing, Sensors Network and Embedded Systems Laboratory

Embedded systems testing

Testing Techniques in Automotive

instructor: Razvan BOGDAN

What is testing?

→ “Testing is the process of **demonstrating** that **errors are not present**.” 

→ “The purpose of testing is to show that a program performs its intended **functions correctly**.” 

→ “Testing is the process of **establishing confidence** that a system does what it is **supposed to do**.” 

→ “Testing is the process of **finding errors**.” 

Why do we need a technique?

→ A **test technique** / **test strategy** is used to **generate tests**

→ An **effective** technique finds **bugs**

→ Testing techniques are based on **requirements**

Example: Car interior lighting system

Customer requirements

List of requirements:

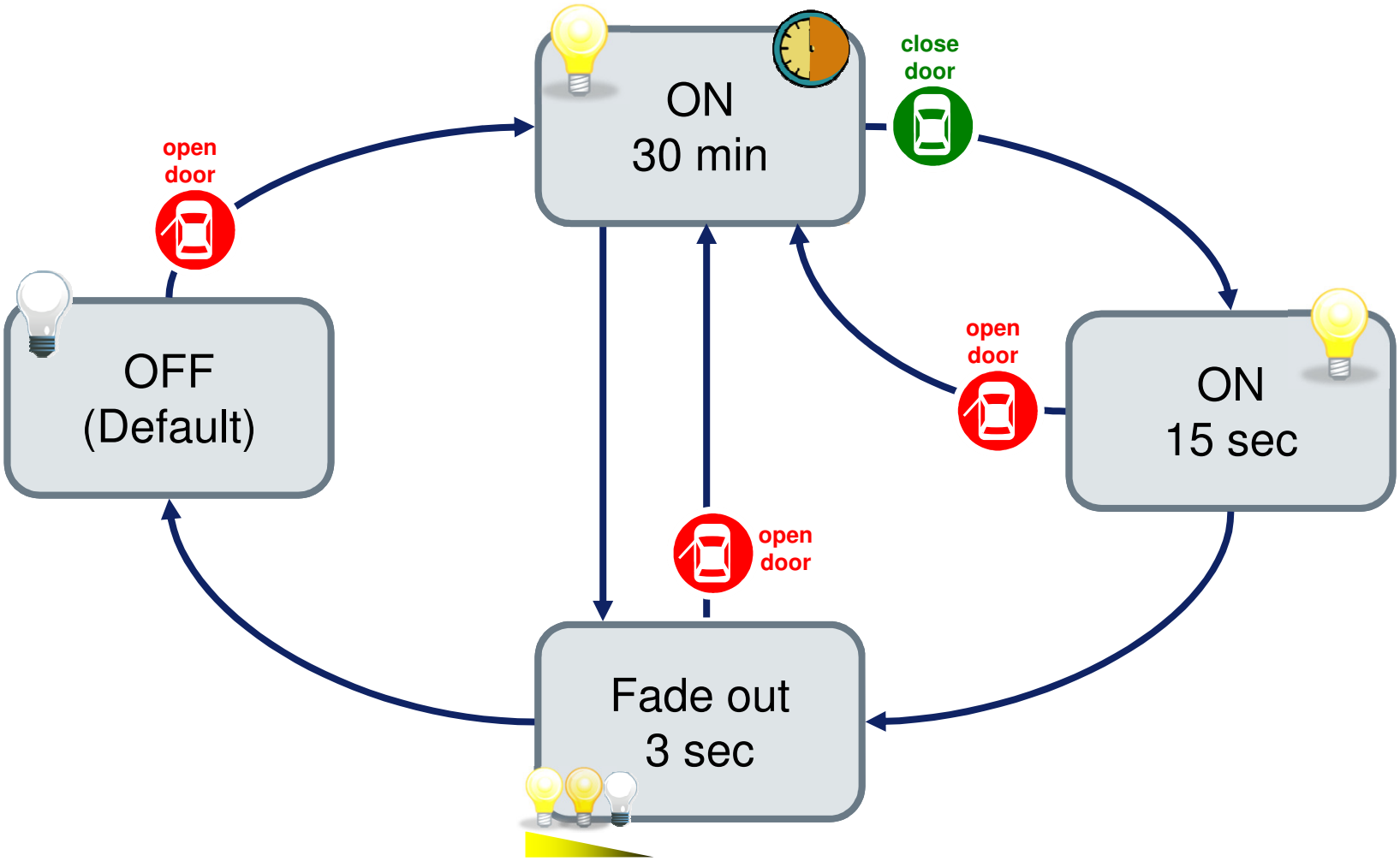
→ By opening the door of the car, the interior light should be turned on

→ If the door stays open, the light keeps lightening for half an hour

→ If the door is closed, the interior light will stay on another 15 seconds and then it should fade out in 3 seconds

Example: Car interior lighting system

Automotive requirements



Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Transaction Flow Modeling
Technique's Description

→ Transaction flows are a **representation of the system states** from the **user point of view**

→ **Identifies all branches, loops, queues and processes** that communicate by messages and defines test cases for these

How to use it?

→ You need a sufficient number of **transaction flowcharts** to cover all possible transactions

→ Select a sufficient number of **paths** through those transaction flows to assure complete coverage (every link and every decision is being exercised at least once)

Transaction Flow Modeling

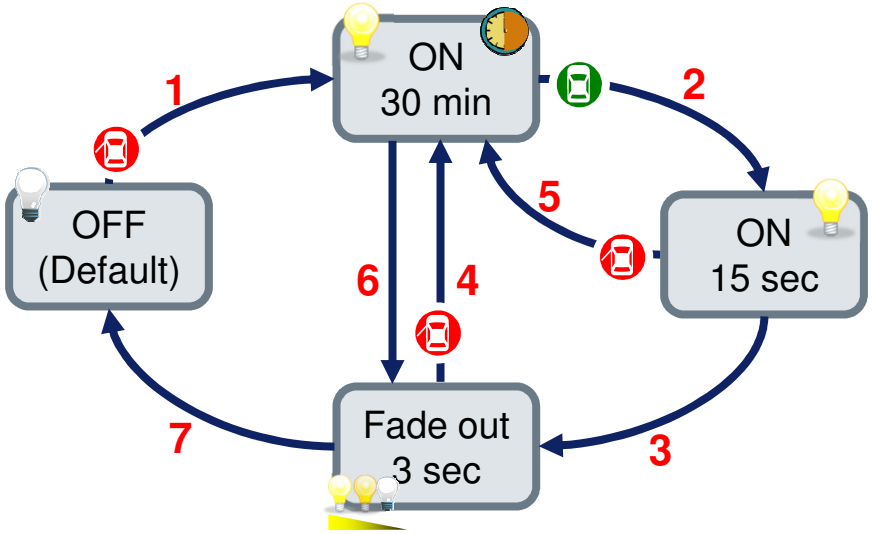
Technique's Implementation Example

→ One transaction flowchart

→ 1 path = 1 test specification

→ Many paths or combinations of paths to be tested

→ Q: what if a path fails?



A = an example of a single path
B = an example of a combination of paths

A: OFF (Default) → 1 → ON 30 min → 2 → ON 15 sec → 3 → Fade out 3 sec → 4 → ON 30 min → 2 → ON 15 sec → 5 → ON 30 min → 6 → Fade out 3 sec → 7 → OFF (Default)

B: OFF (Default) → 1 → ON 30 min → 2 → ON 15 sec → 3 → Fade out 3 sec → 7 → OFF (Default)

B: OFF (Default) → 1 → ON 30 min → 2 → ON 15 sec → 5 → ON 30 min → 6 → Fade out 3 sec → 4 → ON 30 min → 6 → Fade out 3 sec → 7 → OFF (Default)

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

All Round-Trip Paths
Technique's Description

→ Used to traverse the graph (state machine) and to generate a **transition tree**

→ Attempts to exercise **round trip paths**

→ Covers **100%** of the paths

round trip paths = paths that start and end in the same state

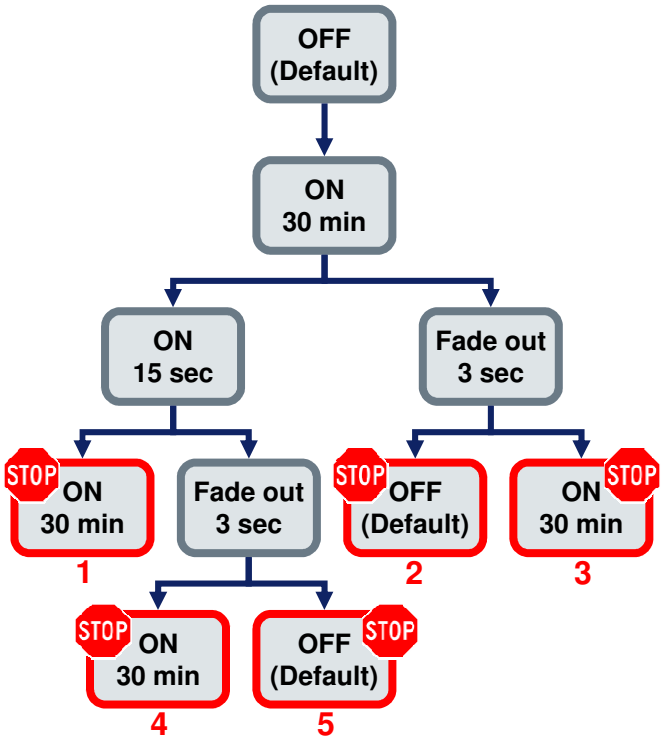
How to use it?

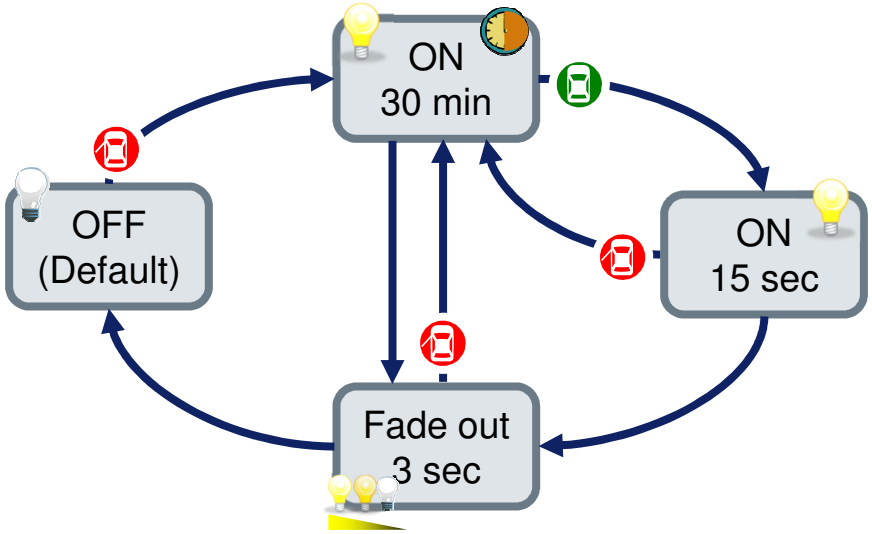
→ Follow a path and when you meet again a **previous state – STOP** and record that path

→ Create a test case for every resulted path

All Round-Trip Paths

Technique's Implementation Example





→ 5 paths resulted with this technique => 5 test specifications (DOORS)

→ If one path fails the other paths can pass

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Loop Testing
Technique's Description

→ **Repetitive processes** are difficult to start or stop correctly

→ Is effective for most graph models that have **loops**

loops = paths that end where they have started

How to use it?

→ Set **preconditions** to start a loop

→ Repeat loop N times

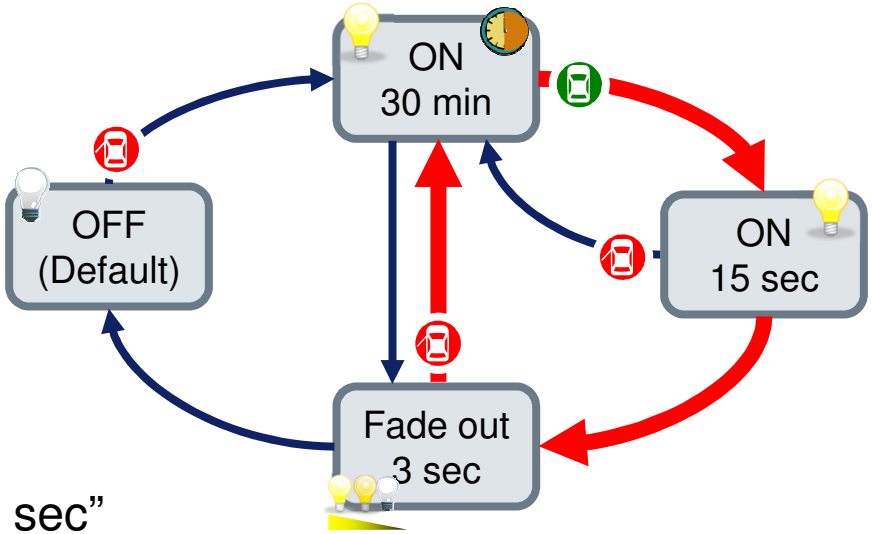
→ Check if the results are the same after every cycle

N = a determined number

Loop Testing
Technique's Implementation Example

Precondition

→ Open door:
set state "ON 30 min"



Test Description

→ 1) Close door:
set state "ON 15 sec"

→ 2) Wait 15 sec: set state "Fade out 3 sec"

→ 3) Open door: set state "ON 30 min"

→ 4) Repeat steps 1, 2, 3 for N times (N = 3)

Expected Result

→ System is in state "ON 30 min" – the interior light is on for 30 minutes

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Data Flow Testing
Technique's Description

→ A **data produced** in one state is expected to be used later

→ **Information** received by a receiving state **Rx** has to be **the same** as the one sent from the transmission state **Tx**

→ Information that passes from state to state is creating a **data flow**

How to use it?

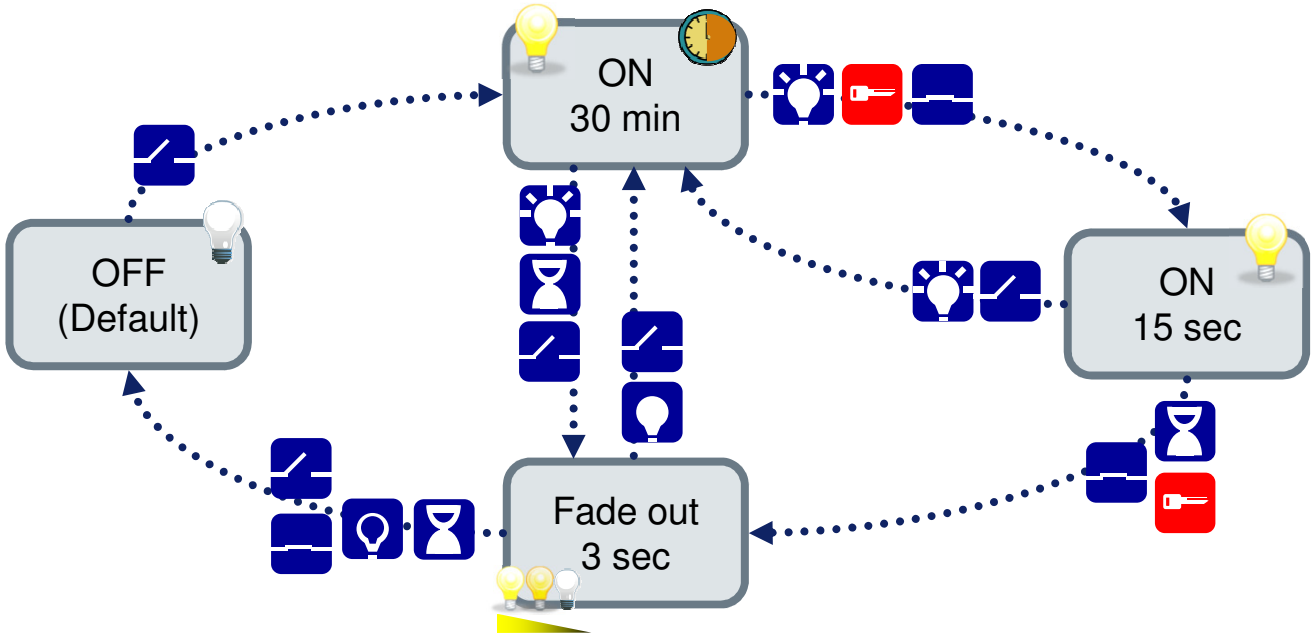
→ Transmit a data package from one state to another state

→ Check if the information that arrives at the receiving state is correct by the action that state takes

Data Flow Testing
Technique's Implementation Example

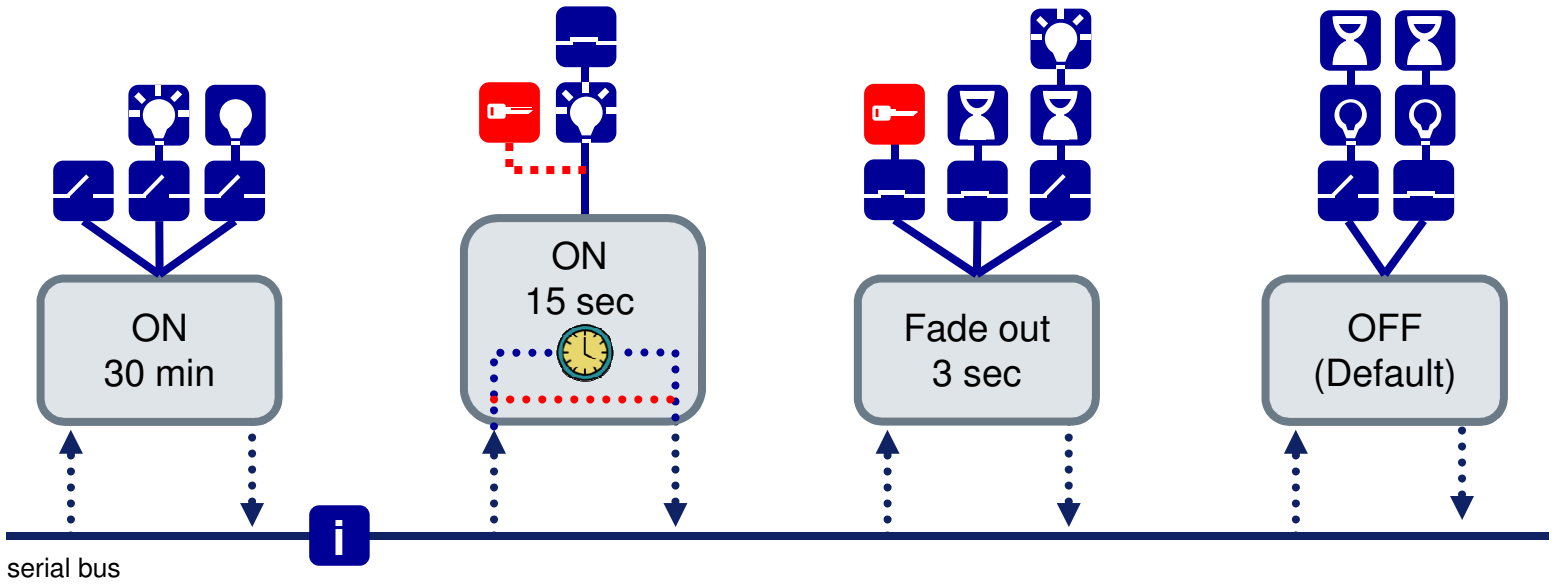
New requirement added:

→ If the light is ON and ignition is ON and the door is closed the light should fade out in 3 seconds

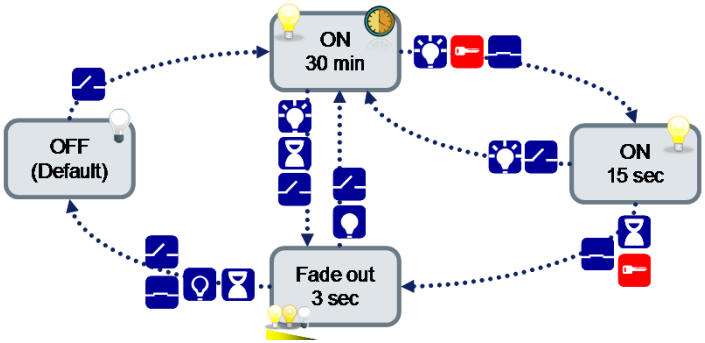


door open
door closed
light on
fade out light
light off
time elapsed
ignition on

Data Flow Testing
Technique's Implementation Example



→ "Ignition on" data should be sent from state "ON 30 min" to state "Fade out 3 sec" through state "ON 15 sec" which is bypassing the timer



Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Equivalence Partitioning
Technique's Description

→ The input of a **domain** is **divided** in classes of data

How to use it?

→ The precondition is to determine what type of input is: a value, a range, etc.

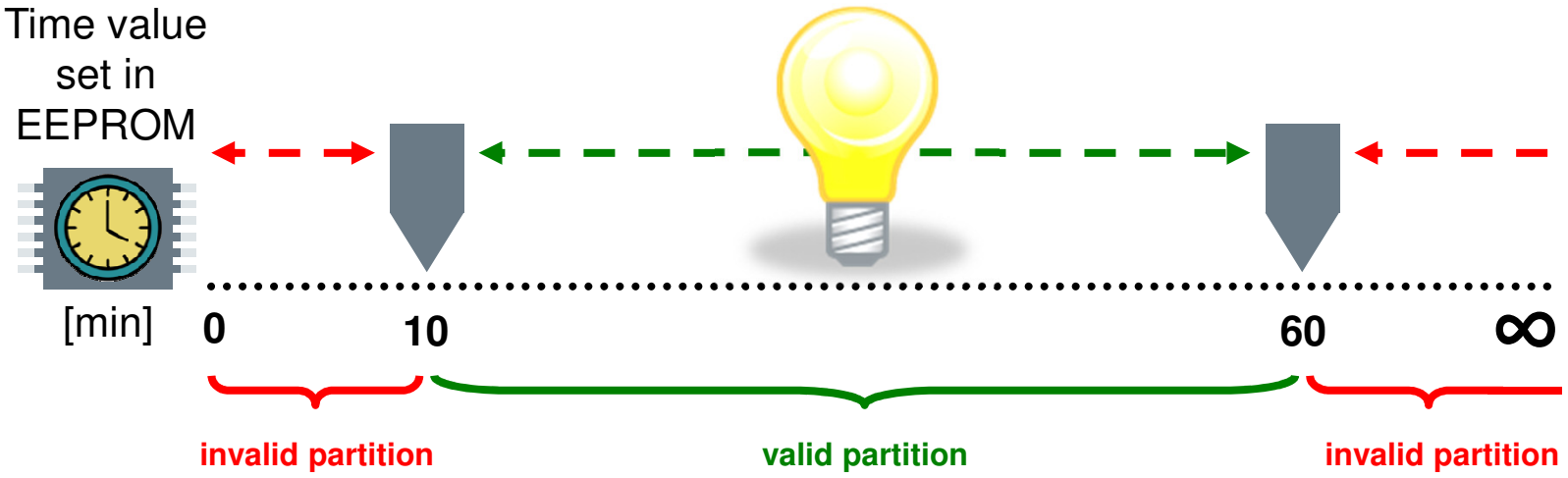
→ Divide the input in classes, named partitions

→ They can be:

- for a value, one valid and two invalid partitions
- for a range, one valid and two invalid partitions
- for a member of a set, one valid and one invalid partition
- for a Boolean value, one valid and one invalid partition

Equivalence Partitioning

Technique's Implementation Example



→ Range of values that are possible to be used: [10…60] minutes

→ All values are divided in 3 partitions

→ Dividing the range of values gives us the number of tests that can be made for the boundary value analysis technique

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Boundary Value Analysis
Technique's Description

→ Based on the **partitions** that where determined with equivalence partitioning technique, we can determine the tests that we have to make

→ **Boundaries** and **values around them** are tested

→ Example for 3 partitions:

How to use it?

→ Determine the **boundary values** and the **values around them**

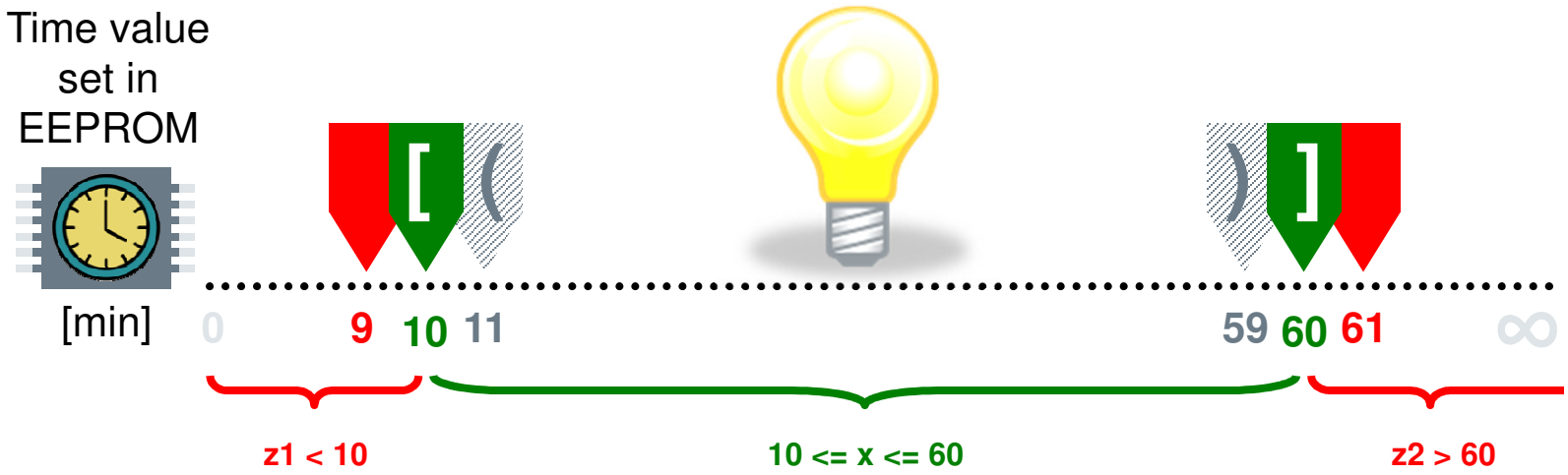
→ Create a test case for every **relevant value**

→ Use these values to set **preconditions**

→ **Measure results** to be accordingly with the pre-set values

Boundary Value Analysis

Technique's Implementation Example



Precondition	Test Description	Expected Result
1) EEPROM time value is set to 9 2) Set state "ON"	1) Check light and start stopwatch 2) Stop stopwatch when light is OFF	Light is ON for X minutes (Resolution = 1)
1) EEPROM time value is set to 10 2) Set state "ON"	1) Check light and start stopwatch 2) Stop stopwatch when light is OFF	Light is ON for 10 minutes (Resolution = 1)

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Regression Testing
Technique's Description

→ In a system a **new feature** is implemented

→ Testing the **newer version** of the system by using the same test cases of the **previous** version of the system, for the **features that are not changed**

How to use it?

→ Test the system by using the test cases that already exist

→ Write new test cases for the domains that include the new features

→ Other testing methods can be used to cover all failure possibilities of the system

→ Use the new test cases and test the system further on

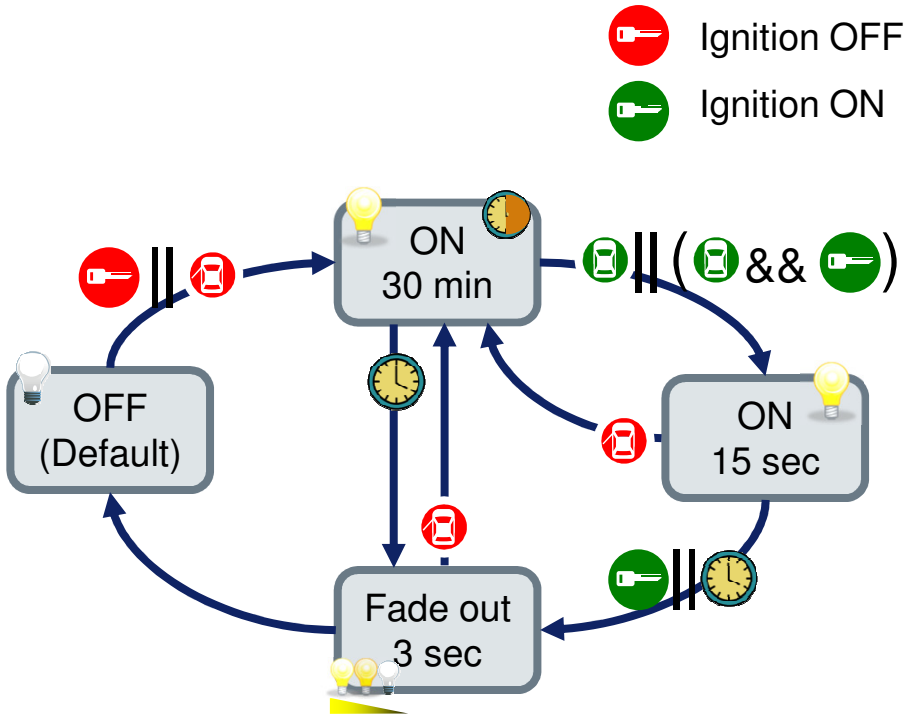
Regression Testing
Technique's Implementation Example

→ If the requirements were changed by adding the "Ignition ON / OFF" condition, new tests have to be added to the specifications

→ The functionality of the system is changed with additional conditions, but the test cases made for the first version can be reused

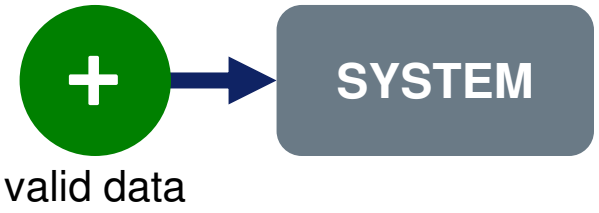
→ For example, the tests made with "All round-trip paths" technique are valid and can be reused

→ For the new feature added to the system, new test cases have to be created including the latest conditions

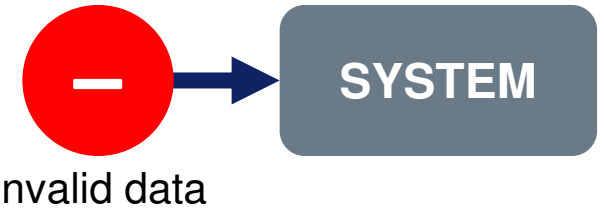


Positive vs. Negative Testing
Techniques comparison

→ Positive Testing – giving as input **valid** data



→ Negative Testing – giving as input **invalid** data



→ Remark: All tests that have been made until now are examples of positive testing

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Negative Testing
Technique's Description

→ Defining as **input** an **invalid data**

How to use it ?

→ Check the preconditions of the system

→ Define some input values that could appear, but are unspecified and seem to be invalid for the system

→ Write test cases that uses those values as inputs

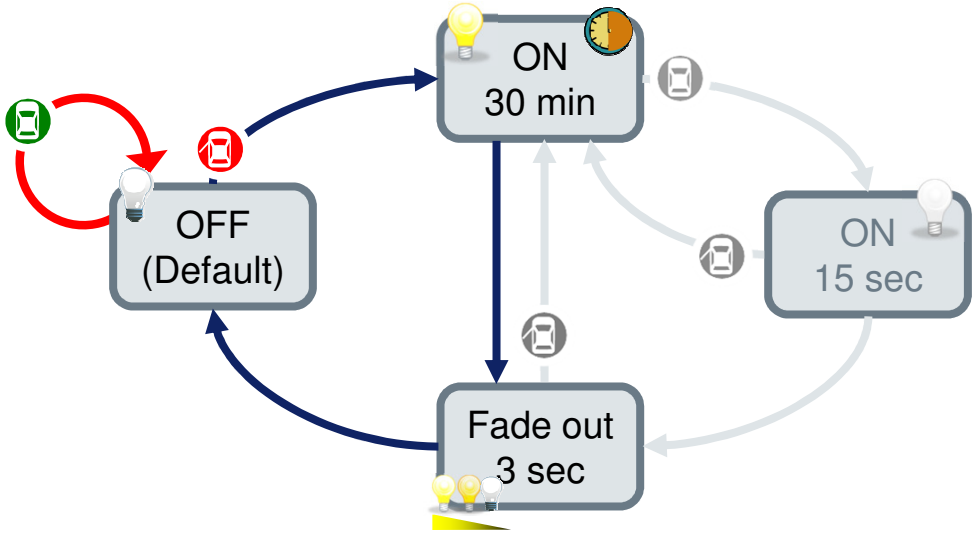
Negative Testing
Technique's Implementation Example

→ After a complete cycle with the door open the system is back to it's default state OFF

→ Then the door is closed

→ There is nothing specified in the requirements about this

→ Q: What does the system should do?



Precondition	Test Description	Expected Result
Open the door and let it open for 31 minutes (until light goes OFF)	1) Close the door 2) Check interior light	The light should NOT turn ON after the door is closed

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Error Guessing
Technique's Description

→ Based on **experience** and judgment of the tester

→ Is the **art** of finding hidden errors

How to use it?

→ Explore systems functionality

→ Think where a location for an error could be

→ The errors produced are mostly placed in exceptional places

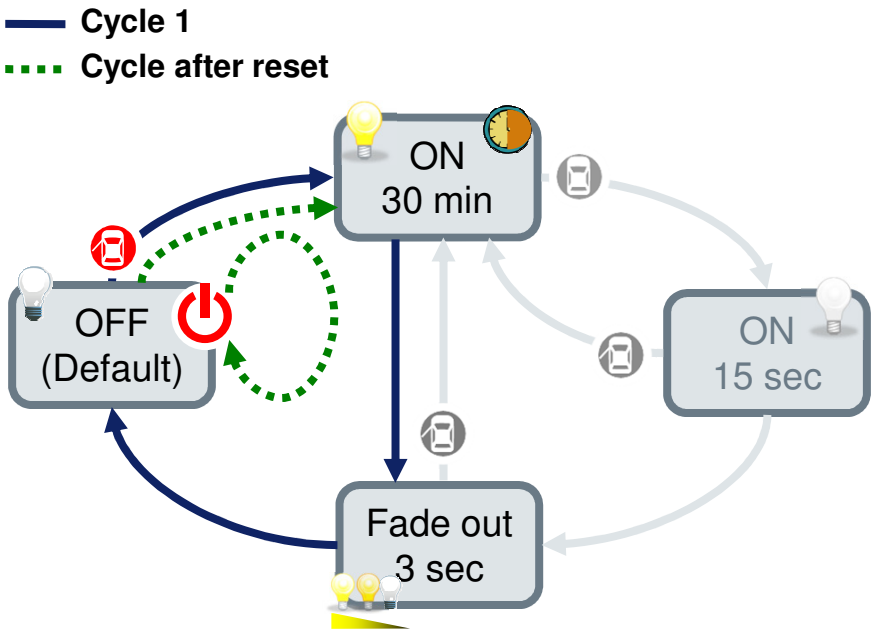
→ The errors are actions that aren't specified or unusual accessed

Error Guessing
Technique's Implementation Example

→ After **one cycle** with the **door open** the system is back to it's default state OFF

→ Then the system is **reset**

→ Q: Interior light is set to ON or it should stay in OFF state?



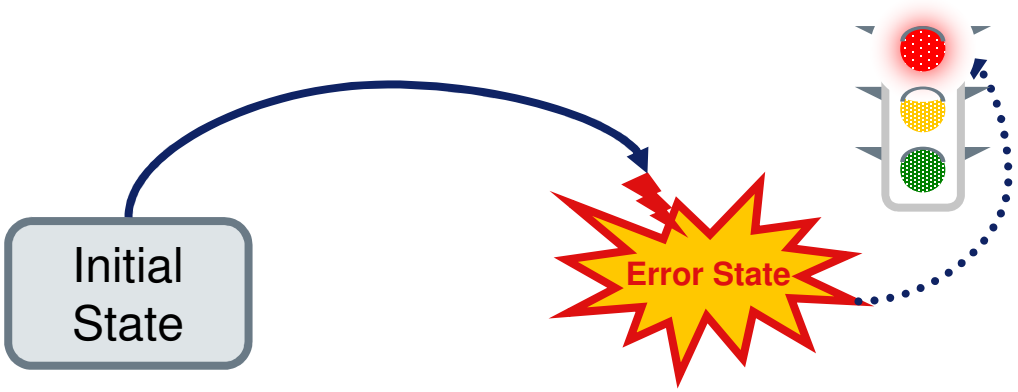
Precondition	Test Description	Expected Result
Open the door and let it open for 31 minutes (until light goes OFF)	1) Reset system 2) Check interior light	The light should turn ON if: door open & Reset

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Error Handling Testing
Technique's Description

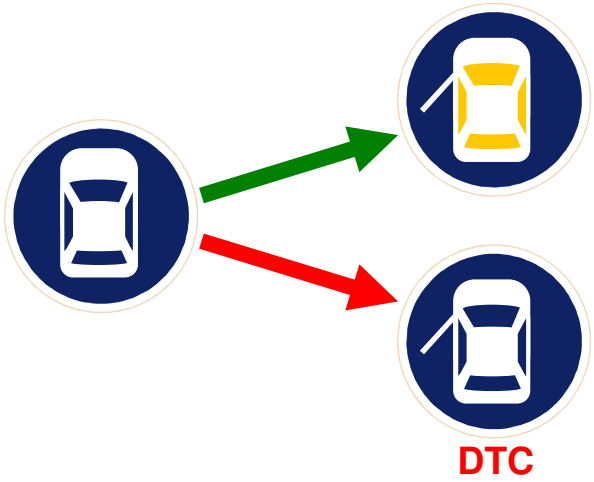
→ A system should **recognize** and **locate** the error

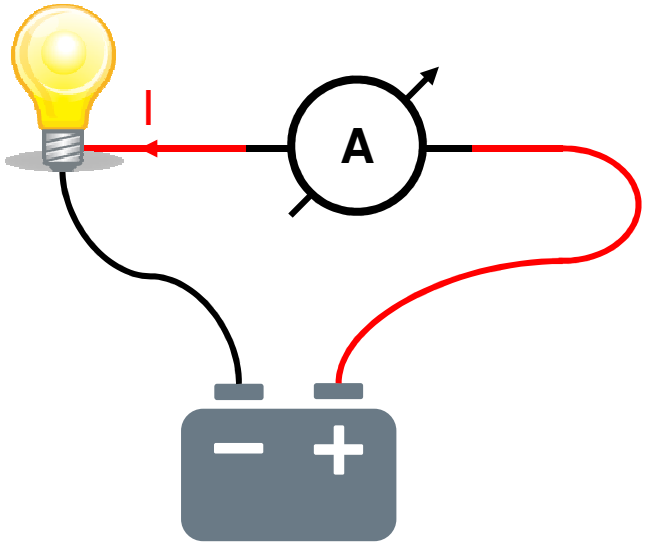


Error Handling Testing
Technique's Implementation Example

New requirement added:

→ The system should detect and save a DTC (Diagnostic Trouble Code) if the light bulb is burned





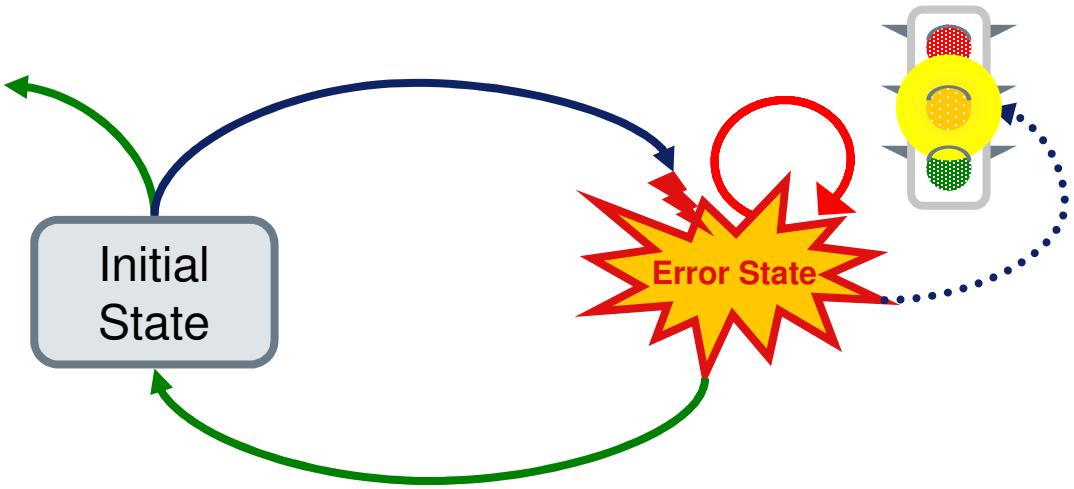
→ If current I is 0 Amps then the DTC should set

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Recovery Testing
Technique's Description

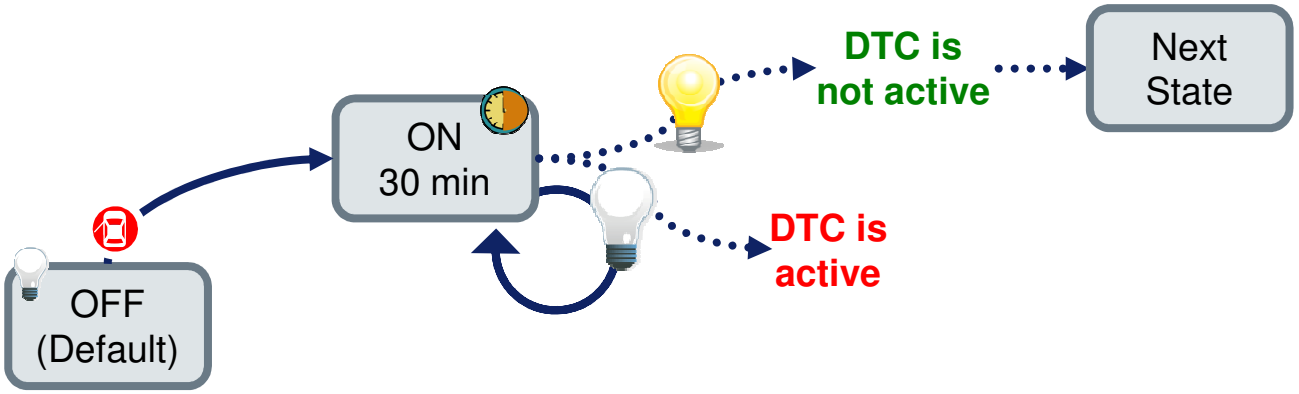
→ A system should **recover** from it's error state to it's initial state



Recovery Testing
Technique's Implementation Example

→ For example, an error could appear if the bulb is burned or if a short-circuit was produced

→ The system needs to locate this error and to announce that the interior lighting system has a problem



→ Afterwards the bulb is replaced with a good one

→ The system is restarted

→ This time, the error shouldn't appear anymore - the system has to recover

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Stress Testing
Technique's Description

→ Requires to hold the system in **continuous action** for a long time

→ System should perform under normal conditions, but it should perform under **extreme conditions**, too

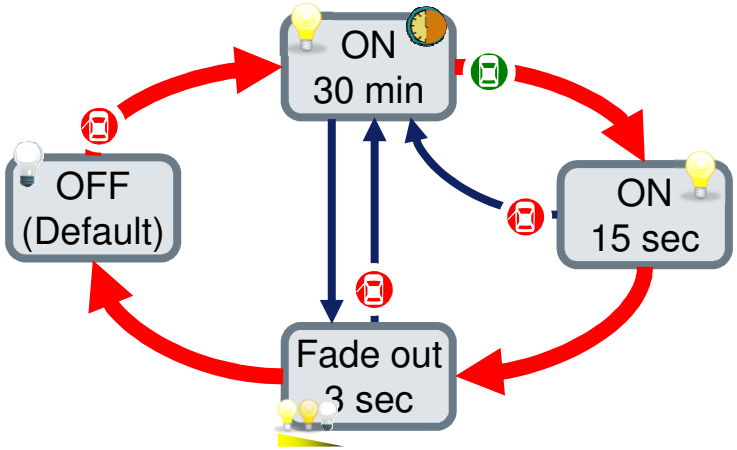
How to use it?

→ Repeating actions repeated times, without giving the system any breaks

→ Every tact the system should process something

Stress Testing
Technique's Implementation Example

→ Testing the timer for the "ON 15 sec" state for 100 cycles



Precondition	Test Description	Expected Result
System is in OFF (Default) mode	1) Open door (1s) 2) Close door (1s) 3) Wait 15s 4) Check light status to be ON 5) Wait 3s 6) Check light status to be OFF 7) Repeat steps 1-6 100 times	Time while light is ON before is completely OFF (fade out) is 20s for every cycle

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing

Load Testing
Technique's Description

→ Putting **heavy demand** on a system

→ Identifies the **maximum operating capacity** of a system

→ **Error** conditions are the **expected** result

How to use it?

→ Determine maximum operating capacity

→ Entertain the system with continuous actions over capacity

→ Expect errors when the system is high demanded

Load Testing
Technique's Implementation Example

New requirement added:

→ The maximum power of the lamp is 15W

Precondition	Test Description	Expected Result
5W lamp is mounted	Set state "ON 30 min"	System functions correctly
10W (default) lamp is mounted	Set state "ON 30 min"	System functions correctly
15W lamp is mounted	Set state "ON 30 min"	System functions correctly
20/25/30W lamp is mounted	Set state "ON 30 min"	System not functioning / relay is damaged

Summary
Testing Techniques

- → Transaction Flow Modeling
- → All Round-Trip Paths
- → Loop Testing
- → Data Flow Testing
- → Equivalence Partitioning
- → Boundary Value Analysis
- → Regression Testing
- → Negative Testing
- → Error Guessing
- → Error Handling Testing
- → Recovery Testing
- → Stress Testing
- → Load Testing