Effective Verification for DO-254 Projects

David Landoll

Applications Architect

May 1, 2008

Mentor Graphics®

Agenda

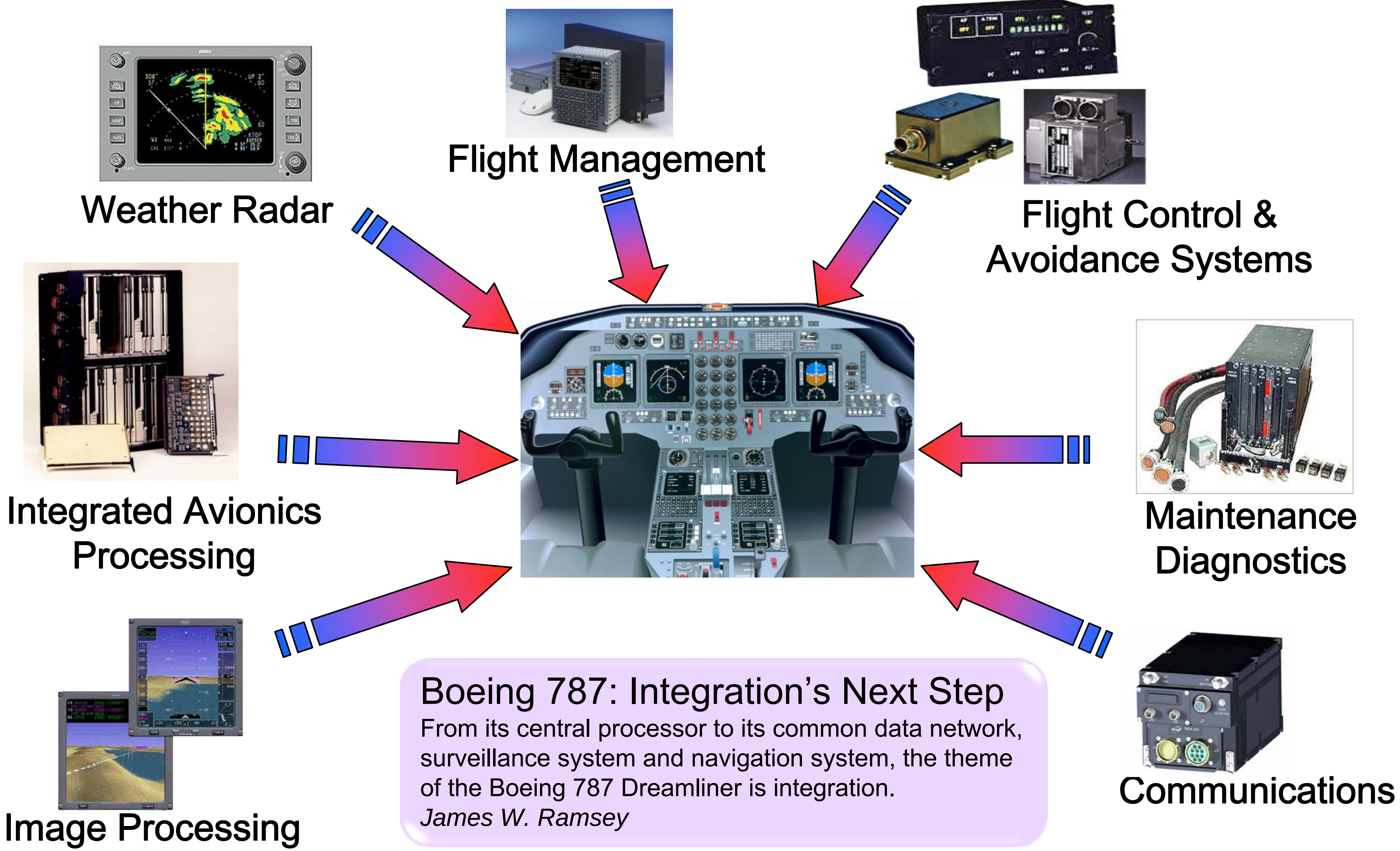
- Verification Challenges and Safety-Critical Design
- DO-254 Requirements for Verification
- Safety-Critical Verification: Recommendations
- Conclusion

Verifying for Safety Critical

- **Know that end product is customer safe**
- **Track that requirements are *thoroughly* verified**
- **Ensure verification processes to meet compliance to pertinent standards**
- **Keep project on schedule**

Adobe
DOORS®
XML
ASCII
Company
Standards
FEDERAL AVIATION ADMINISTRATION
RTL

Avionics Integration – New Challenges



Weather Radar

Flight Management

Flight Control & Avoidance Systems

Integrated Avionics Processing

Maintenance Diagnostics

Image Processing

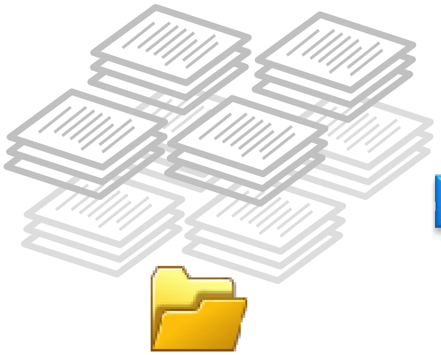
Communications

Boeing 787: Integration's Next Step

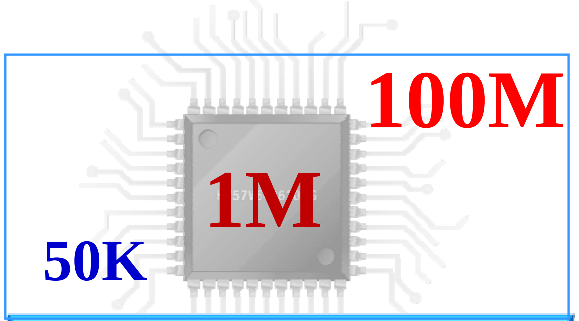
From its central processor to its common data network, surveillance system and navigation system, the theme of the Boeing 787 Dreamliner is integration.

James W. Ramsey

Verification Challenges

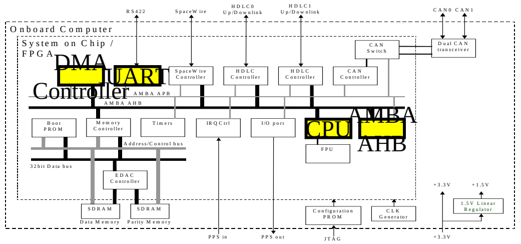


HDL Design Files



Growing Gate Counts

+



Highly Integrated Devices

Very Difficult to Verify!

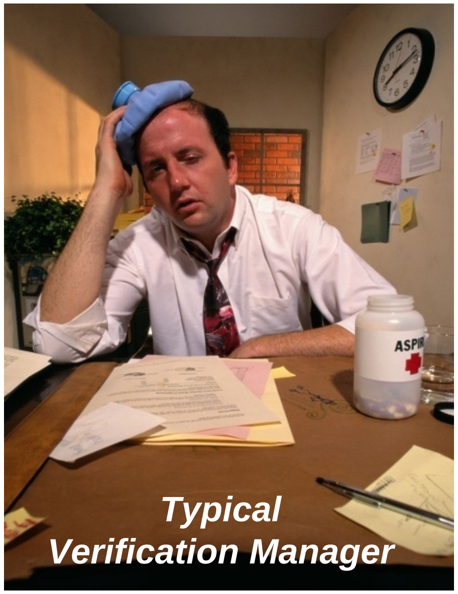
Major Issues

- Verifying Requirements (100's, 1000's, etc)
- Increasing Design Concurrency
- Many Clock Domains (Metastability)
- (Exhaustively) Verifying Safety-Critical Reqs

Is Verification Complete?

Questions for the Verification Manager

- **How do you know your tests really do *comprehensively* verify the requirements?**
  - Design performs its intended function
- **How do you ensure you’re testing the interactions between requirements (i.e., concurrency)?**
  - Design has no unintended functionality
- **How do you ensure you catch anomalous behaviors that might not be tied to requirements?**
- **How do you manage your verification effort, measure your progress, and prove that you’re done?**



Typical Verification Manager

Agenda

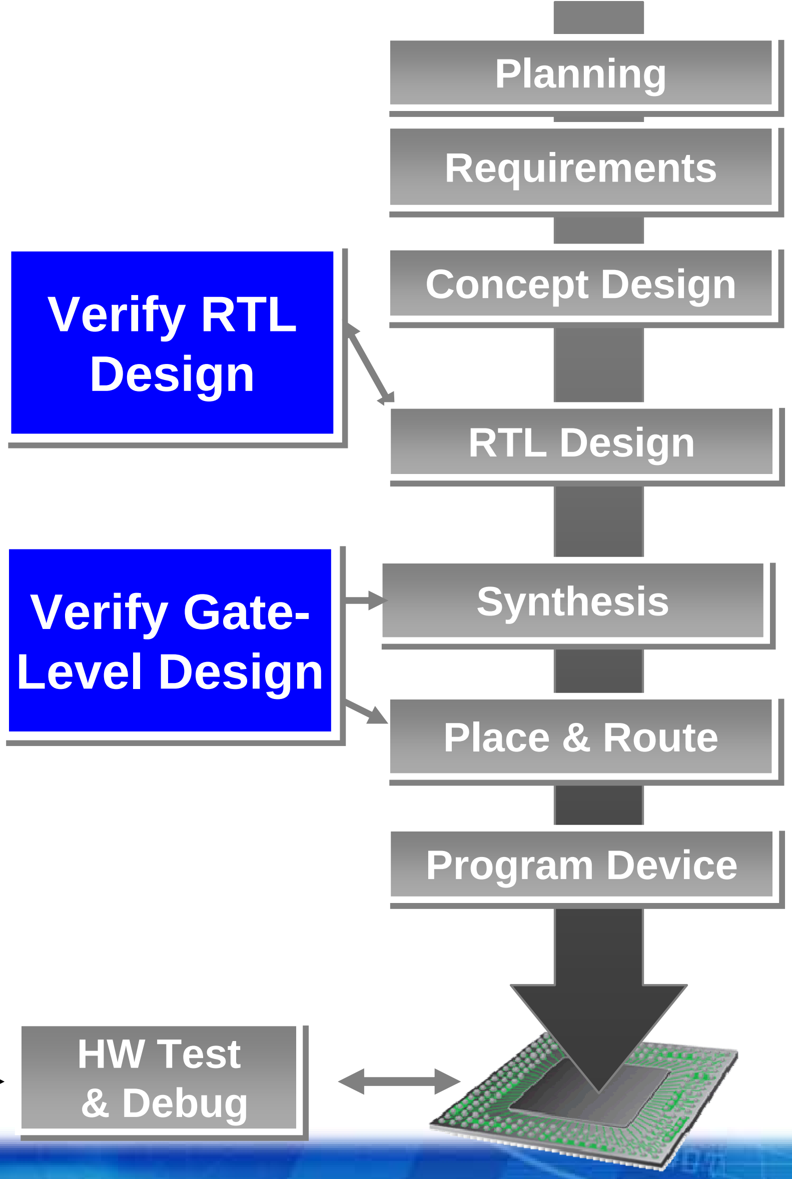
- Verification Challenges and Safety-Critical Design
- DO-254 Requirements for Verification
- Safety-Critical Verification: Recommendations
- Conclusion

Verification

Why it is Important

- **The purpose of DO-254 is *design assurance***
- **Designs must work as intended**
- **Quality verification is essential**
- **Verifying complex designs is *very challenging***

Note: In this presentation we will not be talking about testing the physical HW item, even though this is a requirement of "verification" for DO-254



Planning

Requirements

Concept Design

Verify RTL Design

RTL Design

Verify Gate-Level Design

Synthesis

Place & Route

Program Device

HW Test & Debug

Verification

What Does DO-254 Require?

Verification Independence **so designer doesn't test own code**

Requirements-based test **on both RTL and Gate-Level design representations (as well as end hardware item)**

Traceability from Requirements **to tests and results**

Coverage **to ensure verification is complete**

Advanced Methods **for level A/B projects**

Reporting data **for audits and management**

Verification

What Does Your Business Require?

An effective verification plan **to drive all verification activities**

Cost effective methods **to ensure profitability**

Resources **used wisely**

Metrics **for monitoring progress and completion**

Assurance **of high quality results**

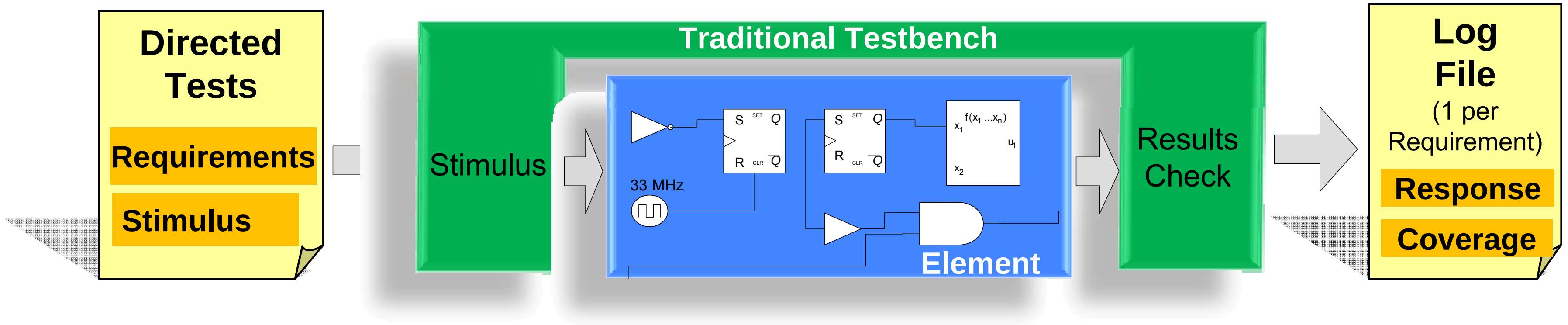
Compliance **to DO-254 requirements**

Agenda

- Verification Challenges and Safety-Critical Design
- DO-254 Requirements for Verification
- Safety-Critical Verification: Recommendations
- Conclusion

Directed Test

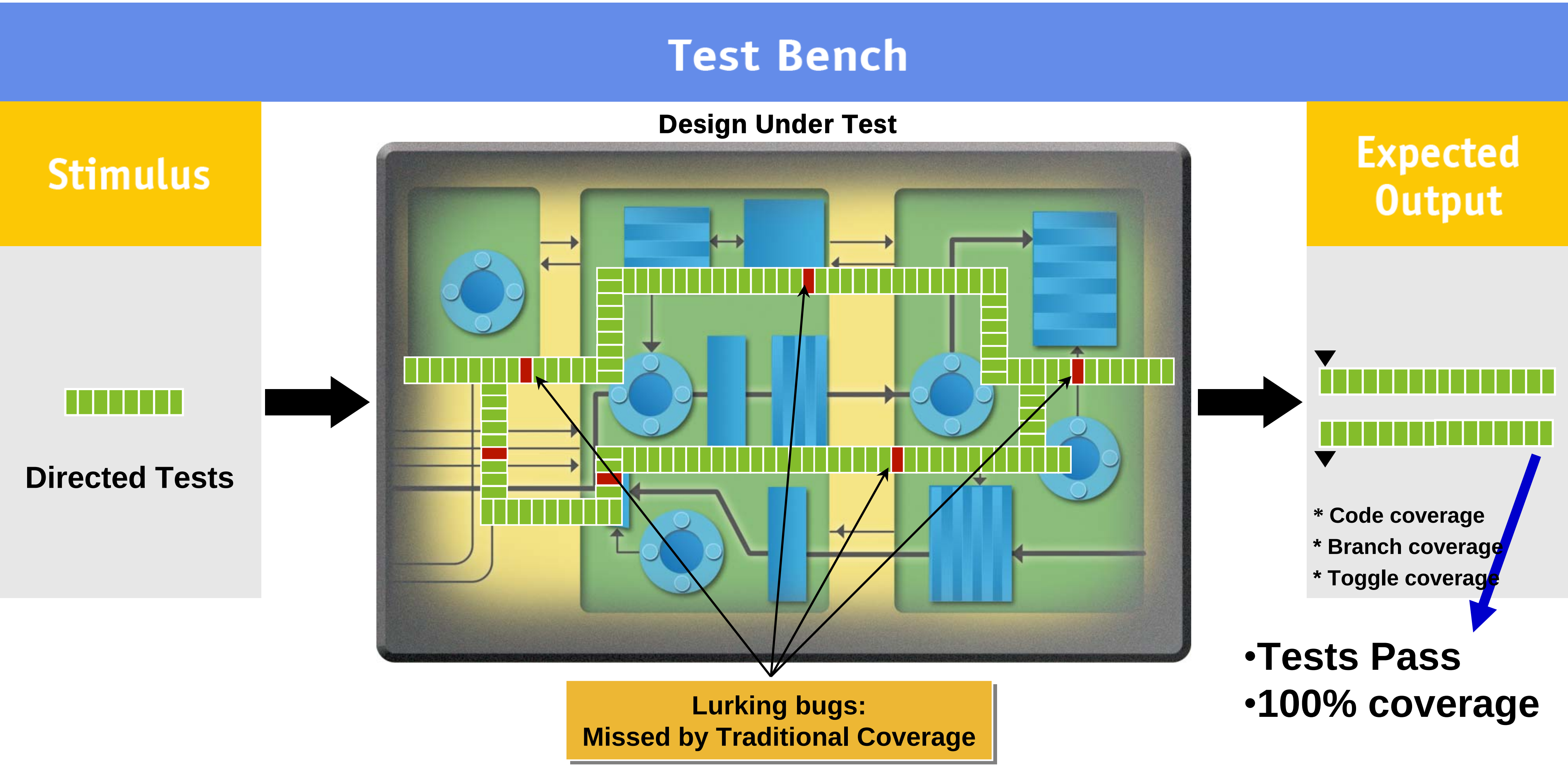
A Traditional Approach



- A good approach for traditional design styles
- Manually-written tests exercise requirements via specified stimulus
- Testbench applies stimulus/checks results
- Log file includes results of test
- Code coverage metrics determine if tests exercise RTL code

> *Note: This method begins to fail with increased device complexity, integration and a large number of requirements*

Traditional Coverage Limitation

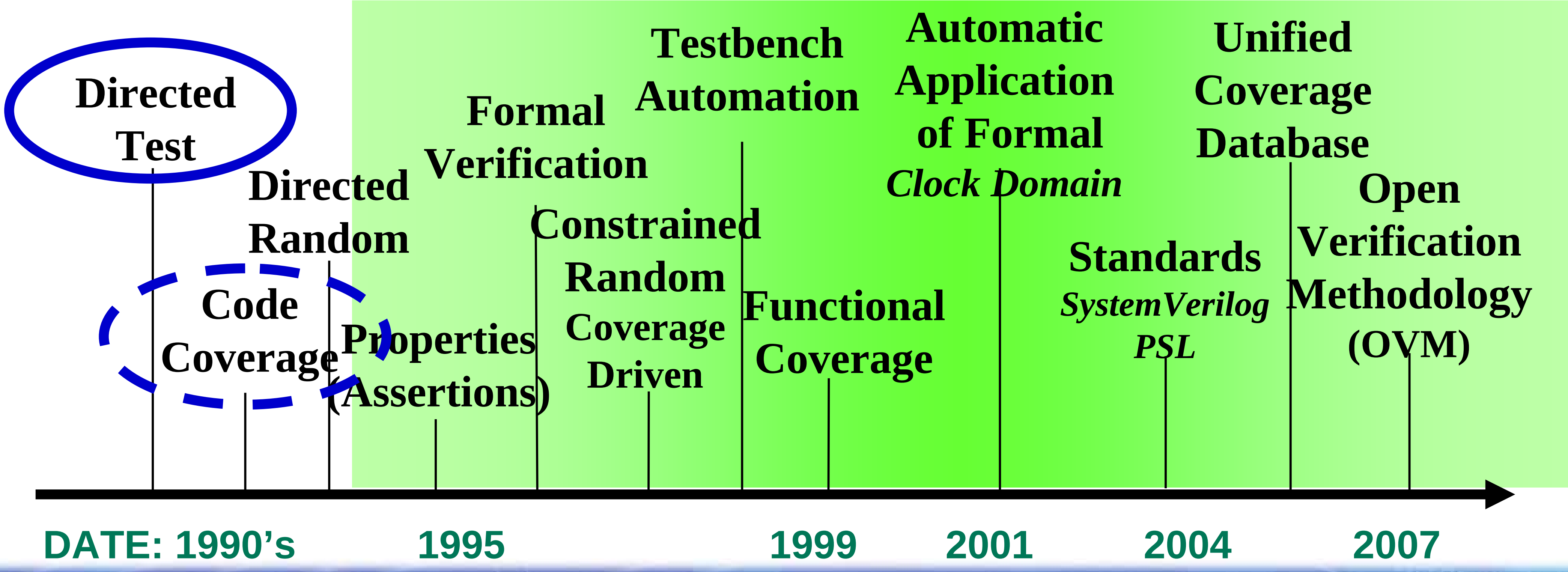


- Tests Pass
- 100% coverage

- These bugs exist, but are undetected
- Failures only appear if test propagates it to the output

Evolution of Verification Methods

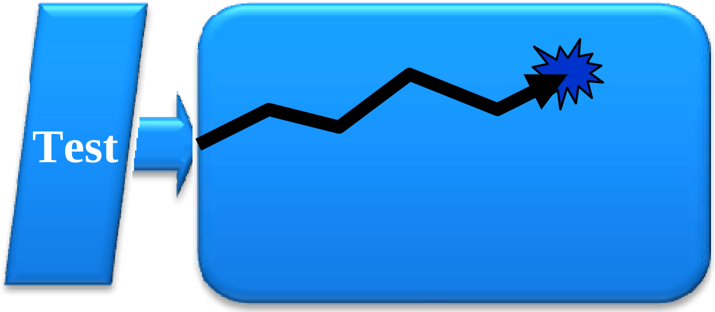
- Most aerospace companies use this traditional approach (directed test/code coverage)
  - More complex designs can benefit from newer techniques

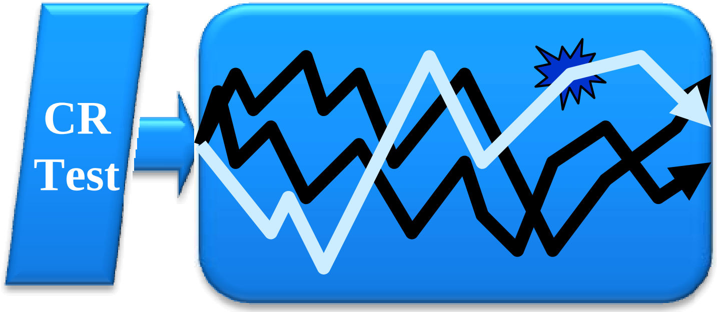


Automating Test Stimulus vs. Directed Test

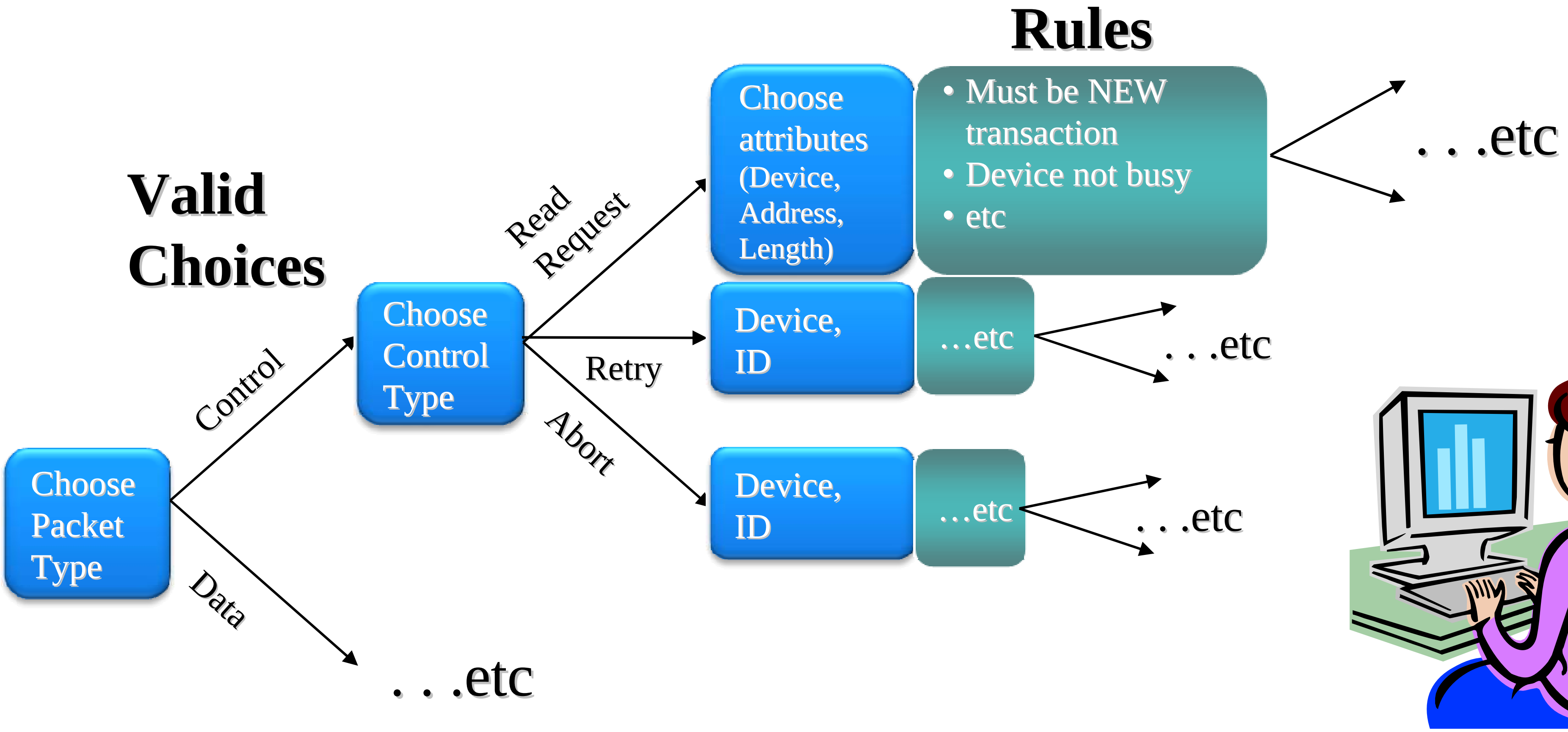
- **Directed tests:**
  - Test writer must code each specific scenario to specify intent explicitly
  - Prone to overestimating completeness of testing
  - Doesn't scale with design complexity

- **Automated test stimulus:**
  - Engine uses constraints and randomness to exercise a wide variety of possible scenarios
  - Completeness driven by progress towards functional coverage goals
  - Scales very efficiently with design complexity





How do you build an Automated Testbench?



1. **<u>Engineer</u> encodes traffic structure and rules per requirements (Testbench)**
2. **SystemVerilog Simulator then chooses paths (Stimulus), per rules (if any)**
3. **Coverage measurements assures all paths taken per requirements**

Directed Test vs. Automated Test Stimulus

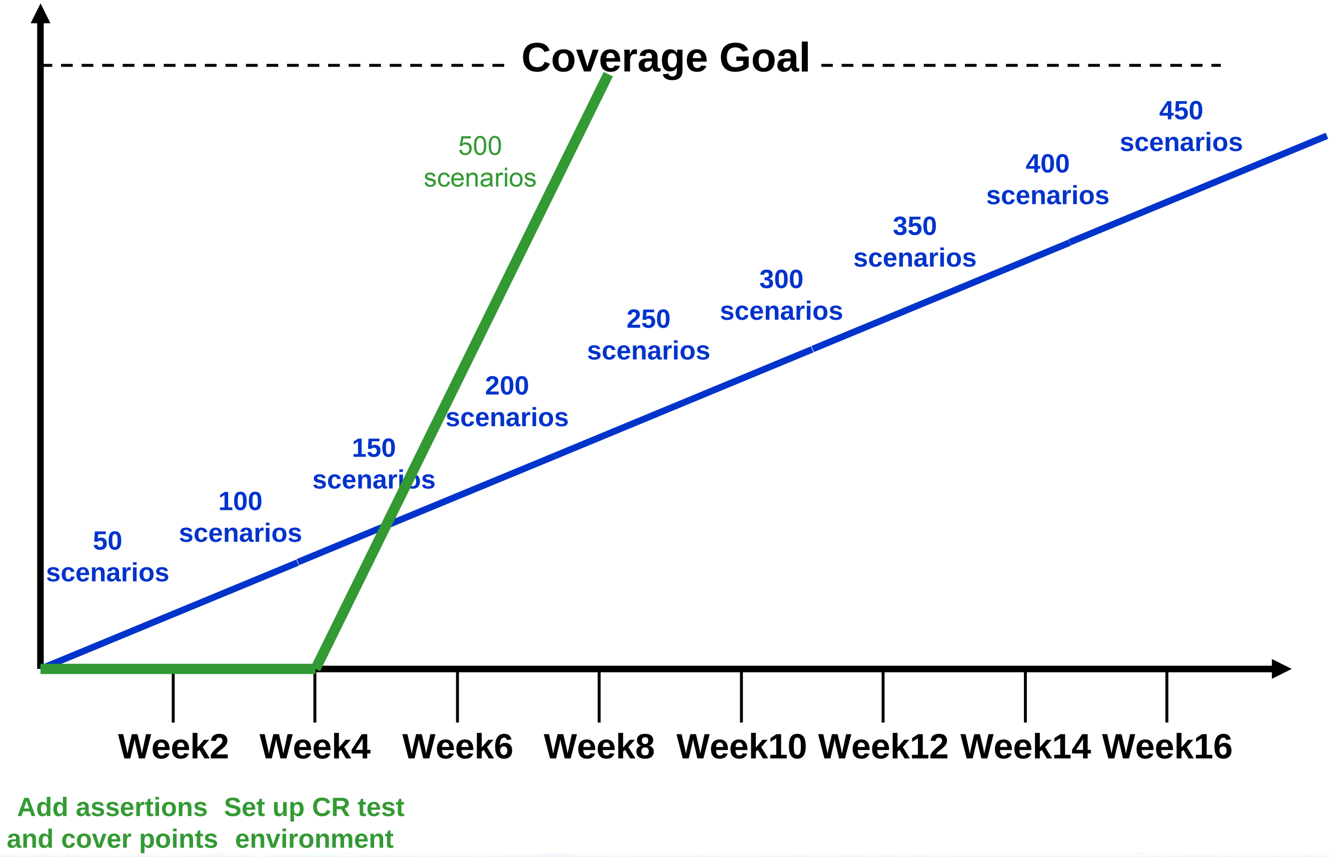
Directed Test

- 1 test/scenario (1 day each)
- Immediate progress!

Automated Test Stimulus

- Up front infrastructure
- 5X productivity increase!

5 verification engrs
500 requirements



Monitoring and Covering Requirements

Assertion Based Verification

- **Assertions are like comments that describe how the design is supposed to work (requirements)**
- **They *actively monitor the design* to ensure it does!**
- **Assertions provide traceability to requirements**

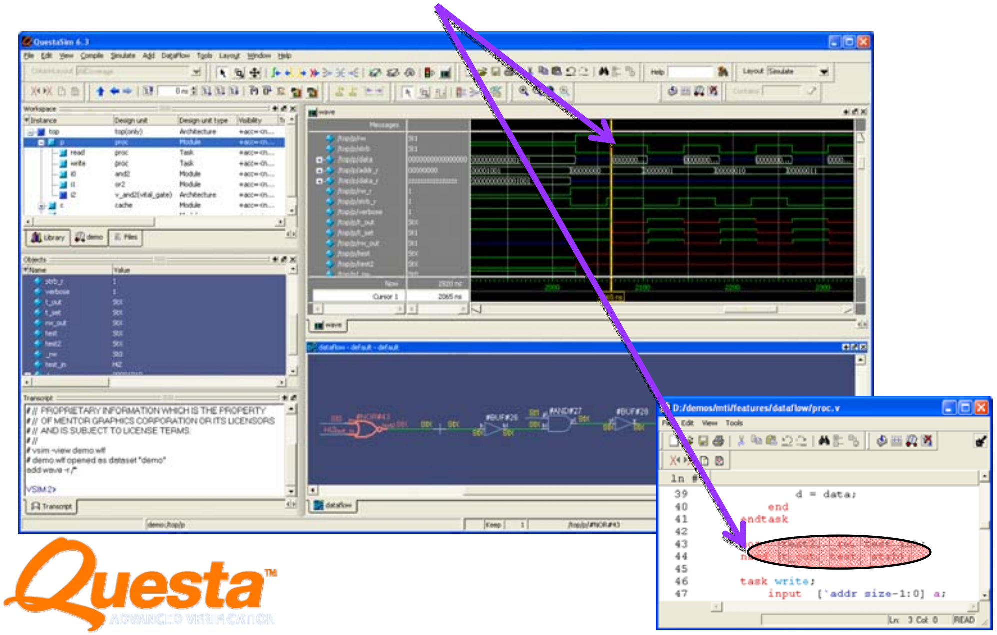
Requirement

"The flight crew shall be aurally warned if the gear is down but not locked"

Assertion

```
property RQ62_LANDING_GEAR_LOCK;
  @(posedge clk)
   GEAR_down_notification |->
   ##[1:$] Gear_down_lock_notification;
endproperty
cover property RQ62_LANDING_GEAR_LOCK;
```

Assertion Failure



```
# // PROPRIETARY INFORMATION WHICH IS THE PROPERTY
# // OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS
# // AND IS SUBJECT TO LICENSE TERMS.
# //
# vsim -view demo.wlf
# demo.wlf opened as dataset "demo"
add wave -r /*

VSIM 2>
```

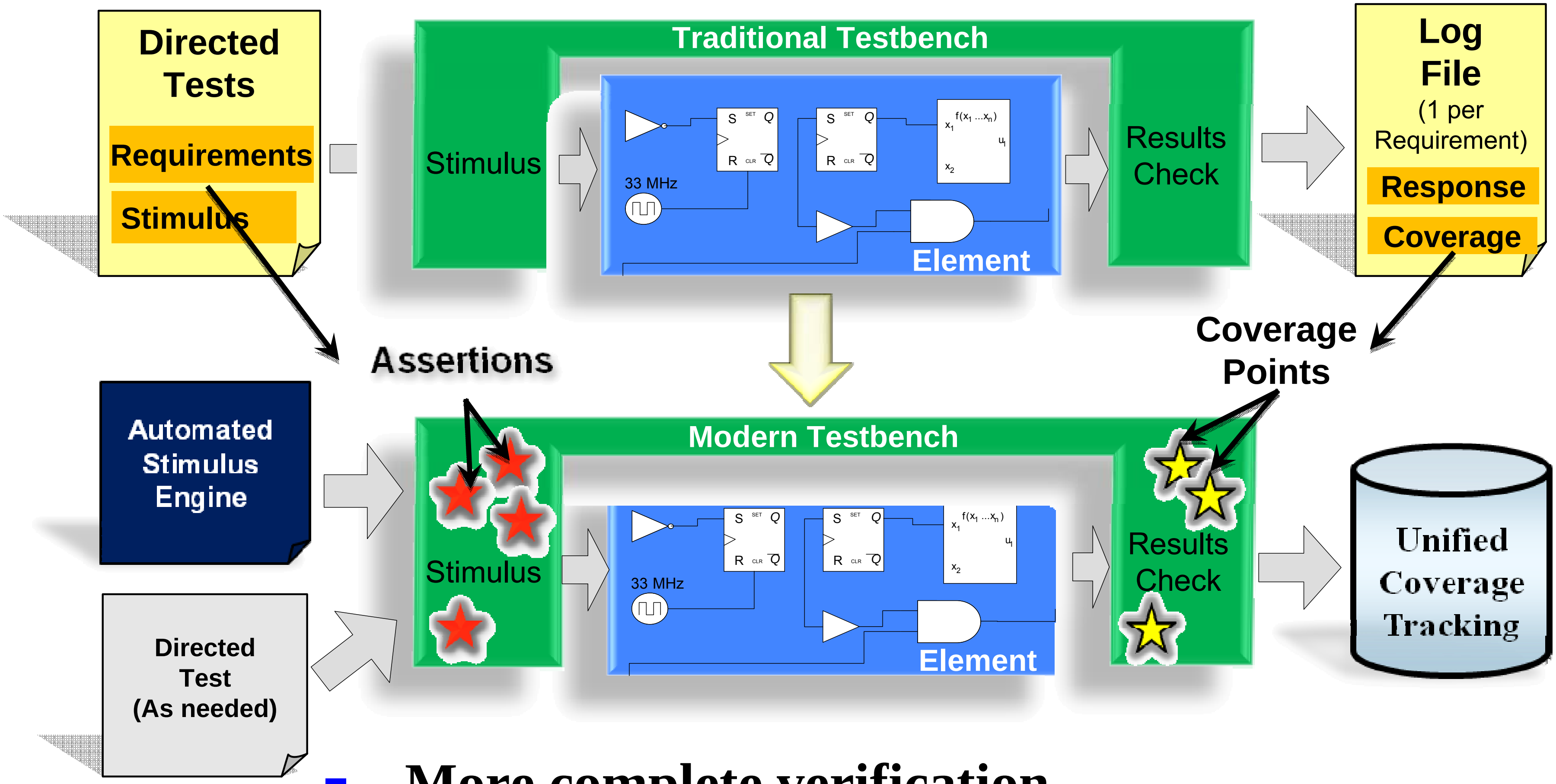
```
                d = data;
            end
        endtask

        nor (test2, _rw, test_in);
        nand (t_out, test, strb);

        task write;
            input  [`addr size-1:0] a;
```

Automated Test Generation Applied to DO-254

Modern Testbench Approach

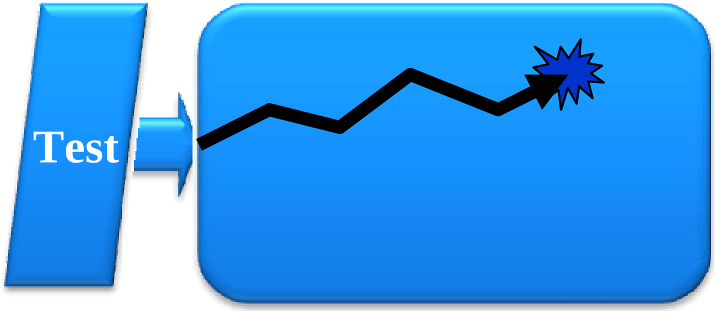


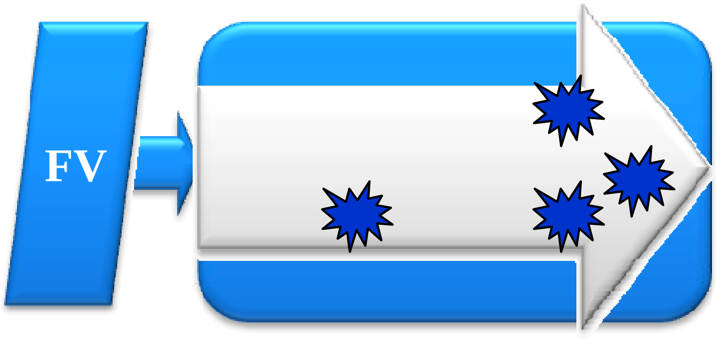
- **More complete verification**
- **Requires fewer directed tests/resources**
- **Direct link back to requirements**

Formal Methods vs. Directed Test

- Directed tests
  - Simulation-based method that requires input stimulus
  - Test writer must code a scenario that hits a bug
    - If stimulus doesn’t exercise a bug, the bug is missed

- Formal Methods
  - Mathematical analysis done on RTL --no stimulus needed
    - Assertion provides description of requirement to be checked
  - Formal engine analyzes assertion against every possible scenario (state)
    - Exhaustive!





**Note: Formal methods should be used in conjunction with (not as a replacement for) directed test and/or automated testing.*

Example: Formal Model Checking for DO-254

Exhaustively Verify Safety-Specific Requirements

- Formal Model Checking finds all possible scenarios
  - Example: enabling reverse thrusters
- Unexpected paths to this situation are called “sneak paths”
  - Is there any way for some event to happen other than the correct way?
- How to apply:
  - Add an assertion stating that the event cannot happen in implementation
  - Apply formal model checking
  - Investigate/fix all unwanted situations
  - Repeat process until no unwanted paths exist

Requirement

Reverse thrusters shall never fire in mid-air.

Assertion

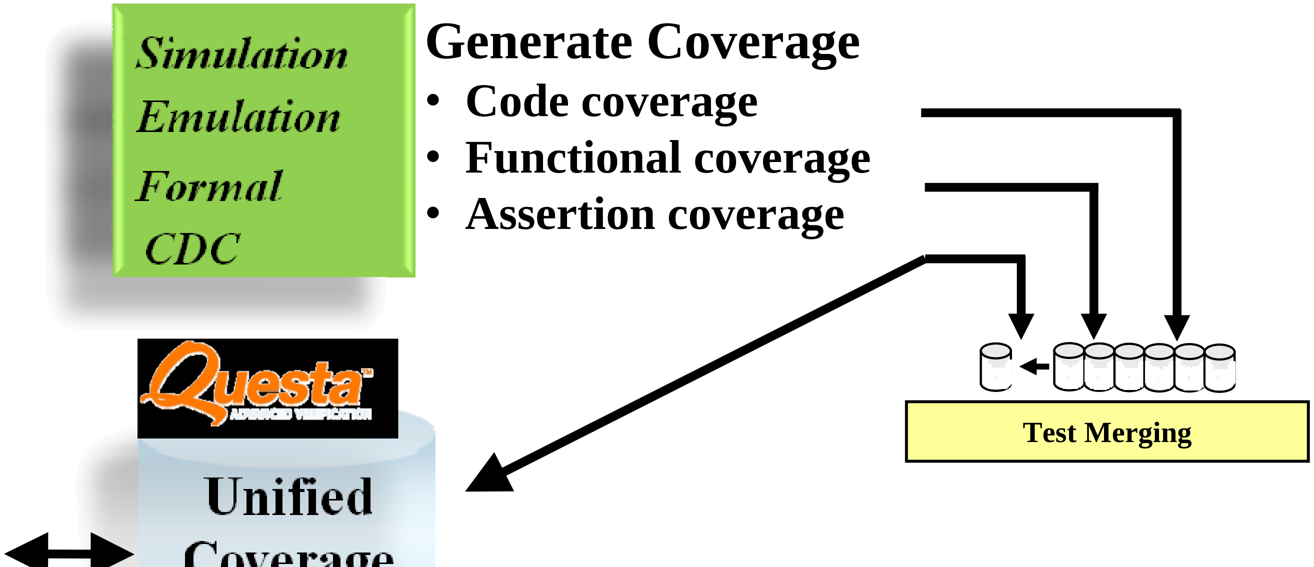
```
assert always fire_reverse_thrusters
|-> Gear_down_lock_notification
    @(clk'event and clk = '1')
```

Managing Verification for DO-254

Needs	Mentor Provides the Solution
Requirements-based test and traceability	▪ **Verification activities mapped to requirements-driven test plan with links for traceability**
Coverage	▪ **Unified coverage database to store coverage data from a variety of sources, with a variety of metrics**
Verification Mgmt and Reporting data	▪ **Verification management facilitates reporting of progress (coverage) of requirements**

Verification Management and Unified Coverage

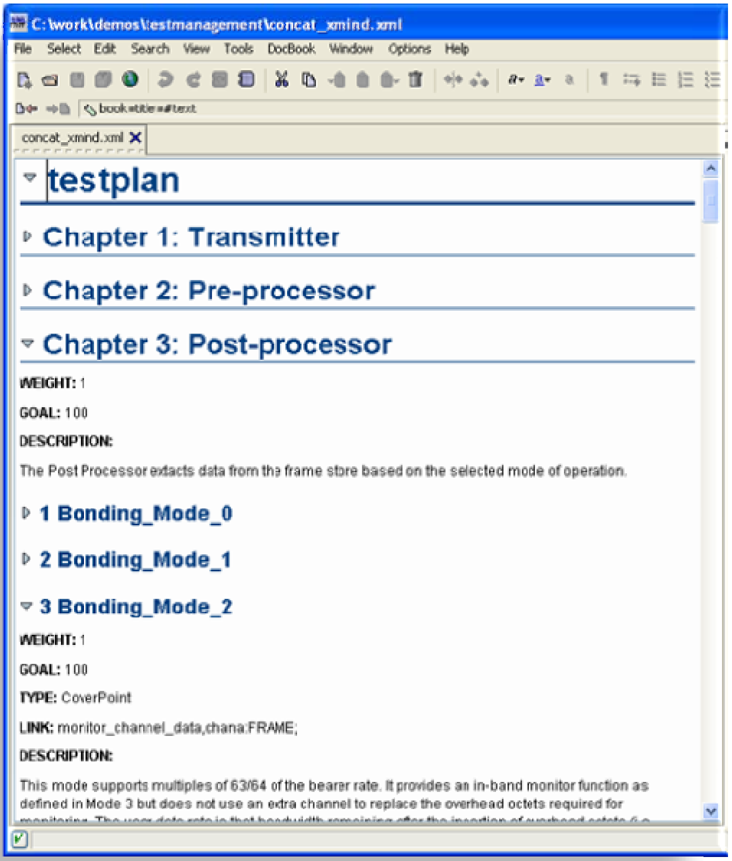
Quality, Progress and Requirements Traceability



Simulation
Emulation
Formal
CDC

Generate Coverage

- Code coverage
- Functional coverage
- Assertion coverage



C:\work\demos\testmanagement\concat_xmind.xml
File Select Edit Search View Tools DocBook Window Options Help
book#title#text
concat_xmind.xml

testplan

Chapter 1: Transmitter

Chapter 2: Pre-processor

Chapter 3: Post-processor

WEIGHT: 1

GOAL: 100

DESCRIPTION:

The Post Processor extracts data from the frame store based on the selected mode of operation.

1 Bonding_Mode_0

2 Bonding_Mode_1

3 Bonding_Mode_2

WEIGHT: 1

GOAL: 100

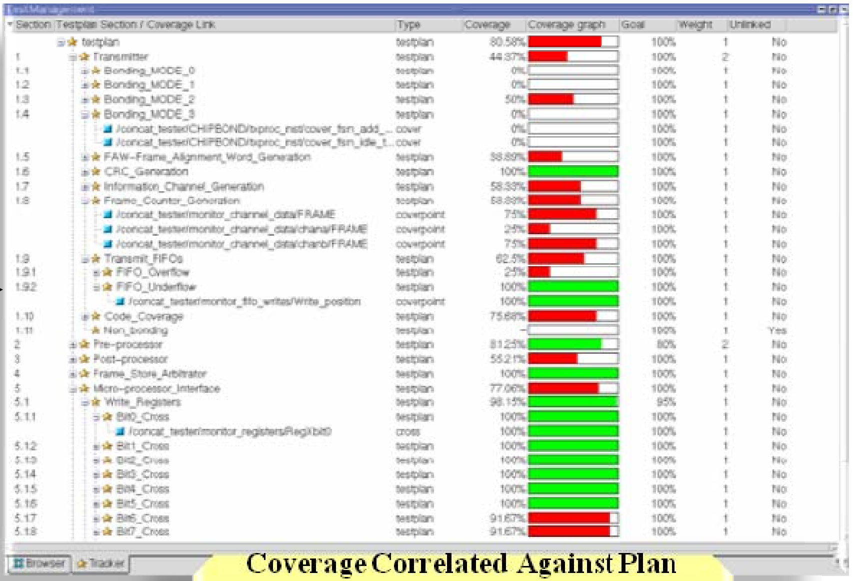
TYPE: CoverPoint

LINK: monitor_channel_data,chana:FRAME;

DESCRIPTION:

This mode supports multiples of 63/64 of the bearer rate. It provides an in-band monitor function as defined in Mode 3 but does not use an extra channel to replace the overhead octets required for

Verification Plan Import (Excel, Word, DocBook)



Section	Testplan Section / Coverage Link	Type	Coverage	Coverage graph	Goal	Weight	Unlinked
	testplan	testplan	80.58%	🟥	100%	1	No
1	Transmitter	testplan	44.37%	🟥	100%	2	No
1.1	Bonding_MODE_0	testplan	0%		100%	1	No
1.2	Bonding_MODE_1	testplan	0%		100%	1	No
1.3	Bonding_MODE_2	testplan	50%	🟥	100%	1	No
1.4	Bonding_MODE_3	testplan	0%		100%	1	No
	/concat_tester/CHIPBOND/txproc_inst/cover_fsm_add_...	cover	0%		100%	1	No
	/concat_tester/CHIPBOND/txproc_inst/cover_fsm_idle_t...	cover	0%		100%	1	No
1.5	FAW-Frame_Alignment_Word_Generation	testplan	38.89%	🟥	100%	1	No
1.6	CRC_Generation	testplan	100%	🟩	100%	1	No
1.7	Information_Channel_Generation	testplan	58.33%	🟥	100%	1	No
1.8	Frame_Counter_Generation	testplan	58.33%	🟥	100%	1	No
	/concat_tester/monitor_channel_data/FRAME	coverpoint	75%	🟥	100%	1	No
	/concat_tester/monitor_channel_data/chana/FRAME	coverpoint	25%	🟥	100%	1	No
	/concat_tester/monitor_channel_data/chanb/FRAME	coverpoint	75%	🟥	100%	1	No
1.9	Transmit_FIFOs	testplan	62.5%	🟥	100%	1	No
1.9.1	FIFO_Overflow	testplan	25%	🟥	100%	1	No
1.9.2	FIFO_Underflow	testplan	100%	🟩	100%	1	No
	/concat_tester/monitor_fifo_writes/Write_position	coverpoint	100%	🟩	100%	1	No
1.10	Code_Coverage	testplan	75.68%	🟥	100%	1	No
1.11	Non_bonding	testplan	–		100%	1	Yes
2	Pre-processor	testplan	81.25%	🟩	80%	2	No
3	Post-processor	testplan	55.21%	🟥	100%	1	No
4	Frame_Store_Arbitrator	testplan	100%	🟩	100%	1	No
5	Micro-processor_Interface	testplan	77.06%	🟥	100%	1	No
5.1	Write_Registers	testplan	98.15%	🟩	95%	1	No
5.1.1	Bit0_Cross	testplan	100%	🟩	100%	1	No
	/concat_tester/monitor_registers/Reg/bit0	cross	100%	🟩	100%	1	No
5.1.2	Bit1_Cross	testplan	100%	🟩	100%	1	No
5.1.3	Bit2_Cross	testplan	100%	🟩	100%	1	No
5.1.4	Bit3_Cross	testplan	100%	🟩	100%	1	No
5.1.5	Bit4_Cross	testplan	100%	🟩	100%	1	No
5.1.6	Bit5_Cross	testplan	100%	🟩	100%	1	No
5.1.7	Bit6_Cross	testplan	91.67%	🟥	100%	1	No
5.1.8	Bit7_Cross	testplan	91.67%	🟥	100%	1	No

Coverage Correlated Against Plan

ModelSim

Verification with Mentor

Advanced Methods

Questa

Assertions
Auto Test Stimulus
Functional Coverage
Verification Management and Unified Coverage

0-IN FUNCTIONAL VERIFICATION

Formal Verification
Clock-Domain Crossing

FormalPro

Logic Equivalency Checking

System Modeling

SystemVision

Mentor Leads in Advanced Verification

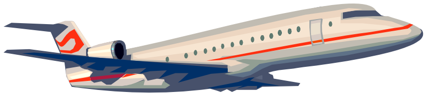
- **Actively monitor adherence to requirements**
- **Automated stimulus generation to reach many more scenarios than directed test**
- **Measure coverage against design requirements**
- **Manage and report on verification progress**
- **Mathematical analysis to exhaustively prove safety-critical requirements, …**
- **Check clock-domain crossings to eliminate metastability**
- **Assure two models are functionally equivalent**
- **Virtual lab for design and analysis of distributed mechatronic systems**

Advanced methods can improve both safety and efficiency!

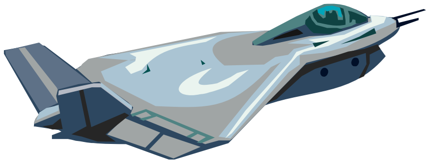
Agenda

- **Verification Challenges and Safety-Critical Design**
- **DO-254 Requirements for Verification**
- **Safety-Critical Verification: Recommendations**
- **Conclusion**

Conclusion



- **Mentor can help you establish a methodology that is efficient, reusable, and certifiable**
  - **Industry leading solutions in wide use**
  - **Supporting DO-254 objectives**
  - **Scalable methods for the simplest to the most complex safety-critical project**
- **Applying advanced methods will:**
  - **Improve verification efficiency and thoroughness**
  - **Reduce development costs**
  - **Improve safety of hardware systems**



More Information

- **Visit our web site: www.mentor.com/go/do-254**
- **Here you will find numerous resources including the following verification-related publications**
  - ***"Achieving Quality and Traceability in FPGA/ASIC Flows for DO-254 Aviation Projects"***
  - ***"The Use of Advanced Verification Methods to Address DO-254 Design Assurance"***
  - ***"Effective Functional Verification Methodologies for DO-254 Level A/B and Other Safety-Critical Devices"***
  - ***"Assessing the ModelSim Tool for Use in DO-254 and ED-80 Projects"***
  - ***"Automating Clock-Domain Crossing Verification for DO-254 (and other Safety-Critical) Designs"***
  - ***"DO-254 Compliant Design and Verification with VHDL-AMS"***

Mentor Graphics®

www.mentor.com