Jump Starting Code Development to Minimize Defects

Class 2: Managing Design Processes

December 11, 2018

Jacob Beningo

DesignNews

CEC CONTINUING EDUCATION CENTER

Presented by: Digi-Key ELECTRONICS

Course Overview

Topics:

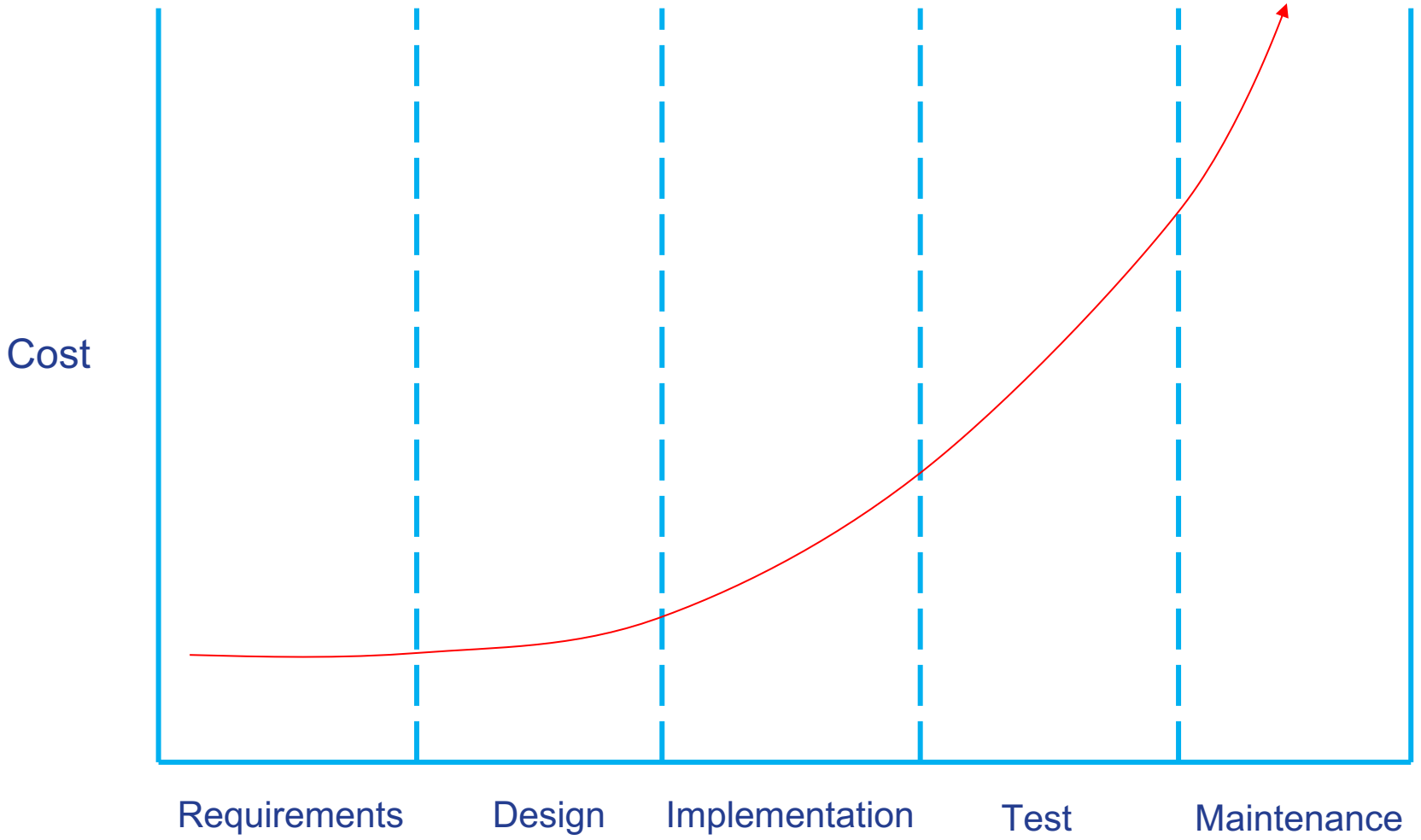
- Errors, Defects and Bugs
- **Managing Design Processes**
- The Jump Start Development Process
- Mastering Application Tracing
- Advanced Techniques

Session Overview

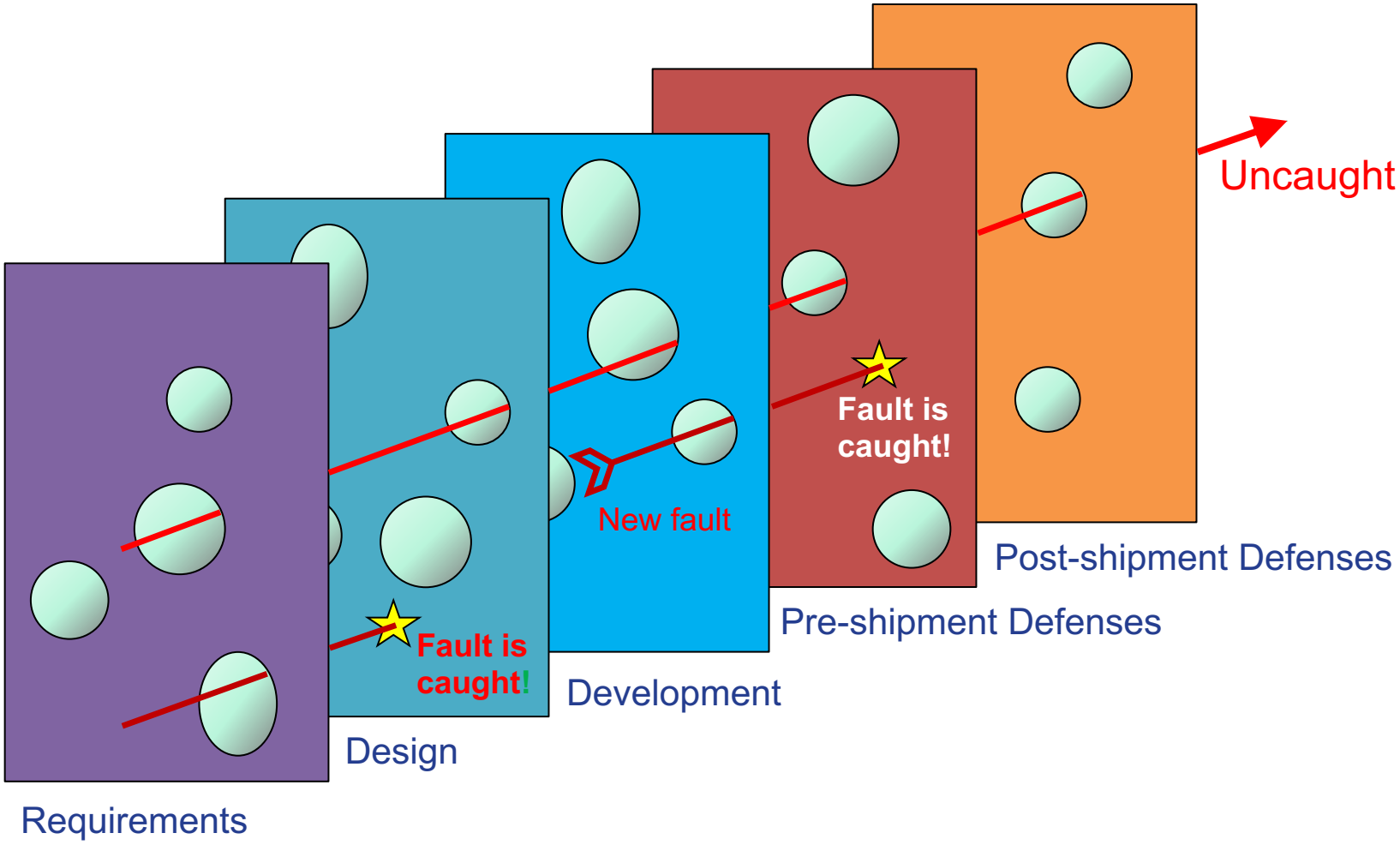
- The cost of defects
- Defect prevention
- TBD
- Design Cycle Tune-up

Presented by:

The Cost of Defects



Defect Prevention



Design Cycle Tune-up

Design Cycle Tune-up

General

- Software is developed in a modular, decoupled manner with reuse in mind
- Leverage 3rd party components to ease time and quality pressures
- Software metrics are regularly tracked and used to improve processes
- Development activities are well thought out and executed in a methodical manner

Design Cycle Tune-up

Requirements

- Clearly defined and communicated before design commences
- SRS developed and maintained
- Feature additions and changes are well documented
- Software metrics from previous projects are used to estimate time and costs

Design Cycle Tune-up

Design

- A software architecture is developed and maintained
- Flow charts, state charts and other diagrams are developed before writing a single line of production code
- Scalability and reuse are built into the software design
- Application models are built and tested to prove out the design before construction

Design Cycle Tune-up

Construction

- Code reviews are regularly scheduled and held
- Static code analysis is performed with each revision of committed software
- We use a detailed style guide so that all software is uniform
- We utilize doxygen to or other documentation tools to generate documentation automatically with each version

Design Cycle Tune-up

Testing

- Unit tests are created and performed with each committed firmware version
- Failures, faults, and defects are clearly documented and assigned priority for resolution
- System level test cases are traceable back to the original requirement that spawned the feature
- Complete test coverage of the code base is clearly executed and documented before release

Design Cycle Tune-up

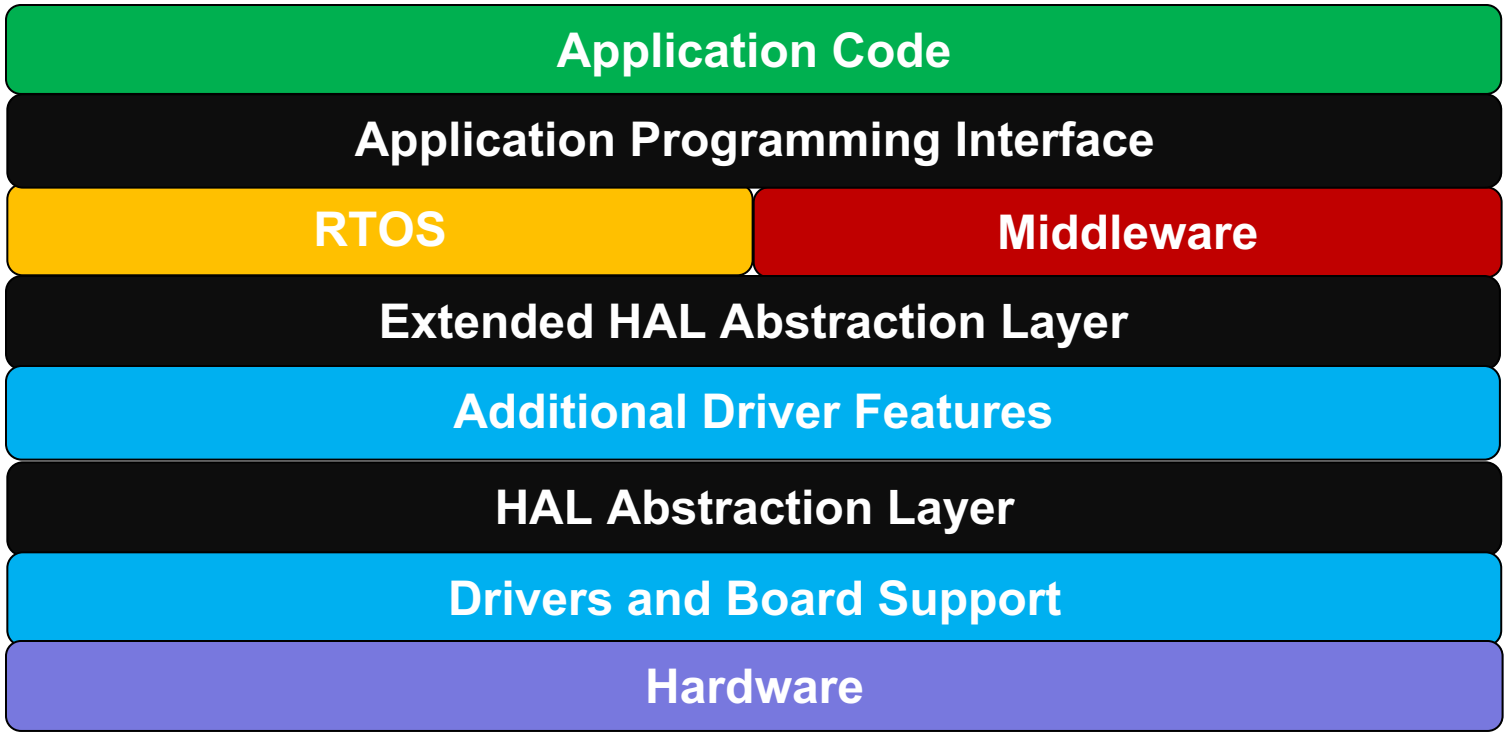
20 Questions

- Ranked 1 – 5
- Total score 100

What area was the weakest?

What area was the strongest?

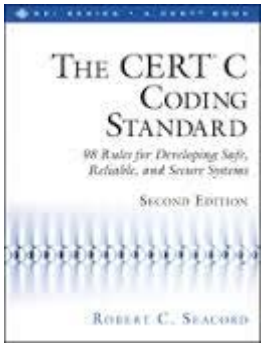
Design Process Tip #1

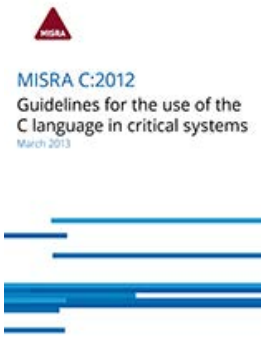


Design Process Tip #2

The Purpose of a Coding Standard

- Consistency (Same style for all developers)
- Readability (Improves maintainability)
- Acceptance of language “inclusions”
- Project Organization
- Provide guard rails for “safe” programming

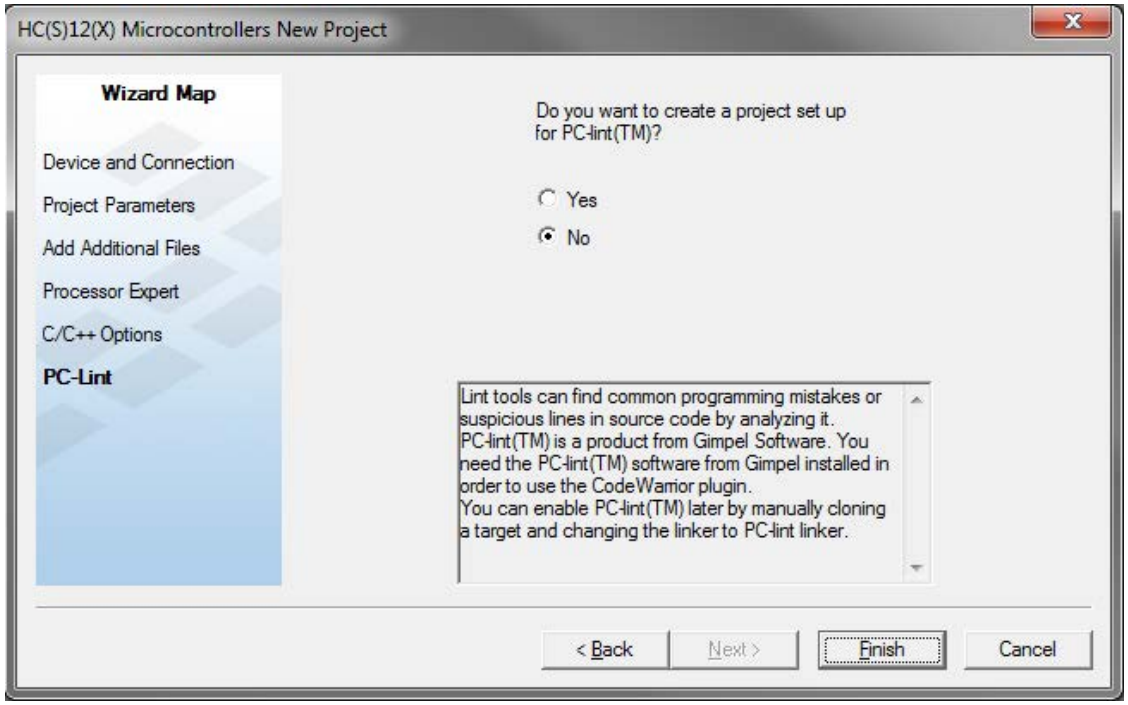




Design Process Tip #3

Ways to Perform Code Analysis

- Complexity Measurements
- Lines of Code
- Comment Density
- Assertion Density
- Static Code Analysis
- Dynamic Code Analysis
- Worst Case Stack Usage
- Automated Tools (i.e. Code Standard Compliance)



Design Process Tip #4

Limiting Function Complexity

- Helps to ensure readability
- Bounds number of test cases
- Forces breaking up into smaller more manageable pieces
- Reduces bugs

~~ File Functional Summary ~~

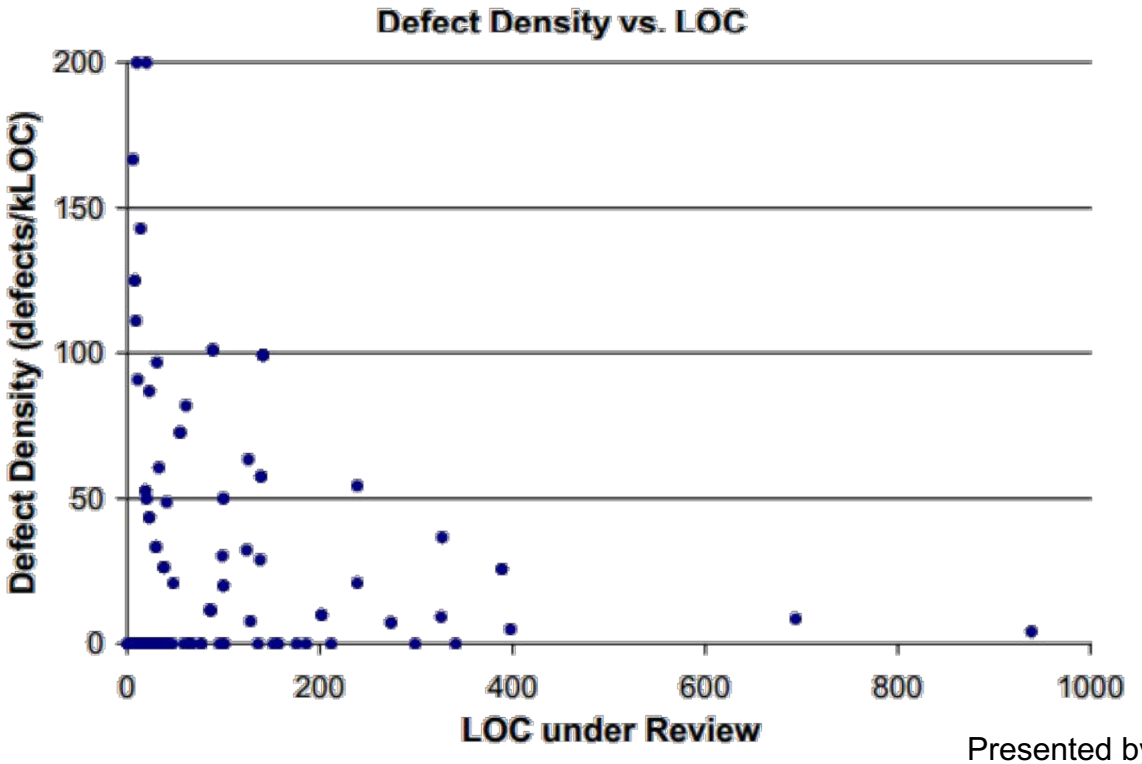
File Function Count....:	5		
Total Function LOC.....:	35	Total Function Pts LOC :	0.5
Total Function eLOC....:	17	Total Function Pts eLOC:	0.3
Total Function lLOC....:	14	Total Function Pts lLOC:	0.2
Total Function Params .:	7	Total Function Return .:	5
Total Cyclo Complexity :	8	Total Function Complex.:	20
Max Function LOC .....:	21	Average Function LOC ..:	7.00
Max Function eLOC .....:	11	Average Function eLOC .:	3.40
Max Function lLOC .....:	8	Average Function lLOC .:	2.80
Max Function Parameters:	2	Avg Function Parameters:	1.40
Max Function Returns ..:	1	Avg Function Returns ..:	1.00
Max Interface Complex. :	3	Avg Interface Complex. :	2.40
Max Cyclomatic Complex.:	4	Avg Cyclomatic Complex.:	1.60
Max Total Complexity ..:	6	Avg Total Complexity ..:	4.00

End of File: C:\SPO2 Module\Common\drivers\src\pwm.c

Design Process Tip #5

A few tips for performing a code review

- Pace - less than 300 LOC / hour
- Keep to 60 – 90 minutes
- Expect 15 defects / hour
- Inspection rates will vary



Additional Resources

- Download Course Material for
  - C/C++ Doxygen Templates
  - Example source code
  - Blog
  - YouTube Videos
- Embedded Bytes Newsletter
  - http://bit.ly/1BAHYXm

From www.beningo.com under

- Blog > CEC – Jump Starting Code Development to Minimize Defects