Mentor Graphics
Predictability through Agility

Using Agile to Improve the FPGA Development Process

Nigel Elliot
Digital Design & Verification
AE Group Leader

November 2015
Many FPGA Projects Miss Schedule

- 2012: 67% behind schedule
- 2014: 59% behind schedule

Actual FPGA design completion compared to FPGA project's original schedule

More Effort Spent on Verification

- **FPGA Verification Consumes Majority of Project Time**
  - 2012: Average 43%
  - 2014: Average 46%

- **Number of Peak FPGA Engineers Increasing**
  - 4.9% CAGR for FPGA design engineers
  - 20.9% CAGR for FPGA verification engineers

- **Where FPGA Designers Spend Their Time**
  - 2012: 2.6% doing verification, 4.0% doing design
  - 2014: 51% doing verification, 49% doing design
FPGA Verification Trends

- Majority using functional and code coverage
- Increasing amount leveraging power of constrained random verification
- We will look at how using an iterative approach can provide predictability for these efforts
Sources of Scheduling Uncertainty

Even running simulations can be difficult to schedule

How can you accurately schedule for debug?

Significant time spent debugging

Changes or problems with specification

Root Cause of FPGA Functional Flaws


Root Cause of FPGA Functional Flaws

DESIGN ERROR
CHANGES IN
SPECIFICATION
INCORRECT or
INCOMPLETE
SPECIFICATION
FLAW IN
INTERNAL
REUSED BLOCK,
CELL, MEGACELL
or IP
FLAW IN
EXTERNAL IP
BLOCK or
TESTBENCH
OTHER

2012
2014

DESIGN Project

Root Cause of FPGA Functional Flaws * Multiple answers possible


Mentor Graphics Corp. Company Confidential
www.mentor.com

Agile for FPGA
Scheduling Predictability

Why is it difficult to create accurate schedules?

**Low**
- Debug
- Bug fixes
- Verification closure
- Hardware bring-up
- Hardware validation

**Medium**
- Defining requirements
- Synthesis
- Timing closure
- Implementation

**High**
- Write spec
- Write RTL
- Write TB
- Write constraints

*Complete When Fully Tested*  
*Complete When Done*
Agile Overview

- **Waterfall**
  - Sequential – going back to previous steps can be difficult and costly
  - One big bang, production ready release at the end of the project
  - Relies heavily on initial requirements
  - Most testing occurs near the end of the project
  - Lessons learned only for the next project
  - Task driven development

- **Agile**
  - Adaptive and iterative process
  - Many production ready releases during project based on most important features
  - Allows for changes throughout process
  - Testing happens often and early
  - Many opportunities for feedback
  - Value driven development with every iteration
Predictive vs Adaptive Process

*Waterfall vs Agile*

**Predictive Process (Waterfall)**

The plan creates the schedule and cost estimates.

**Adaptive Process (Agile)**

The vision drives feature estimates.
Predictability through Agility

- Leverage iterative nature of Agile to take the highest priority features from inception to completion
- Focus on the customer’s most important features for each iteration and revisit with each new iteration
- For each target feature in the iteration, fully design, verify, implement and test in hardware
- Fully develop features to spread difficult to schedule tasks across the project
- Fully develop features to improve visibility of nebulous tasks as well as end of project cloudiness
- Use lessons learned from previous iterations to better predict future iterations
- Pipe clean tools and process early to streamline future iterations
Adapting to Change with Agile

Waterfall

Specification → Design → Verification → Implementation → Hardware

→ Competition Changes → Requirement Changes → Timeline Changes → Market Changes → Technology Changes

Agile

→ Specification → Design → Hardware → Implementation → Verification

→ Specification → Hardware → Design → Implementation → Verification

→ Specification → Hardware → Implementation → Verification

Mentor Graphics Corp. Company Confidential
www.mentor.com
Visibility Considerations

Waterfall vs Agile

Waterfall

Specification

Design

Verification

Implementation

Hardware

50% Done?

Agile

Specification

Design

Verification

Implementation

Hardware

Feature 1

Feature 2

Feature 3

Feature 4

Feature n

50% Done
Agile FPGA Advantage
FPGAs a natural fit for Agile over ASICs

- Waterfall often must be used in cases where changes are costly (e.g. ASIC tape-out)
- FPGA development flows provides software like low cost changes
- FPGA’s re-programmability inherently enable fast changes throughout the flow
- Leverage custom boards or off-the-shelf FPGA reference boards
Requirement Changes

Waterfall

Project Timeline

Specification → Design → Verification → Implementation → Hardware

Agile

Assess Requirements → Specification → Design → Implementation → Verification → Requirement Changes

Feature 1, Feature 2, Feature 3

Feature 4, Feature 5, Feature 6

Feature 7, Feature 8, Feature 9

A

ssess

Reassess Requirements

Mentor Graphics Corp. Company Confidential
www.mentor.com
Quality Considerations

Waterfall vs Agile

Waterfall

Specification → Design → Verification → Implementation → Hardware

Project Timeline

New Project Timeline

Agile

Specification → Design → Verification → Implementation → Hardware

Feature 1 → Feature 2 → Feature 3 → Feature 4 → Feature 5 → Feature 6

Out of Time or Money

Agile for FPGA

Mentor Graphics Corp. Company Confidential
www.mentor.com
When is Verification Done?

*Coverage Closure Is Generally Not Linear...*

- Difficult to predict for accurate scheduling
- Challenging to understand state of progress and time left
- 80% coverage closure does not mean 20% effort remains
- It is more likely that the last 20% of coverage takes 80% of time!

**Coverage closure requires:**
- Adding new tests
- Evaluating and modifying coverage
- Tuning randomization constraints
- Fixing bugs uncovered during closure
## How Much Effort is Left?

### Verification Management Tracker

<table>
<thead>
<tr>
<th>Sec#</th>
<th>Testplan</th>
<th>Section / Coverage Link</th>
<th>Type</th>
<th>Coverage</th>
<th>Status</th>
<th>Bins</th>
<th>Hits</th>
<th>Link Status</th>
<th>% of Goal</th>
<th>Weight</th>
<th>% Hit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>testplan</td>
<td>Wishbone Interfaces</td>
<td>Testplan</td>
<td>58.34%</td>
<td>Partial</td>
<td>638</td>
<td>333</td>
<td></td>
<td>58.34%</td>
<td>1</td>
<td>52.19%</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>MAC2COP</td>
<td>Testplan</td>
<td>52.22%</td>
<td>Partial</td>
<td>54</td>
<td>26</td>
<td></td>
<td>52.22%</td>
<td>5</td>
<td>48.14%</td>
</tr>
<tr>
<td>1.1</td>
<td></td>
<td>COP2MAC</td>
<td>Testplan</td>
<td>100%</td>
<td>Unlinked</td>
<td>0</td>
<td>0</td>
<td></td>
<td>100%</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>1.2</td>
<td></td>
<td>Wishbone Operation Types</td>
<td>Testplan</td>
<td>40%</td>
<td>Clean</td>
<td>30</td>
<td>9</td>
<td></td>
<td>40%</td>
<td>1</td>
<td>30%</td>
</tr>
<tr>
<td>1.2.1</td>
<td></td>
<td>Wishbone Operation Mode</td>
<td>Testplan</td>
<td>100%</td>
<td>Clean</td>
<td>2</td>
<td>2</td>
<td></td>
<td>100%</td>
<td>1</td>
<td>100%</td>
</tr>
<tr>
<td>1.2.2</td>
<td></td>
<td>Wishbone Cycle Termination</td>
<td>Testplan</td>
<td>25%</td>
<td>Clean</td>
<td>4</td>
<td>1</td>
<td></td>
<td>25%</td>
<td>1</td>
<td>25%</td>
</tr>
<tr>
<td>1.2.3</td>
<td></td>
<td>Wishbone Burst Mode Wrap</td>
<td>Testplan</td>
<td>25%</td>
<td>Clean</td>
<td>4</td>
<td>1</td>
<td></td>
<td>25%</td>
<td>1</td>
<td>25%</td>
</tr>
<tr>
<td>1.2.4</td>
<td></td>
<td>cop2mac wishbone cycles</td>
<td>Testplan</td>
<td>25%</td>
<td>Clean</td>
<td>16</td>
<td>4</td>
<td></td>
<td>25%</td>
<td>1</td>
<td>25%</td>
</tr>
<tr>
<td>1.3</td>
<td></td>
<td>Wishbone Tx FIFO</td>
<td>Testplan</td>
<td>75%</td>
<td>Clean</td>
<td>12</td>
<td>10</td>
<td></td>
<td>75%</td>
<td>1</td>
<td>83.33%</td>
</tr>
<tr>
<td>1.4</td>
<td></td>
<td>Wishbone Rx FIFO</td>
<td>Testplan</td>
<td>41.66%</td>
<td>Clean</td>
<td>12</td>
<td>7</td>
<td></td>
<td>41.66%</td>
<td>1</td>
<td>58.33%</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>MIL Interface Operation</td>
<td>Testplan</td>
<td>52.46%</td>
<td>Clean</td>
<td>295</td>
<td>120</td>
<td></td>
<td>52.46%</td>
<td>2</td>
<td>40.67%</td>
</tr>
<tr>
<td>2.1</td>
<td></td>
<td>TX Errors</td>
<td>Testplan</td>
<td>55.84%</td>
<td>Clean</td>
<td>23</td>
<td>16</td>
<td></td>
<td>55.84%</td>
<td>1</td>
<td>69.56%</td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td>[RX / TX Payload sizes]</td>
<td>Testplan</td>
<td>49.07%</td>
<td>Clean</td>
<td>272</td>
<td>104</td>
<td></td>
<td>49.07%</td>
<td>1</td>
<td>38.23%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Buffer Descriptors</td>
<td>Testplan</td>
<td>49.56%</td>
<td>Clean</td>
<td>174</td>
<td>83</td>
<td></td>
<td>49.56%</td>
<td>1</td>
<td>47.7%</td>
</tr>
<tr>
<td>3.1</td>
<td></td>
<td>BD Diagnostic Read and Write</td>
<td>Testplan</td>
<td>50%</td>
<td>Clean</td>
<td>10</td>
<td>2</td>
<td></td>
<td>50%</td>
<td>1</td>
<td>20%</td>
</tr>
<tr>
<td>3.2</td>
<td></td>
<td>TxIDxs</td>
<td>Testplan</td>
<td>48.69%</td>
<td>Clean</td>
<td>154</td>
<td>76</td>
<td></td>
<td>48.69%</td>
<td>1</td>
<td>49.35%</td>
</tr>
<tr>
<td>3.3</td>
<td></td>
<td>RxIDxs</td>
<td>Testplan</td>
<td>50%</td>
<td>Clean</td>
<td>10</td>
<td>5</td>
<td></td>
<td>50%</td>
<td>1</td>
<td>50%</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Code_Coverage</td>
<td>Testplan</td>
<td>84.52%</td>
<td>Clean</td>
<td>2</td>
<td>0</td>
<td></td>
<td>84.52%</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Misc</td>
<td>Testplan</td>
<td>41.66%</td>
<td>Clean</td>
<td>14</td>
<td>5</td>
<td></td>
<td>41.66%</td>
<td>1</td>
<td>35.71%</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Directed Tests</td>
<td>Testplan</td>
<td>100%</td>
<td>Clean</td>
<td>99</td>
<td>99</td>
<td></td>
<td>100%</td>
<td>1</td>
<td>100%</td>
</tr>
<tr>
<td>6.1</td>
<td></td>
<td>Register Read &amp; Writes</td>
<td>Testplan</td>
<td>100%</td>
<td>Clean</td>
<td>94</td>
<td>94</td>
<td></td>
<td>100%</td>
<td>1</td>
<td>100%</td>
</tr>
<tr>
<td>6.2</td>
<td></td>
<td>Register Reset Reads</td>
<td>Testplan</td>
<td>100%</td>
<td>Clean</td>
<td>5</td>
<td>5</td>
<td></td>
<td>100%</td>
<td>1</td>
<td>100%</td>
</tr>
</tbody>
</table>
Coverage Closure Scheduling Challenges

High level view

Which project is further along?

<table>
<thead>
<tr>
<th>Project #1</th>
<th>Project #2</th>
<th>Project #3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Testplan Section</strong></td>
<td><strong>Testplan Section</strong></td>
<td><strong>Testplan Section</strong></td>
</tr>
<tr>
<td><strong>Verification Status</strong></td>
<td><strong>Verification Status</strong></td>
<td><strong>Verification Status</strong></td>
</tr>
<tr>
<td>Testbench</td>
<td>Testbench</td>
<td>Testbench</td>
</tr>
<tr>
<td>80%</td>
<td>80%</td>
<td>80%</td>
</tr>
</tbody>
</table>

Which project is further along?
### Coverage Closure Scheduling Challenges

Which project is further along?

<table>
<thead>
<tr>
<th>Project #1</th>
<th>Project #2</th>
<th>Project #3</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="#" alt="Testplan Section" /></td>
<td><img src="#" alt="Testplan Section" /></td>
<td><img src="#" alt="Testplan Section" /></td>
</tr>
<tr>
<td><strong>Testbench</strong></td>
<td><strong>80%</strong></td>
<td><strong>80%</strong></td>
</tr>
<tr>
<td><strong>Block A</strong></td>
<td><strong>80%</strong></td>
<td><strong>90%</strong></td>
</tr>
<tr>
<td><strong>Block B</strong></td>
<td><strong>80%</strong></td>
<td><strong>90%</strong></td>
</tr>
<tr>
<td><strong>Block C</strong></td>
<td><strong>80%</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

#### Furthest Along
- Full Tested
- Highest Priority
- Releasable Products
### Agile Coverage Closure

**Improving schedule predictability**

- Fully close highest priority features
- Removes uncertainty associated with coverage closure
- Avoid “uncertain” tasks associated with that functionality once at 100%
- Learn overall effort to close items and apply that to future features

<table>
<thead>
<tr>
<th>Testplan Section</th>
<th>Verification Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testbench</td>
<td>80%</td>
</tr>
<tr>
<td>Block A</td>
<td>80%</td>
</tr>
<tr>
<td>Feature A-1</td>
<td>80%</td>
</tr>
<tr>
<td>Feature A-2</td>
<td>80%</td>
</tr>
<tr>
<td>Feature A-3</td>
<td>80%</td>
</tr>
<tr>
<td>Feature A-4</td>
<td>80%</td>
</tr>
<tr>
<td>Block B</td>
<td>80%</td>
</tr>
<tr>
<td>Feature B-1</td>
<td>80%</td>
</tr>
<tr>
<td>Feature B-2</td>
<td>80%</td>
</tr>
<tr>
<td>Feature B-3</td>
<td>80%</td>
</tr>
<tr>
<td>Feature B-4</td>
<td>80%</td>
</tr>
<tr>
<td>Block C</td>
<td>80%</td>
</tr>
<tr>
<td>Feature C-1</td>
<td>80%</td>
</tr>
<tr>
<td>Feature C-2</td>
<td>80%</td>
</tr>
<tr>
<td>Feature C-3</td>
<td>80%</td>
</tr>
<tr>
<td>Feature C-4</td>
<td>80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Testplan Section</th>
<th>Verification Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testbench</td>
<td>80%</td>
</tr>
<tr>
<td>Block A</td>
<td>90%</td>
</tr>
<tr>
<td>Feature A-1</td>
<td>100%</td>
</tr>
<tr>
<td>Feature A-2</td>
<td>85%</td>
</tr>
<tr>
<td>Feature A-3</td>
<td>85%</td>
</tr>
<tr>
<td>Feature A-4</td>
<td>90%</td>
</tr>
<tr>
<td>Block B</td>
<td>90%</td>
</tr>
<tr>
<td>Feature B-1</td>
<td>80%</td>
</tr>
<tr>
<td>Feature B-2</td>
<td>90%</td>
</tr>
<tr>
<td>Feature B-3</td>
<td>95%</td>
</tr>
<tr>
<td>Feature B-4</td>
<td>95%</td>
</tr>
<tr>
<td>Block C</td>
<td>60%</td>
</tr>
<tr>
<td>Feature C-1</td>
<td>70%</td>
</tr>
<tr>
<td>Feature C-2</td>
<td>60%</td>
</tr>
<tr>
<td>Feature C-3</td>
<td>40%</td>
</tr>
<tr>
<td>Feature C-4</td>
<td>50%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Testplan Section</th>
<th>Verification Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testbench</td>
<td>80%</td>
</tr>
<tr>
<td>Block A</td>
<td>100%</td>
</tr>
<tr>
<td>Feature A-1</td>
<td>100%</td>
</tr>
<tr>
<td>Feature A-2</td>
<td>100%</td>
</tr>
<tr>
<td>Feature A-3</td>
<td>100%</td>
</tr>
<tr>
<td>Feature A-4</td>
<td>100%</td>
</tr>
<tr>
<td>Block B</td>
<td>100%</td>
</tr>
<tr>
<td>Feature B-1</td>
<td>100%</td>
</tr>
<tr>
<td>Feature B-2</td>
<td>100%</td>
</tr>
<tr>
<td>Feature B-3</td>
<td>100%</td>
</tr>
<tr>
<td>Feature B-4</td>
<td>100%</td>
</tr>
<tr>
<td>Block C</td>
<td>40%</td>
</tr>
<tr>
<td>Feature C-1</td>
<td>80%</td>
</tr>
<tr>
<td>Feature C-2</td>
<td>40%</td>
</tr>
<tr>
<td>Feature C-3</td>
<td>20%</td>
</tr>
<tr>
<td>Feature C-4</td>
<td>20%</td>
</tr>
</tbody>
</table>
Agile Timing Closure

- Waterfall approach can make it difficult to make changes in earlier phases
- Agile’s iterative approach encourages continuous development throughout all phases
- Revisiting early stages of design can result in higher impact
- Requires the whole process to be repeated (e.g. verification)
- Tuning timing as new features are added improves predictability of timing closure by breaking the problem into smaller pieces

Figure 1-4: Impact of Design Changes Throughout the Flow
Agile Timing Closure

**Waterfall**

**Specification** → **Design** → **Verification** → **Implementation** → **Hardware**

**Agile**

**Specification** → **Design** → **Verification** → **Implementation** → **Hardware**

- Feature 1
- Feature 2
- Feature 3
- Feature 4
- Feature 5
- Feature 6
- Feature 7
- Feature 8
- Feature 9

Timing Closure Problems

Hardware

Design

Verification

Implementation

Mentor Graphics Corp. Company Confidential
www.mentor.com
Summarizing Waterfall vs Agile
Summary
Advantages of applying iterative development

- FPGAs well suited to Agile
- Improved predictability
- Improved visibility
- Improved quality
- Reduced risk
- Improved adaptability

Waterfall
- Success: 14%
- Challenged: 57%
- Failed: 29%

Agile
- Success: 43%
- Challenged: 48%
- Failed: 9%

Source: The CHAOS Manifesto, The Standish Group, 2012

Realities of FPGA Projects

#1. It is impossible to gather all the requirements at the beginning of a project.

#2. Whatever requirements you do gather are guaranteed to change.

#3. There will always be more to do than time and money will allow.

Challenged:
- Late
- Over budget
- Lacking functionality
- Low quality