bitvis

1100010110100111101001110110110100111110011

Achieving the right FPGA design quality –

Could Reviews save the day?

Bitvis

- Independent Design Centre for SW and FPGA (& ASIC)
- Located in Asker
- 10 designers – and increasing….
- Building on the Digitas legacy
  - Efficiency/Quality → Methodology
  - Customer partnership relation
- Concept, Specification, Architecture, Implementation, Verification, Test
- Methodology, Reviews, Sparring partner
- Project development partner
  - *Kongsberg Devotek, Informasjonskontroll, Ant Micro*
  - Enables large projects of complete embedded products

Quality in every bit

- Coordinated and real improvement in the design flow
  - Not just a paper exercise, but competence, tools and IP
- Customer focused
  - A real partnership – with common goals
  - Quality of service – rather than dependency

FPGA & ASIC

- Major experience with FPGA and ASIC
- SW developers with a good understanding of HW & FPGA
- Good overview of pitfalls, time wasters and quality issues
  - ➔ A structured and layered design approach
  - ➔ Thorough and structured verification
  - ➔ Focus on critical issues like timing/clocking/CDC
  - ➔ Documenting the important issues
  - ➔ Focus on understanding and modifiability
- Development Review and Sparring
- *'FPGA development Best Practices'* – Course on demand

FPGA Best Practices

- Bitvis offers the well established course: "FPGA Development Best Practices" (see http://bitvis.no/)
  - Based on the Digitas course previously presented in DK, SE, NO
  - Two full days of learning
  - Major focus on how to improve your FPGA-based projects
    - Design Structure and Methodology
    - Verification Structure and Methodology
    - Clocking and timing
    - Focus on Efficiency and Quality improvements – that can be applied today
  - Very good feedback from participants in all countries
    - So far attended by **85** Norwegian FPGA designers
- Open or on-site course
  - Last courses:
    - Oslo, September 2012, Open course
    - Bergen, May 2012 (combined on-site, open)
  - Next course:
    - None planned. On-site on request. Open course when sufficient demand

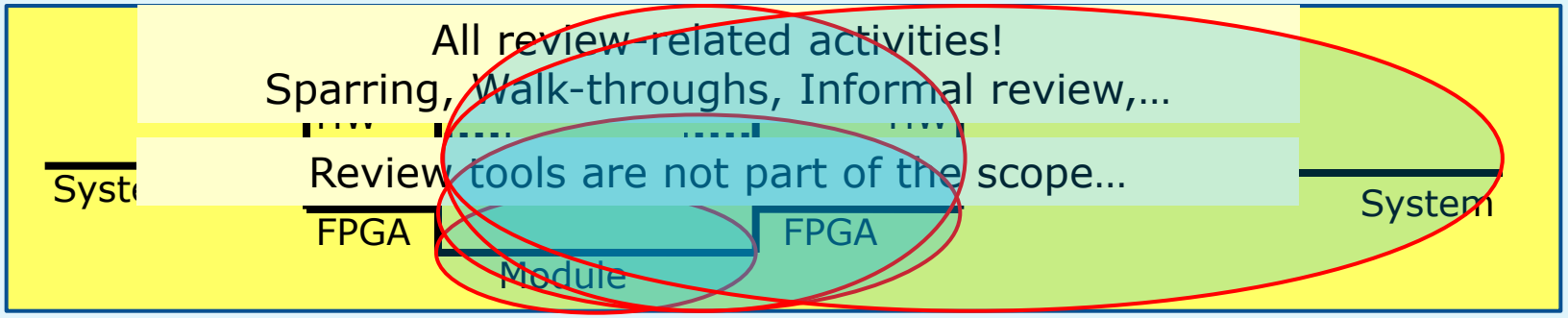
About this presentation

- First presented at FPGA-forum 2013
- Released as PDF
  - Animation not shown
  - Only one layer on each slide is shown.
    (Others layers may be hidden or covered)
- This presentation will be included in the next version of 'FPGA Development Best Practices'

- Are you interested in this presentation?
  - Please, get in touch, - and maybe we can present it on-site.
    (or as a webinar – depending on location)

Scope in this presentation

➔ An FPGA module - and the effects of review on that module
(after module requirement specification)



For the development of that module

FPGA

For FPGA integration

w

For parallell activities on the FPGA and the system

For system integration

For the final product

For future projects

State of the community - Reviews

Independent of ISO certification and nice QA processes…

- The vast majority of reviews are very inefficient, and a significant amount are close to useless

Typical findings in such reviews:
- Spelling mistakes
- Bad explanations – (but seldom the lack of such)
- Maybe some bad signal naming
- Maybe some strange coding style
- Lack of comments
- Register-functionality and description
- (If lucky – some issues with PLLs and I/O)

The review paradox

- Reviews are mandatory in most companies
  - because reviews are believed to be good for quality assurance
    - but lots of reviews are close to useless
      - and many developers know this....
- So why do we continue with reviews?
  - and Why do we not change the way we do reviews?
  - Do we continue just because we are "forced" to?
    At least from a marketing point of view….
  - Do we continue just because the QA dept doesn't really understand what is happening?
  - Do we continue as is – without change – because we don't believe in reviews anyway?
  - Do we continue as is – without change - because we don't want to spend more time on boring activities?

State of the community – Development Time & Product Quality

(No time to repeat the examples from our FPGA Best Practices course and previous presentations at FPGA-forum)

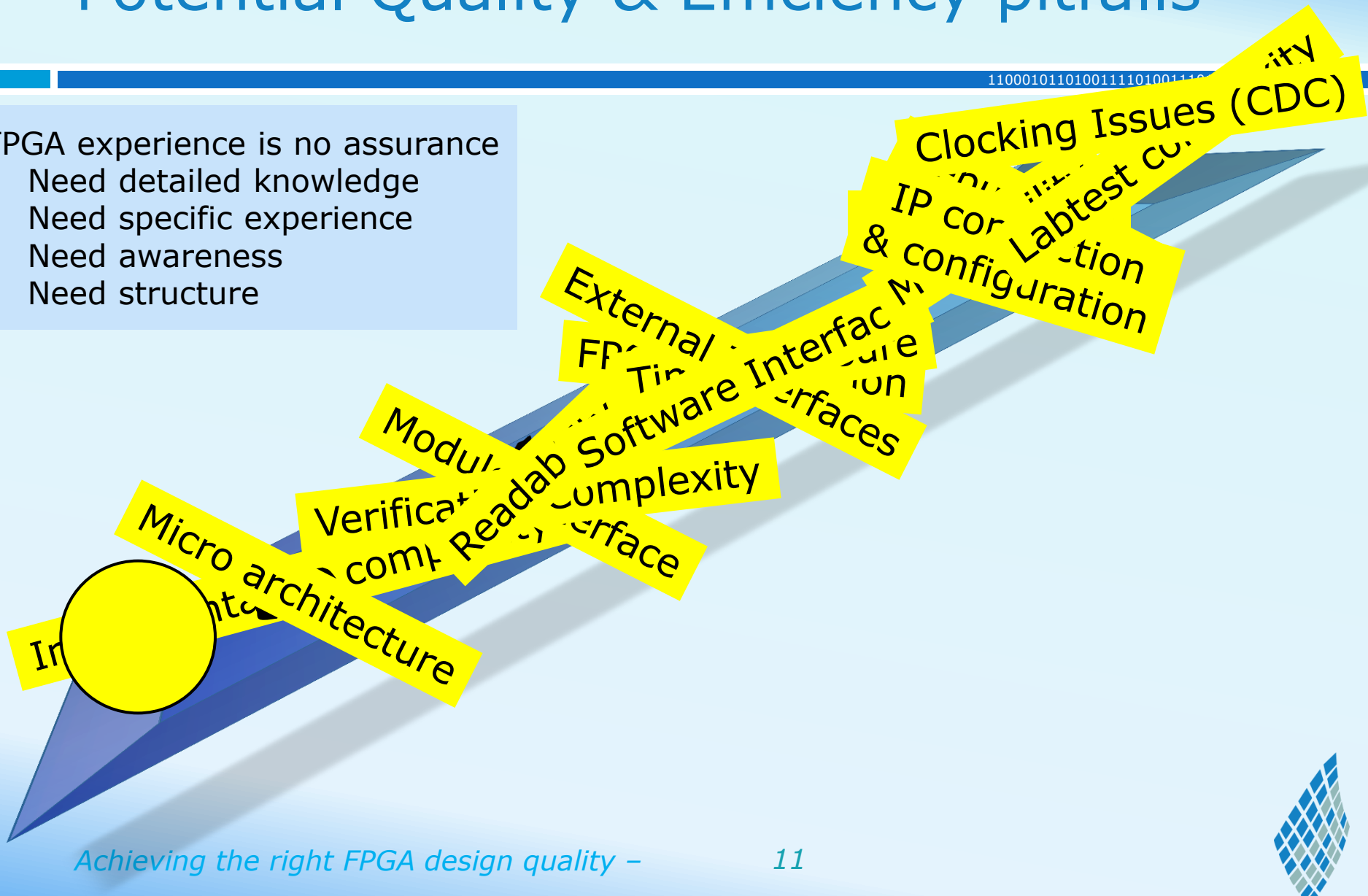
Q
- Product deficiencies
  - Not uncommon with serious bugs late in the project - or far worse – with serious bugs in the end product

q
- Product delays
  - Quite common with several months delay

q
- Time saving potentials
  - Most projects waste a lot of time due to insufficient quality in the design flow

Potential Quality & Efficiency pitfalls

FPGA experience is no assurance
- Need detailed knowledge
- Need specific experience
- Need awareness
- Need structure



How can we avoid the pitfalls?

- Tools and standard development stages:
  - Static timing analysis
  - Simulation
  - Testing
  - Lint, code coverage, functional coverage, formal verif, …

All very important

But in most cases - not at all sufficient

➔ **We do need help from other developers**

➔ We do need
Sparring, Walkthrough and/or Reviews

Pitfall prevention and mitigation

- Let us briefly consider:
  - The various pitfall areas
  - What can happen?
  - What are the consequences?
  - How can we avoid the pitfalls?
  - How can we reduce the impact and the consequences?

  - Do we really need to review each pitfall area?

Module Interfaces – Pure functional view

- Physical interfaces: Signals and Protocols
- Software interface: Registers and order of access

➔ No full verification or test until integration, but

- One of the issues properly checked at most reviews
  - and often easy to fix
  - Detected early in simulations and test

Sparring and Walkthrough:

➔ Improves interfaces and may detect problems

Will normally be detected. Reduces risk slightly. Low effort.

➔ Good ROI, but seldom critical for quality

Module Interfaces – Timing related view

- Module FPGA-external interfaces:

  Must document and Analyse. Timing diagram is mandatory.
  → Time consuming…
  Sparring and Walkthrough helps a lot. Review is critical.

- Software interface:

  Corner cases – normally easy to detect

➔ May waste a lot of time if detected in the lab/system-test

Sparring, Walkthrough and Review:
→ May be critical for quality – and may be lucky…

Micro architecture

- Important or Critical for
  - Timing closure ('10x vs 1.2x')
    - More a schedule & cost problem than a quality issue - but: schedule & cost problems inevitably lead to lower quality
  - Design structure and overview
    - A significant design quality parameter
  - Low coupling and high cohesion
    - Significantly decreases the probability of bugs

Sparring and Walkthrough:

A very high ROI if done early
Normally even saves time in current project

→ Maybe too late for project. Excellent for the future.

Readability & Modifiability

- Important for
  - Design structure and overview
  - Specification and design changes during the project
    - Less error prone (and much faster)
  - Reuse from a previous project (or in the next)

Sparring and Walkthrough:
→ A significant impact on readability/modifiability
Review:
→ Too late for project. Good feedback to designer and flow.

A good ROI if done early
Otherwise – hopefully helps on the next project

Clocking, CDC, Async

- Very often causes major problems at all levels
  - May waste a lot of time if detected in the lab/system-test
  - Very high risk being detected far too late
  - Follows all the rules of Murphy

Sparring and Walkthrough:
→ May be critical for quality
→ May save significant time in the current project
Review:
→ Very important. May result in important design changes.

→ From good to extremely high ROI
→ Very critical

Normal, basic functionality

- Critical for - - - Normal, basic functionality, - but…
  - The first to be checked - with basic module verification
  - - and if not – then with basic FPGA verification
  - - and if not – then with first basic lab-tests
- Micro architecture, readability, etc is critical
  - Checking the TB for the same = is not a review task

Normal basic functionality must be assumed verified through simulation and test

➔ **ROI : Not relevant in a normal design flow**
(other than what is already mentioned)

Corner cases

- Corner cases are dictated by the specification
- The implementation should not add more corner cases
- Corner cases are error prone and add time at all

Sparring and Walkthrough:
→ Important for quality and development time
Review:
→ Too late to avoid adding corner cases
→ Important in order to detect bugs and potential bugs

Corner cases should still be properly reviewed

→ A good to very good ROI
→ The earlier the better

How can we make reviews work?

- We must define
  - Our intention
  - Our review stages
    When to apply sparring, walkthroughs amd reviews
  - Our level of ambition
- We must allocate sufficient time
  - For preparations
  - For the actual review
  - For follow-up
- We must evaluate and improve

Review stages

- Allocate a sparring partner from the start
  - Always for inexperienced designers
  - Always for complex modules
  - Always for tightly interacting modules (FPGA or SW)
- Apply Walkthroughs (W) and Reviews (R) for:
  - Implementation Proposal (W)
  - Verification Proposal (Description) (W)
  - Additional walkthroughs when required
    - To be determined up front – and during development
  - Completed Design (R)

Level of ambition – main options

1. Pure structural review
   - General "look and feel" and documentation
   - Architecture, hierarchy and structure
   - Code (overview, readability, structure, error prone code)
   - Internal Clocking

Major step in workload

2. A closer view on selected modules
3. Module, IP and FPGA interfaces
4. Protocols and handling of such
5. System, solutions
6. Functional corner cases
7. External timing

Major step in workload

8. Functional normal mode – incl. Normal internal timing

Typical findings in a structural review

- Code simplifications
- Coding style and dangerous code
  - vector comparison, slv value comparison
  - multiple similar expressions (rather than variables)
  - Parallel FSMs and potential deadlocks
- Real example:
  Found all of the above in a 3-day review for customer
- Clock domains and CDC
- Resets and reset synchronization

ROI in comprehensive reviews

Example: Ethernet switch

- 3 weeks review (~9 months dev.) → Adding 8%
- Structural findings as on previous slide
- Also found several serious functional bugs; - some critical, others "just" significantly reducing the throughput.
- The FPGA was already "proven" in the lab

Real example.
- 8% overhead may seem a lot, but
  Could easily have been 100% overhead if issues detected later.
- Avoided costly interruption of operation in the field
- Avoided unhappy customer

Conclusions

- Given the right approach
- Given sufficient time
- Given the right priorities – at all levels

Proper sparring, walkthrough and/or reviews

may significantly

improve the product quality

- and at the same time significantly reduce development time

Bitvis review services

- Bitvis offers services on
  - Sparring partner
  - Walkthrough
  - Review
  - Mentoring
- We have major experience with
  - Pitfalls and debugging
  - Design structuring
  - Verification structuring
  - Clocking and CDC
  - Documentation at the right level

Achieving the right FPGA design quality –

Could Reviews save the day?

YES – they can!!!

bitvis

Your partner for Embedded SW and FPGA

www.bitvis.no