Date – 07/15/2011

Minutes of the IEEE-1149.1 Working Group Friday meeting

Attendees:
Adam Ley,
Brian Turmelle,
Carol Pyron,
Craig Stephan,
Roland Latvala,
Carl Barnhart,
CJ Clark,
Dave Dubberke,
Josh Ferry
Ken Parker
Wim Driessen
Ted Eaton

Meeting called to order at 8:35 am MST

New Draft: P1149 1 Draft 20110617.pdf (_clean.pdf)

Agenda/Overview:
REGISTER_ACCESS, REGISTER_FIELDS and REGISTER_ASSEMBLY (continued discussions)

Minutes:

Ted – Ken do you want to leave register access alone as is?
Ken – If we have register fields and register acesss we need a place to define lengths:

- What I'm most concerned about are mandatory registers. That we have a simplified way to define them. Flat description that doesn't include 'Use Packages'. Flat and standalone for registers that have well understood meanings.
- IP registers for MBist and Serdes may get complicated and require a dozen IP files. It gets complicated, but for EXTEST those are not relevant. I see multiple BSDLs as Carol pointed out.

Ted – If you run into a BSDL extension you don't understand you ignore it. Have different constructs for both to automatically ignore stuff you don't want to deal with. Keep the new optional Std instructions separate from the more complex MBIST/SERDES instructions.
Carol – Today you get hardened IP that includes the IO,IP and Boundary Scan segments. For that reason we don't want to use hierarchy on the boundary register. And so if we get a boundary register segment for Serdes we will have to flatten it. We could do the same for the IOCONFIG register.

Ted – You hit on a good point. If you can't make it all hierarchical, then what good does partial hierarchy?
Carl – Manually flattening is an issue, but easy for a tool.
Ted – If I have to flatten the Boundary register, I can flatten the other sections too.
Carol – FPGA vendors hopefully will embrace the init_data TDR
Josh – Sometimes I go through the FPGA process to get me something I can work with. I've never seen a nice way that is clean across all FPGA vendors. (3 tool sets). It's not easy.
Carol – One way if you want to separate internal TDRs from standard ones is to separate the Use statements. We could have a different keyword to separate types of includes.
Ted – Typically an FPGA BSDL file is a product of the tool used to generate it.
Josh – There is a generic BSDL that is not configured. If insitu and configured a designer can put it through a Tool and create a 'configured' BSDL. BIDI changed to Output only, etc. User code fields, and so on.
Ken – It sure would be nice to have FPGA vendors to the call.
Ted – If that tool auto-generates the BSDL, and we had a new standard instruction and the tool can dump into the register fields only the bits you care about, then hide the bits you don't care about.
Ted – To separate the new Std instructions from IP embedded instructions.
Carl – You are confusing package files from hierarchy.
Package file / PDL they can take ownership of the content.
Josh – Teradyne has dynamic reconfigurable FPGAs to mirror and match protocols.
Carl – Register fields for sparely defined bits works fine, no package file is needed.
Ted – Doesn't the PDL change?
Ted – Serdes Bist, Mbist. vs. Serdes macro (Carol). One related to Boundary register definition and the other does not. If you have to integrate into the BSR register, how much more difficult is the other internal test Mbist fields?
Carl – Ted you don't want package files?
Carol – Yes, Ted and Ken want to avoid side files.
Ted – Ken also wants to separate out the new Std instructions (3 new Inst, 3 new registers). from the custom user instructions ones.
Ke n – What I'm suggesting is to define how to document in simple way to describe what those things are. Other more complex instructions could be kept isolated. I'd like to look at basic way to describe the 3 new Std instructions.
Carl – Part of my argument with this, is that initialization register(s) is probably one of the most complex registers on the chip, and 3rd party hard IP. Today there is no standard way to describe one IP providers contribution to the TDR. In a form such as a 'package file' this is relatively well understood. We have to remember that the register mnemonics, register fields, register assembly are not documenting anything standard.
Ted – Carl if you cannot go all the way with hierarchy the usefulness is mute.
Carl – It goes both ways.
Ted – I thought we wanted to associate init-data register bits with package pins?
Carl – We are not done with that yet. We need to look at the boundary register.
Carl – Some BSR cells could be 'INTERNAL' instead of BIDIR.
Ted – Same in an TDR register if you don't care a bit needs to change to DON'T CARE.
Carl – You would have to define a new mnemonic.

CJ – If it is hard coded you don't have to define a mnemonic at all.
CJ – I think some folks were interested in hierarchical Boundary register, but we haven't gotten that far yet.
CJ – Carol where are we?
Carol – Trying to get our heads around this week's email.
Carl – Take register access and expand its definition then you wouldn't need to provide anything else. The register, the instruction and its length.
Carol – Chips are not simple now.
Ken – Not so useful. Test engineer needs to define side files. The test engineer can see his choices from the mnemonics without worrying about bit orders.
Ken – Maybe we mandate register fields at a minimum. I think we need to assist the test engineer to avoid problems for him. He will be scratching his head about which mnemonics to use, but don't make him worry about bit swizzling.
CJ – Force the length to be defined in register_fields
Ted – This doesn't solve it.
CJ – True you cannot enforce 'good practices' Folks can be sloppy.
Carol – True. Good Std examples will help this.
Carol – Today in-circuit tests do EXTEST, and other tests. Is that true?
Ken – Yes, today's EXTEST based stuff, and future 1687 Tests. We have guys to train on configuring IOs, and other advanced test engineering tests.
Carl – As editor I'd like to ask that we get a decision. Do we exclude the length of the optional TDRs from register access, thus requiring them to be defined in register fields?

Carol – Motion to: Require REGISTER_FIELDS for init-data, reset-select, and init-status registers. (ClampHold TMP controller has only 1bit register so nothing to define for it).

Vote Results:
Adam – Yes
Brian – Yes
Carl – abstain
Carol – Yes
Craig – Yes
Dave – Yes
Josh – Yes
Roland – abstain
Wim – OK
No objections from what I heard taking minutes. (Roland)

Next Carl reviewed his latest init_data EXAMPLE from sub-clause B.8.18 and REGISTER_ASSEMBLY example from B.8.19.

Carl has renamed the BSDL iREAD, iWrite, and related keywords from last week. He'll send out update in next draft.

Carl's review of register assembly example introduced the concept of init_head and init_tail to help assemble the init-data register more cleanly when misc bits are interspersed.
CJ – One thing that trickled down nicely. (although I'm not a fan of head and tail), is the benefit of counting bits relative to short segments as opposed to the whole init-data register
CJ – The init_head and init_tail help register assembly a little loose.
Carl – The original definition from dot6 a year ago was strongly typed to enforce checking.
CJ – I get it. The software engineer asks do you really want to exit?' or do you really want to open that file?' Annoying!
Carl – There will always be misc bits at top level to interleave with the arrays of reusable ip block segments. I think this is likely to be awkward.
CJ – I did send you an example of a package. The 'Use model' will be less valuable if there are bits interspersed. We lose the ARRAY capability.
Carol – Even though you have a 3rd party package file, they could have all been in one BSDL file.
Carl – Yes. A tool could change XYZ_package to XYZ_entity and use the flattened model. There is advantage to keeping the package file as-is to guarantee his definitions are being used.
Carol – True
Carl – Any questions or discussion before we wrap up?
Ken – Yes. You used a register assembly in this example, and register fields in prior example.
Carl – Yes.
Ken – You have init-status in register fields and init-data in register assembly
A parser has to keep track of this. Is this a complete BSDL or is something missing?
Carl – A 2 pass symantics check may be needed.
Carol – Overall question: Does this provide simplicity in the end?
Head, serdes, ddr, tail
CJ – pll, serdes, pins, other, this would be more descriptive.
Carol – CJ please show your other stuff next.
Carol – Team please review this is our first full up example of register assembly
CJ – Add BIDIR Serdes example with RX and TX power up/down fields

Meeting adjourned: 10:00am MST

Action Items:

-

Next Friday Meeting:

- Next week Friday July 22, 2011