Date – 03/25/2011

Minutes of the IEEE-1149.1 Working Group Friday meeting

Attendees:
Dave Dubberke,
Adam Ley,
Brian Turmelle,
Carol Pyron,
Roland Latvala,
Heiko Ehrenberg
CJ Clark
Carl Barnhart
Ken Parker
Wim Driessen

Meeting called to order at 8:30 am MST

Current Draft: Still P1149.1 Draft 20110309 clean.pdf

Agenda: (Informal)

1. Discussion of this weeks' email thread related to gated clock vs. free running clock implementation styles for TDRs and the Boundary Registers.
2. CJ continued review of his PDL work.

Minutes:

Gated Clocks vs. Free Running Clock Styles:

Carl – I took another look and the only place updatedr shows up is for the boundary scan registers. Now it affects init_data registers. In clause9 we want to develop a state based free running interface.

CJ – Providing those cells triggered the issue about instruction decodes

Carl – T\here is a note in clause 6 about gating the clocks to the update

Ken – We have rules and non-normative diagrams, but people worked this out. We have a lot of flexibility to get signals around the chip. At the same time this frustrates others who have to manage the implementations.

Carol – Style is something we have to be consistent on for 3rd party vendors

Carl – Figure 8-2 thru 8-13 all relate to the boundary reg

Carl – Fig 10.1 uses clockdr for the bypass register.

CJ/Carol – Figure 9-1 and 9-6 only mentions ‘clocks and controls’

Ken – Any summary/conclusion on all of this?

Carl - Fig 9.3 Capture and Update capability. Will show with clock-dr interface and inst decode. Also will show free running clock version. I will discuss both styles and in the rules make it a recommendation about a standard interface to a design specific interface. I’m going to develop 9.3.2 into much more than it is today. This will help IP users if it is standardized.

Ken, Is this the same as Wetzel (sp?) and Jason Doege are doing?

CJ – That is a standard TDR study group.

Adam – This is distinct from TAP reuse study group. Jason’s standard TDR interface has an approved PAR 1149.1.1. No meeting so far. This effort will overlap with him.

Carl – We should bring his work in then.

Adam – If 1149.1 is willing to take it up in totality it may be worth it.

Carl – We’d have to see how extensive his work is.

Adam – We met for several months. Fairly well developed at this point.

Carl – I’ll update fig 6.5 to include the boundary reg. Also include the free running states in fig 9.2 and 9.3. Already using this in new registers in this draft.,

CJ – Even if you draw another fig 6.5 I think the boundary cells need to be updated for the instruction decode.

Carl – Bringing the instruction decodes into the 30 figures to date.
Update-DR and Clock-DR are decoded near the TAP today. We need to continue this. It saves on gating around the chip. I want the signals used in the boundary register to be decoded near the TAP.

CJ/Carl – It is not clear what the decode is in these figures. Fig 6.5 change the name of the Update-DR state, and Update DR without the dash.

CJ – How do folks feel about this? What is the destinction, Update DR without the dash?

Ken – Somewhere there is a rule that says when a boundary reg changes state because of Update DR there is no restriction on the # of switching allowed. To avoid ground bounce the Update DR signal could be delayed around the chip. Does this stay the same? This is in section 11.3.1 permission G.

Carl – We are correcting the oversight of using Update DR without the Update Dr select register decode. Fig 6.5 will have one more signal into it.

Ken – Still have only 1 signal to boundary register but with more qualified state now.

CJ – Carl has to fix 1149.6 now too.

Carl – I'll write it up and present to the group.

CJ – You have fig 6.5 Gated clocks vs. State based figures. Readers will think we have 'Two heads'. How many signals are we talking about to the registers?

Adam – Boundary already distinguishes between shift and capture

Carl – Shift DR, Clock DR in gated clock dr scheme
Free running state based (data wrap back) – Shift DR, Clock DR, and TCK

CJ – Show diagram 9.3.

Carl – You have one more signal to distribute around the chip.

CJ - Let's look at the figures. I don't agree we should have two approaches.

Ken – This is 25% more signals. Give people the freedom to pick the method they want.

Carl – People have their own signals in addition to this.

Carl – You could bring these signals down from the TAP and not gate it individually for every cell.

Adam – My recorded opinion is to push all this discussion into a separate annex and leave the legacy stuff alone. I think we should change the boundary update-dr to something else.

Carl – Update-BSR

Adam – If we are updating them all then perhaps make them all uniform.

CJ – The editor said it wouldn't take too long to update the figures ☺

Carol – One more comment. The 2 muxes in the figure.should wrap back in the mux closest to the flop.

Carl – Easier to code as-is. CJ did it that way. I'll include examples of vhdl and verilog. Synthesis will do what it wants based on its primitives, etc. These figures are just to explain to the reader.

PDL Review (continued from last week)

Use model Fig C-1 – U3/U4 two instances of same IC.

Carl – I had asked if 1687 level0 and level1 work with a plain old 1149.1 chip?

CJ – It is difficult to answer at this stage. 1687 is not writing for 1149.1.
We are trying to get the commands standardized. The register names are different between ICL (instrument connectivity language) and BSDL.

Carl – I'm hesitant to discuss level1 and TCL if it requires separate access other than BSDL.

CJ – No, level1 can read BSDL. However 1687 constructs go beyond 1149.1
TCL could still run on level 0. Level 1 for 1687 returns varilables which level 0 does not.

CJ -
Over-shifting:
No over-shift for iWrite and iRead, but it could be done for iEndState. Will show more next time around.

C.8.2 Stored Scan Frame:
iRead sets the expected values for the registers. The tools are supposed to determine which instructions are needed to load the registers with iApply.

When you say iWrite to several registers, then say iApply, a massive number of scans occur to load all those registers. This doesn't seem possible for the boundary register. It seems you couldn't bundle the BSR with other registers. You would have to do two explicit iApply. One for BSR and another for the others.

Carl – You could apply them 1 by 1.

CJ – Which BSR instruction do we need to load? EXTEXT, SAMPLE, CLAMP, PRELOAD.

Carl – Use PRELOAD as a default. Else specify something else.

CJ – If it is not clear flag this as an error. Want to make sure tool vendors are ok with this.

Carol – For iApply –IR do you want this for any TDR that has multiple instructions, or restrict to the Boundary register?

CJ – Do we need this restriction?

Ken – Since BSR has set the presidence, then other TDRs could follow this sane approach.

Wim – Why not allow a default?

CJ – I said the opposite. I wanted to avoid debugging problems for the user.

CJ – Showed C.10.6 example.

Wim – Questioning flexibility of how you want to pass through the state machine. You can't only say you have IR or DR apply without knowing the current state you are in. You have to tell how you go through the state machine to inform the users/tool vendors.

CJ – We want to standardize loading the TDRs. The traversing the state machine is beyond this. That extra information can differentiate tools.

Wim – You can give guidance about how you are passing through the state machine.

CJ – OK I can add some information about what happens with iApply.

Wim – If you want to comply to the language you use this path through the state machine.

CJ – Anything that requires something beyond the update is beyond this standard.

Meeting adjourned: 10:00am MST

Action Items:

Next Friday Meeting:

- Next week Friday April 1, 2011