Date – 10/12/2010

Attendees: : CJ Clark, Bill Tuthill, Ken Parker, Craig Stephan, Adam Ley, Wim Driessen, Carl Barnhart, Brian Turmelle, Carol Pyron, Dave Dubberke, Heiko Ehrenberg, Francisco Russi

Missing with pre-excuse

Missing:, Roland Latvala, Adam Cron, , Lee Whetsel, Ted Eaton, Neil Jacobson, Bill Eklow,

Agenda:

1) Restore - 35 Mins

2) Final words on init-clamp

3) Beat up Editor for not having more done

Meeting Called to order at 11:00 am EST

Minutes:
Restore

Concept to cause an internal reset to occur to the IC

IC has an external reset pin and this instruction would cause this reset pin would be pulled low and then released.

Still can't guarantee that the chip isn't in mission mode, but it brings it close to the restore function rather than just loading bypass.

Carol – doesn't like the name restore. Implies that you are going back to a previous condition.
CJ – things RESET would be a nicer word. Such as JRESET
Carol – likes RESET better. Name it something that stands apart from other resets though.
KPP – agrees with changing the name. RESTORE could be misleading. Something short but meaning full. Such as SYSTEM_RESET
CJ – IC reset isn't bad.
KPP – original idea was that if device had system reset pin on it that this was a tap based way of issuing same signal. So whatever the definition of that reset is, then this would be a tap way of doing it.
CJ – envisioned something like a POR
KPP- is aligned with this idea.
Carol – this is an optional instruction
CJ, KPP – yes
CJ – this instruction causes any POR to reset.
Carol – language is complicated when there is no reset pin or multiple reset domains.

Point is to put the chip in to some state ready for the next state and not in some bizarre internal and pin state.
CJ – what is happening when we load instruction? How do I control the signal?
Carol – in and out of Test Run Idle
KPP – that is a good idea. And UPDATE IR the reset is asserted and then spend some time in RTL (1 - ? cycles) and then when you exit you have choice to go to SELECTDR unless you have asserted a POR signal and that has reset the TAP.

Does SystemRESET jtag function require you to go to TLR because the TAP is reset

Do we divorce the TAP resetting from the Chip Resetting?
CJ – it is already divorced. The only thing we have is a POR to reset at power on.
KPP – if we reset the POR with a JTAG instruction do we have to go back to TLR?
Carol – update IR doesn't do anything until you go to RTL. If you bypass RTL you can shift bypass.
KPP – with TAP based reset it would only affect the upper AND gate and not the lower AND gate. Looking at figure 6-8 – use of power up reset for system and test logic of 2001 Spec. (affects only system logic and not tap controller)

CJ – should make this as easy as possible. Makes sense to implement this in the chip as easy as possible as well as not break anything in the standard. Load instruction.. Instruction has to be active when it goes through Update IR.. When does it change? When instruction is decoded than it asserts reset. Cant guarantee that it is in any state. Either in RTL or it is in RESET. The only thing you can do after implementing this reset that is safe is a 5 clock TMS reset.
KPP – with instruction assert only to system logic (like POR) and don't reset tap. Don't want to get out of sync.
CJ – show in figure with 3 input AND gate to reset system logic only. What is the easiest mechanism to make this release?

Just clock with test clock? Like RUNBIST?
KPP – keep reset low for certain amount of time that can be equivalent to test clocks. After clocks are up than leave reset state.
CJ – should be a time to wait with RESET low and a time to wait with RESET High
KPP – Do we want a data register for this instructional
Carol – it should be bypass to keep it simple.
KPP –
CJ – when you have the TDR you have opportunity to use the TDR bit to make it high and low.
Carol – have done dozen of chips in the past with no reset flops in the chip other than TREST for dot 1 domain.

Reset the part by shifting zeros down the internal scan chain.

Zero set flops
Carl – that was the same methodology that IBM used for their Flush reset.
Carl – the way the normal resets work is that while the reset is low there is a certain number of flops that get cleared. When that signal is released is when the Reset state machine starts.

It has been a long time since I have seen a chip with a single reset. If you are going to support resetting, it is a mistake to support only a single POR type. Should be a TDR to support multiple resets. And reset pulse needs to be generated by run test Idle.

In to RTL to assert it low and allow enough time for resetting the flops to take place and then when you come out you would have to wait until state machine finishes.
CJ – Clear on need for time to be high and low. Like the TDR approach where you specify at least a single bit but could use more for multiple resets.
Carol – want to mimic pin behavior would be the clearest way to define it.

For the case where there isn't a TRST pin we need to take care of condition when TAP comes up in a random state.
Carl – needs to be a POR or TRST so it doesn't come up in a random state. Required by standard.

Carol—Can you put resets into PROC statements
CJ – Yes. You can make different PROCs for different actions. INIT SETUP, INIT-RUN, ICRESET
Carol – RUNLOOP will put state machine in RTL
CJ - only used to create wait time. Will provide TCK's
KPP—After RESET is set and TAP gets blow away. What happens to the chip? Does it stay in reset? How do you get RESET back high?
Carol – the chip will be lobotomized again.
Carl – if TDR bits are used as RESET without qualified with RTL than it would need to be cleared with TLR
KPP--- should be a trap to get chip into reset mode without getting it out.
CJ – simply load instruction again and scan register.
KPP – imagining a scenario that the boundary scan master doesn't work anymore. How do I get the board back. Short of cycling the power.

One way is to put header back on and sync up with TLR
CJ - - we will need to go off line to think about this?
KPP – if assertion was removed by going to TLR we could be safe.
CJ—wouldn't be in favor of that.. Thinks the answer is to target the instruction and rescan the chain to set the TDR and then release the reset.
Carol – could think of it as reset disable bits. Have to set them to do a reset. Make them inverted. TLR will clear bits.
Carol – does RUNBIST only take affect at RTL
CJ – every instruction today takes affect when you go through UPDATE-IR
Carl – Logic BIST requires RTL before it starts.
KPP – Pulse Train does the bulk of its work in RTL
Carl – benefit of going to RTL. Can simply load TDR with bits and use RTL to trigger state machine.
KPP – divorcing what you want to reset and the assertion of reset.
CJ – what is the benefit of having the RLT gate the RESET signal(s)?
CJ – 1149.1 tap and logic is available independent of system logic.
KPP – would get rid of the need for second scan if you 1's for reset and 0's for reset. When you leave RTL the reset is cleared. RLT is where the resets are executed.
CJ – let's pick one way to do the resets. Don't want multiple ways.

CJ – may be more work to trigger off of the RTL loop.
Carol – gate if off with the state of the state machine.
CJ – difference is that we are not defining the low and high times.
Carol – when you exit RTL it de-asserts.
Carol – de-asserts on leaving RTL. It is gated with that state.
Carol – the only time the RESET takes affect is in RTL. Gate RESET assertion with RTL state.

CJ – can live with it. But doesn't think it will solve the problems that Ken are envisioning. Doesn't see much difference in gating with RTL or having trigger after UPDATE IR.
Carl – RTL is one more safety mechanism to make sure that reset isn't asserted on bad data from TDR if you assert when leaving UPDATE IR.
Francisco – instruction Reset rather than ICRESET. ICRESET is used as a pin name.
CJ – open for different name.. not for Instruction Reset.
Francisco – have something called Local RESET.
Carol – would be a collision in BSDL if pin name and instruction are the same name.
Carl—that is why I liked instruction named RESTORE. Restoring reset state of the chip. And not likely to be a collision with a pin name.
Carol – there is a lot of JTAG done without doing a reset. No reset to restore to.
CJ – how about IC-RESET
KPP – you can't have "–" in instruction name.. could be" _ "
Carl- and only a single "_ "
KPP – no leading and trailing "_"
CJ – how about IC_RESET
Adam L – all identifiers need to be unique in BSDL.
CJ – keep it as IC_RESET. And take it offline.

Meeting adjourned: 12:13 EST.

Next Meeting: 10/19/2010 11:00 AM EST

NOTES:

ITC meetings
Public meeting at 10:30-11:30am on 11/2
Power Session at 6-8PM on 11/2
Meeting for 11/2 is canceled

Action Item by Carl to elaborate on concerns that he has with OO s on power pins and any rules that would need to be added to the standard to address those concerns.

Current Issues listed and who will champion that issue.
1 Observe only. – Ken and Carl

1. Directionality linkage. - CJ
2. Power Pins. - Heiko
3. Pairing power pins with functional I/O - CJ
4. Sample / Capture. – Carol (Freescale) & Roland
5. TRST included in PCB level diagram. – Adam L.
6. Slow to Fall/Rise signaling issue – CJ
7. "No Connect" – Ken and Francisco.
8. Device ID – Still needs work
9. Low-Voltage self observe shorts coverage problem – JJ & Intel
10. Init – Carol & Carl

Action Items:

- CJ will post 1149.1 draft on website with line numbers to make it easier to refer to items in discussion
- Comment #10 CJ will take action to look at possibilities to add to the 1149.1WG website a document which shows which standards are based on 1149.1
- Comment #8 CJ will make changes to draft for observe only
- Comment #7 CJ will get in touch with Doug to get input regarding Comments
- Comment #5 CJ will Add a figure and little text to address TRST use with interconnection of components
- Comment #4 Adam L to add comment about TRST. Update figure 6.8
- Comment #3 Adam L will update language for any proposed change for this section.