Useful references and resources from this webinar

PolarFire FPGA
www.microchip.com/en-us/products/fpgas-and-plds/fpgas/polarfire-fpgas

PolarFire SoC
www.microchip.com/en-us/products/fpgas-and-plds/system-on-chip-fpgas

Get Ready for PolarFire 2FPGAs...

The PolarFire 2 FPGA family will go even further up the performance and power efficiency curve and add new RISC-V-based high-performance compute elements, adding 15× more compute capability to our roadmap: https://page.microchip.com/register-for-polarfire-product-updates.html

Related Doulos Training

Doulos Comprehensive VHDL Training:
www.doulos.com/cvhdlol

Doulos Comprehensive SystemVerilog Training:
www.doulos.com/csvol

Doulos Essential Formal Verification Training
www.doulos.com/essentialformalol

Doulos Training
www.doulos.com/training

Full Doulos Course Schedule
www.doulos.com/training/course-calendar

Q&A Log for Session 1 **Q&A log for Session 2 »**

Audience Question:
Q: Good morning. Could you please advise how to consider mesochrone clocks (same frequence but not the same waveform)?
A: When two clocks are derived from the same primary clock then they have a fixed phase relationship, are mesochronous, and require no synchronisation circuitry, just appropriate clock constraints.

Audience Question:
Q: How does the capacitance depend on the fan-out?
A: Fan-out involves more routing nets and capacitance is proportional to total net lengths.

Audience Question:
Q: But why bother calculating the MTBF if the final state is not predictable?
A: It is statistical. Voltage, temperature and process variations affect behaviour. Setup and hold requirements are determined by silicon vendors based on worst case conditions. Marginal infringements will not always trigger metastability but meeting all setup/hold requirements result in MTBF measured in 100s years.

Audience Question:
Q: Which situation we would be needing the optional guard flip flops, is there any mathematical computation for number of optional guard flip flops required?
A: Guard FFs are necessary when a single bit signal has come from an unrelated clock domain and will sometimes be sampled within the setup/hold timing window. One FF for level detection and two FFs for edge detection.

Audience Question:
Q: How many resync flop is necessary with respect to clocks ratio
A: The minimum number of guard flip-flops is always one. Two guard flip-flops is the minimum requirement if you need to detect an edge in the Rx domain.

Audience Question:
Q: How would the xor help? If cdc_sig is 0, N2 is 0 and N3 is 0, N2 XOR N3 would give 1
Is that not the incorrect operation?
A: On slide 9, you could connect an XOR gate to N2 and N3. In the Rx domain you would then get a pulse of one Rx clock cycle width indicating that an edge has been detected.

Audience Question:
Q: My understanding was that the 2 flip-flop technique is valid when we cross from a clock to a faster clock, but not the other way around. A different technique should be used in that case. Is that correct?
A: Relative clock speeds are not the issue. It is how often the data changes compared to the receiving clock domain.

Audience Question:
Q: Can you please explain what is the use of EXOR gate here and why not other logic gates?
A: Assuming the question relates to the XOR gate mentioned in scenario 1: CDC for single nets. The XOR gate will indicate when the synchronised input changes value, 1 to 0 or 0 to 1. XOR because it is necessary to recognise a difference between two successive samples, the XOR function.

Audience Question:
Q: CDC Vector, would one hot encoding be the solution?
A: One hot encoding is one possible solution. Gray coding is suitable in some scenarios too, for example shaft encoding.

Audience Question:
Q: How Asynchronous FIFO solve the problem of fast to slow domain even it has inside a double flip flop synchronizer? Why not just use a double flip flop synchronizer
A: Asynchronous FIFO is helpful when transferring data words where bit coherence is important.

Audience Question:
Q: If we have a synchronised slow->fast single bit, can we use that as a qualifier to mux multibit data slow -> fast?
A: Yes.

Audience Question:
Q: Can we use virtual clocks or generated clocks for rectification of cdc?
A: The generated clock is as good as any other clock to synchronise data. The virtual clock is not relevant as it does not touch the device being constrained.

Audience Question:
Q: To add on to the asynchronous fifo answer, an async fifo is also more resource efficient than a double ff synchroniser. simpler to implement and fixes the issue
A: Double FF is perfect for synchronising a single bit (eg: control flag or enable), asynchronous FIFO is perfect for synchronising multiple coherent bits from a data bus passing between asynchronous clock domains.

Audience Question:
Q: What is a gray-coded value?
A: A value that uses the properties of Gray codes. See the wiki page: https://en.wikipedia.org/wiki/Gray_code

Audience Question:
Q: For adding in a gray counter, would it make sense to make that only a few bits wide, then upon conversion expand to wider size? What would be the optimum grey code size then (to achieve stability, yet minimise bit count)
A: Gray codes on data buses that are counting, up or down, between adjacent values have only one bit change at a time, regardless of length. If the counting rate is fast but still less than half the clock speed, then gray code works. If counting rate is much slower, then synchronising a single data valid bit to sample the whole data bus at a safe time will have fewer gates. Depends on the scenario.

Audience Question:
Q: Is it mandatory to use the gray code even when using the sync+guard logic to avoid metastability? won't a binary counter work including the sync+guard logic?
A: The gray code is necessary because a binary counter has adjacent values where more than one bit changes. eg: 0b0111 and 0b1000.

Audience Question:
Q: A bin-to-gray encoder is combinational circuit. Does it need a flip-flop in the source clock domain?
A: Yes. That would give the best solution.

Audience Question:
Q: What are the different constraints that need to be included in the constraint file for CDC analysis?
A: Depends on the synthesis tool. See your vendor's timing constraints user guide. There are different approaches to replacing the inappropriate single cycle setup/hold requirement. One reliable approach is a set_max_delay constraint on the data path and set_property to identify synchronisation FFs as ASYNC.

Audience Question:
Q: If the input to the bin-to-gray converter (pure combinatorial logic) changes from "0111" to "1000", then surely there are still a sequence of possible changes such that more than one output from the converter can change? Meaning in this particular example a more than one metastable bit might be sampled by the destination register? Vivado seems to think there is a CDC-10 error with the design being proposed. (I did some homework before joining.)

Vivado says your gray code solution as presented is not optimal, not completely right. It's subtle I grant you, but I can see why it points out this issue. I propose that the solution presented needs the outputs from the gray converters registered in both directions to avoid CDC-10 errors. i.e. not LUTs on inputs to the synchroniser in the destination clock domain. This will then avoid sampling of any intermediate metastable events.
A: I agree with you. The diagram is simplified and the bin2gray and gray2bin blocks must include FF outputs.
Please Note: The recording of the video presentation has been edited to address this – thank you.

Audience Question:
Q: Should we add False path for 2 different clocks generated by same clock source domain?
A: No. These clocks are related and need no constraints for safe timing.

Audience Question:
Q: How is the handshake synchronizer technique exercised?
A: One example of using the handshake synchroniser is in asynchronous FIFOs where empty and full information must be transferred between read/write clock domains.

Audience Question:
Q: To my knowledge using false paths is bad practice and max delay paths should be used instead. When should I use which?
A: False paths are used where data is never going to be transferred along the path.

Audience Question:
Q: What about the routing delay between the 2 clock domains, maybe we have a huge routing delay and we simply ignore it by setting false path constraints?
A: Good point. Prefer set_max_delay (-data_path_only) constraint to limit the distance between synchroniser FF placement.

Audience Question:
Q: How can we get detail information on SDC?
A: See your tool vendor's user guide, and/or join a Doulos technology course. Doulos has also run a webinar on this in the past. Contact webinars@doulos.com if you'd like to view it.

Audience Question:
Q: What if the gray counter increments faster than the receiving clock? Will it still be a good solution?
A: No. Only suitable when receiving clock is much faster.

Audience Question:
Q: After adding two flip-flops (2FF) to avoid clock domain crossing (CDC) issues between two clock domains, we can add a false path constraint between the two clocks to prevent timing analysis between them.
A: Yes that works by eliminating the inappropriate setup and hold timing checks. Set max delay is another solution.

Audience Question:
Q: When is it appropriate to use set_false_path vs set_max_delay?
A: When the path exists but is not relevant it is a false path. For example, the path from a FF through an IO port and back into a FF. The first FF is relevant only on the read cycle, the second FF only relevant on the write cycle.

Audience Question:
Q: Why we call it false path?
A: The functionality never depends on data transferring along the false path.

Audience Question:
Q: Even in case of a binary counter that has adjacent values where more than one bit changes (0b0111 and 0b1000), why does that matter when an async FIFO ensures that any errors are accounted for? So are gray codes not necessary when using async FIFOs?
A: Yes, gray codes are unnecessary when using an asynchronous FIFO.

Audience Question:
Q: After adding two flip-flops (2FF) to avoid clock domain crossing (CDC) issues between two clock domains, can we add a false path constraint between the two clocks to prevent timing analysis between them?
A: Yes, but the max_delay constraint is better to lead placement tools to place the FFs close to each other. Allows maximum time for metastability recovery.

Audience Question:
Q: What is Polarfire tool used for in the industry?
A: Please check out the use cases on the Microchip website. You can also reach out to Microchip using the exit survey.

Audience Question:
Q: Is it correct that the sync+guard approach removes metastability but does not guarantee that all the bits are correct, whereas an async FIFO also guarantees all bits are correct?
A: Yes. You are correct.

Audience Question:
Q: Is the set_false_path the only way to constraint the CDC?
A: No. There are several ways to modify the default single clock cycle setup/hold timing checks. Setting a false path is sufficient in some cases.

Audience Question:
Q: CDC mitigation is better using Asynch FIFO, Gray codes or 2-3 step Synchronizers?
A: Depends on whether you have a data bus (FIFO), counting data (Gray code) or single data bit level (2 FFs), and single data bit edge (3FFs).

Audience Question:
Q: If a signal travels between 2 clock domains, after adding two flip-flops (2FF) to avoid clock domain crossing (CDC) issues between two clock domains, can we add a false path constraint between the two clocks to prevent timing analysis between them? Is it a right way?
A: Yes, but a max delay would be better to force placement to put the two FFs close together, maximises metastability recovery time.

Audience Question:
Q: How to set_input_delay for DDR signals?
A: These questions are beyond the scope of synchronisation circuitry. They are covered in other webinars and on Doulos technology courses.

Audience Question:
Q: What should be considered when we have an async reset resetting entities on different clock domains
A: The reset should be synchronised into every clock domain it should act upon.

Audience Question:
Q: "The reset should be synchronised into every clock domain it should act upon." - meaning the reset signal should be copied/registered in each entity before it is used?
A: Every clock domain yes. Once for each unrelated clock domain, not necessarily every entity.

Audience Question:
Q: What makes an async FIFO better than a sync+guard approach for a data bus?
A: The asynchronous FIFO can give much faster data rates.

Audience Question:
Q: Is it possible to verify the clock domain crossing in gate level simulation and how?
A: It is enough to completely and properly constrain a design that uses standard synchronising topologies then run static timing analysis to achieve timing closure. You can still use gate level simulation as a timing sanity check but there are other reasons for gate level simulations.

Audience Question:
Q: Does async reset also need guard flip flops?
A: Yes. Synchronise the reset before fan-out.

Audience Question:
Q: The double FF can reduce the probability of a meta-stable state being carried forward. However, can it ensure that the bit that is being carried forward is correct (as the bit from meta-stable state can go back to its original state or to the new state)?
A: The data rate must be lower than the synchronising clock, so it does not matter if the new value is synchronised on the 1st or 2nd clock edge.

Audience Question:
Q: A general question is debated when writing timing constraints for CDC's. Some use set_false_poath/clock_groups to tell the tool to not evaluate CDC paths but others strongly advise not to. Instead evaluating each cdc path and set the appropriate cdc measure, then set a set_max_delay constraint. I see this is practiced very different from developer to developer. What is your standing on this and recommendations/best-practice to how to ensure a proper constrained CDC design?
A: Set max delay includes the direction to place CDC FFs close together. False path does not. The shortest net between synchronisation FFs leaves more time for metastable recovery.

Audience Question:
Q: Would it not be more efficient to define clocks as asynchronous to each other instead if defining set_false_paths? STA won't try to close timing between asynchronous clocks.
A: That is another approach that would work and is used by some designers. There's more than one way to do it.

Audience Question:
Q: If the clock supplied to the FPGA is coming from the PLL of the PS in an MPSoC, For instance, Ref clock of PS is 33.33MHz and the clock supplied to PL is by the PS is 80 MHz through PLLs. Do we need to consider CDC ?
A: No. These are related clocks and STA will apply the appropriate setup and hold constraints.

Session 2 Q&A on following page

Session 2 Q&A log below Q&A log for Session 1 »

Audience Question:
Q: Does metastability always go away on the *next* clock cycle *if* the inputs are stable by then?
A: Not always: there is a slide coming up. A rule of thumb is "up to five times the nominal switching time if the set-up and hold times are observed".

Audience Question:
Q: Does metastability always go away on the *next* clock cycle *if* the inputs are stable by then?
A: So it does depend on the clock rate on the receiving side.

Audience Question:
Q: Isn't the metastability recovery related to the speed of the logic?
A: Not quite. Metastability can occur if the flip-flop is running outside of the safe operating conditions. The clock rate at the transmitting side or the clock rate at the receiving side is not really important. Important is whether the flip-flop switches inside or outside the safe operating conditions as seen from the receiver side.

Audience Question:
Q: So your 5 times rule of thumb means if your flop clock to Q is 2ns then you can safely operate a 2 stage synchronizer at 100Mhz or less?
A: The rule of thumb is intended to guess the maximum settling time. If your 100 MHz clock hits the point of time at which the flip flop switches then a rule-of-thumb says the settling time can be up to 10ns.

Audience Question:
Q: Why isn't the output of the first guard flip flop necessarily metastable if the input to it is metastable?
A: Metastability is a statistical observation. Not every sample outside of the safe windows will become meta-stable but it can happen. We don't know whether. If the guard flip-flop sees a stable (but possibly incorrect) sitate on N1. It will not become metastable itself.

Audience Question:
Q: What keeps the guard from latching and propagating the metastable state (or simply the wrong state)? Is it only the ratio between the clocks? With rcv.clk much faster than txm.clk (on this slide).
A: It depends what you mean by wrong state. What cannot be resolved, even with the sync/guard flip-flops, is whether the final state at the end of the guard chain is the next state or the previous state but it will be a stable state.

Audience Question:
Q: Is there an issue with the release of resets occurring at different times in different clock domains? (Since the clocks are asynchronous)
A: Yes, that was mentioned I think. With asynchronous resets the method should be reset asynchronously but release synchronously. Please look at scenario 1.

Audience Question:
Q: The vector synchronization method mentioned may not be coherent for the first rcv.clk. Correct ?
A: To which slide are you referring? Is it scenario 2 or scenario 3
A: For scenario 2, the use case is that the pattern on the Rx side does not always have to match the pattern from the Tx side. Important is that both side eventually reach the same stable pattern. Again, a possible use case is multiple buttons all sampled in the same vector. Possibly you do not to see every pattern as sent by the Tx. It is OK if the Rx pattern appears to settle in multiple steps.

Audience Question:
Q: Scenario 2
A: So yes, your observatino is correct. The Tx to Rx pattern is not necessarily coherent. The use case only refers to situations where this is your design requirement.

Audience Question:
Q: Does this imply that the sync clock must be 2 times (when there's sync and guard FFs) or 3 times (when there's sync and two guard FFs) faster than the input clock domain we are synchronizing with?
A: Not quite. What is important is the relationship between the Rx clock rate and the worst case settling time of the Rx flip-flops. If your Rx clock rate is very fast, then you need more guard flip flops because here the left-most guard flip-flops may indeed also become metastable. Metastability is a probability. As shown in the formula, the probability of metastability decreases exponentially with each guard flip-flop

Audience Question:
Q: How do we know how many guard FFs we need? If FF clk to Q is 2ns and I have a 200Mhz clock how many stages do I need in my synchronizer?
A: Yes, that's a good question. It depends whether you are building a satellite out in space for forty years or just a tamagotchi for consumers. Theoretically, a meta stable may never settle. But that is just theoretical. Due to fluctuations in temperature or on the voltage rails a flip-flop will never lock in the metastable state forever. I suppose the best approach is to divide the worst case switching time (in the presence of metastability) by the clock rate. For a critical product always consider adding one or maybe two additional guard flip-flops.

Audience Question:
Q: Is simply adding sync and guard flip-flops enough to ensure crossing a signal from a faster clock domain to a slower clock domain gets sampled? Or would there be some sort of acknowledgement from the receiving clock domain before the transmitting clock domain transmits its next value?
A: Well yes, we do have slide showing the use case you just described (Scenario 3?) The problem here is the latency introduced. Otherwise, consider using an asynchronous FIFO. Also, if safety is an issue, also consider adding parity or checksums to the data you are passing from Tx to Rx domains.

Audience Question:
Q: Wouldn't a clock-to-clock false path be dangerous ... as it would be agnostic to whether or not the user included appropriate clock-crossing circuitry? It would seem that the appropriate constraint would always be on "approved" clock crossing circuitry (as opposed to globally on clock domain relationships)!
A: With the clock-to-clock false path we are saying that we cannot determine (and therefore cannot specify) the worst case timing delay on these nets, since the clocks are sweeping at an unknown rate against each other. In the case you describe you should also apply Linting tool checks to verify that synchronisation logic on CDC boundaries is indeed in place.

Audience Question:
Q: Can you please share some probability/statistics for CSC error with and without parity for the various techniques?
A: This is a tough one. The data is technology specific and very few manufacturers publish all necessary data to resolve the formula we showed on one of the early slides. An approach, if you have the resources is to setup a test on your lab with two clocks sweeping against each other and a detectino logic to count the errors e.g. on a counter passed across the CDC boundary. The problem here is that your result is only valid for one piece of silicon. If the manufacturer changes their process then it is back to the drawing board. Again, if safety is critical you need to go beyond what we have presented and use implementations such as redundancy, check-sums, parity checks etc.

Audience Question:
Q: Does this imply that the sync clock must be 2 times (when there's sync and guard FFs) or 3 times (when there's sync and two guard FFs) faster than the input clock domain we are synchronizing with?
A: The idea behind the scenarios is that they are independent of the clock rates. I suppose the only case where this is not really true is scenario 2. Here, if the Tx rate is faster than 2x Rx clock rate then yes you can miss patterns. If that is a problem please look at scenarios 3 or 4.
A: The solutions presented here are intended to be independent of the clock rates, except maybe scenario 1 and 2. Here if Tx is 2x or more faster than the Rx clock patterns across the CDC interface can get lost. If that is a problem, please look at scenario 3 or 4. Of course if the FIFO in scenario 4 runs full we still have a problem. It is a design decision how you mitigate this problem. Allow patterns to be dropped? Pass in wider vectors across the CDC domain? i.e. pass two or more samples from Tx across the CDC domain at any Tx edge.

Audience Question:
Q: I got in very late into the presentation. I see early questions about metastability. Was there a discussion about synchronizing external signals brought into an FPGA? My rule of thumb was always to use a two-series-flip-flop input circuit (both using system CLK) any incoming signals, and the metastability would dissipate in the period defined by the maximum rated operating frequency of the FPGA. if the rated maximum operating frequency of the FPGA was 50Mhz, then the metastability was expected to have dissipated within a maximum of 20nsec. Is this still a valid rule of thumb?
A: Yes, that generally works. But it does depend on your clock rates or to be more accurate the rate at which your input data are switching
Q: (re answer above): the "clock rates" you referenced would, by definition, have to be less than or equal to the datasheet's absolute maximum clock rate of the device?
A: Absolutely. A clock rate faster than specified by your ratget technology can also be a cause of metastability. If the next event occurs before the previous event has settled.

Audience Question:
Q: One of my personal CDC rules is that you never put any combinatoric logic in the path between the source domain FF and the destination domain FF - this path must always be direct from one FF to the other. Combinatoric logic can glitchy. For the bin-to-gray encoder, would it not be better to include another register after the bin-to-gray, before launching it to the other domain? Or is the bin-to-gray encoding glitch-free?
A: Definitely. We mention that somewhere in one of the early slides. The synthesis objective should also be to force synthesis to place the flip-flops as close together as possible. e.g. specify a very short propagation delay between sync and guard and guard and guard flip-flops

Audience Question:
Q: An important point was missed in the discussion of constraints. Minimizing the Tpd between the metastable flop and the second flop in order to decrease the probability of a metastable event making it through the second flop. If the is not constrained, it it can easily make it through the second flop with a Tpd approaching the clock period
A: Yes. For an FPGA the optimum solution is to have sync and guard flip-flops in the same LUT. For an asic flow to nudge synthesis to keep the flip-flops as close together as possible e.g. by specifying very short max delays between syn and gard and guard and guard flip flops. We did mention somewhere at the beginning that the capacitive load is important.

Audience Question:
Q: I missed the statement during the static timing analysis where the presenter noted that you can determine if a CCD path needs to be addressed. How is this determined? How would one know if they missed designing logic correctly for a CDC path using the Static Timing Analysis tool.?
A: An STA (static timing analysis) tool is probably not good enough. Using a Linting tool to check all CDC boundaries should also be part of your design flow.

Audience Question:
Q: OK but is there ANY indication in the Static Timing Analysis that may prompt me to take another look at my design?
A: Probably not. You may even get different STA results if you resynthesise your design. e.g. if the synthesis tool starts with a different seed. Even if you make a small change in your design this can also happen. STA is really only reliable if your design is synchronous. CDC boundaries are not synchronous.

Audience Question:
Q: Do we still need to run Libero in windows 10?
A: I expect you do. Libero is the Microsemi/Microchip synthesis tool, or the framework around this. Libero is the IDE

Audience Question:
Q: What's maximum PolarFire SoC bandwidth of the Coherent Switch from FPGA logic to the DDR4 Controller?
A: That would be a question for Microsemi/Microchip. I expect it is specified in the description of their IP core. If not, please get back to us.

Audience Question:
Q: What's maximum PolarFire SoC bandwidth of the Coherent Switch from FPGA logic to the DDR4 Controller?
A: You can reach out to them via the exit survey, but possibly quicker to contact them directly via their website: https://www.microchip.com/en-us/support