**Xilinx WP272 - Get Smart About Reset: Think Local, Not Global**

WP272 mentions an essential point at FPGA (Field Programmable Gate Array); ***RESET***. It says that resetting all the registers in the FPGA design could be unnecessary and even gets cost more like timing issue. When an asynchronous reset is applied, some FFs (Flip-Flops) could be get in metastable situation because of setup time relative to the clock edge and it gets even worse when the clock frequency is increased. Also resetting all registers (global reset) create high fan-out network in the FPGA and this effects also timing issue badly. This timing situation rate are rarely (like a design is not work first time but may works after powered-on the FPGA or retrying the reset) but not guarantee that it works all the time. If a design is safety critical, it is matter and need to be evaluated carefully.

Some circuits do not care about reset releasing like a pipelined or feed forward designs because the output eventually get the correct data after a few cycles, so these are do not need reset like tap register in the FIR (Finite Impulse Response) filter. However, if a design has feedback structure than releasing the reset could create an unwanted situation because some FFs could be released before the other FFs and gets wrong data like in the IIR (Infinite Impulse Response) filter or one-hot state machine encoding.

Xilinx FPGAs has a built-in feature that can be initialize registers with the wanted value, and even RAM (Random Access Memory) data in the power-on configuration of FPGA. With this feature, reset could be unnecessary.

If a reset is desired, when the asynchronous reset is used it should be used with synchronizer reset circuit that get assert reset essential FFs asynchronously, but de-assert reset synchronously, so with it reset is localized.

Also using a reset brings cost. When the reset is used, routing gets more complicated, more logic resources are used, prevents some dedicated logics like SRL16E.

To sum up, reset situation are critical and should be evaluated carefully. Using reset (especially global reset) could be generate timing issues, uses more logic resources and increase place and routing time. So, Xilinx FPGAs initializing FFs and RAMs feature could be used for reset purpose on the power-on of FPGA. If reset is needed (or insist of some reason), only the necessary signals (like FSM (Finite State Machine) registers, output ports, control signals) should be reset not the intermediate signals (like FIR taps or a counter register that its value is assigned before used). Also, when the asynchronous reset is used, it should be localized with the local clock. Always remember that it if you are using reset, think always the effects/costs and ask the question “Does this bit need to be reset”?

**WP275 - Get your Priorities Right – Make your Design Up to 50% Smaller**

WP275 mentions on fundamental building blocks in FPGA (Field Programmable Gate Array) like LUTs (Look Up Tables) and FFs (Flip-Flops) and their correct usage. Being aware of the exist hardware/logic resources, knowing its features and writing HDL (Hardware Description Language) code according to this knowledge could be optimized the design.

Different type of FFs are exist on the Xilinx FPGAs like one has synchronous reset and one is asynchronous reset etc. FFs features have priority; reset > set > clock enable. If the code is written according to that priority, Vivado synthesis tool could be infer correct hardware with the correct structure. If a code is written that is not fit that priority (like using set assignment before the reset assignment), synthesis tool infers the structure that is mentioned in the HDL code but in a way that using more resource like LUT, not using the exist feature. So, realizing the exist logic resources of FPGA and writing code correctly in a way that synthesis tool infers the exist structure directly are important and HDL code writing mechanism effects the logic resource uses so design can be a smaller.

In addition, using a mixed reset style (synchronous and asynchronous) on the same FF is not recommended.

To sum up, always think the code in a way that synthesis tool what can be infer with it according to FPGA resources and follow the guides of FPGA vendors.