RTL DEVELOPMENT AND TESTING FOR MEDICAL DEVICES

Blue Pearl Software

Medical Device RTL Development and Testing

INTRODUCTION

The medical device industry is one of the biggest industries in healthcare, driven by new and innovative technologies. This unprecedented growth in state-of-the-art medical devices has driven advancements in the healthcare industry as well as attracted many new suppliers. The industry is known for its diversity, not to mention its daily innovations.

However, medical device regulation is complex, in part because of the wide variety of items that are categorized as medical devices. A "device" may be a simple tool used during medical examinations, such as tongue depressors and thermometers, or high-tech life-saving devices that are implanted in the patient, like pacemakers and coronary stents.

In this market, ASICs and FPGAs are critical components used in many leading applications such as:

- Diagnostic imaging including X-rays, ultrasound, computerized tomography (CT) scan, magnetic resonance imaging (MRI), and nuclear positron emission tomography (PET)
- Electro-medical including patient monitoring, life support, and anesthesia equipment
- Cardiac Rhythm Management including Pacing systems, implantable cardiac defibrillators (ICDs), and automatic external defibrillators (AEDs)
- Life Science & Hospital Equipment including lab instrumentation, radiation equipment, and various hospital equipment

While the medical device industry has seen dramatic growth, it has also seen an increase in compliance and diverse regulatory requirements both foreign and domestic. The good news for designers is that with IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems, all integrated circuits used are generally considered as black-boxes for regulatory purposes, including ASICs and FPGAs. However, in systems where the safety function is highly dependent upon the ASIC/FPGA functionality, the detailed design of that device may become a central part of the safety assessment.

WHAT STANDARD MAY APPLY TO YOUR ASIC OR FPGA

When choosing to use an ASIC or FPGA, it is important to determine which regulatory requirement is relevant. Below are what experts agree are good rules of thumb to follow:

1) If the ASIC or FPGA is purely based on RTL, then you may only need to apply IEC 60601-1 - Medical electrical equipment - Part 1: General requirements for basic safety and essential performance.
2) If the ASIC or FPGA has a processor that runs a small amount of C code that did not require software architectural design, both the hardware and software may be considered as a black box and only IEC 60601-1 is required.
3) If the ASIC or FPGA is a System-On-Chip, then apply IEC 60601-1 if the program portion is tiny enough, otherwise apply IEC 62304.

However, it is up to the software and hardware designers to decide if IEC 62304 is applicable or not. It really is subject to the complexity of software.

DESIGNING RTL FOR SAFTEY

When designing an ASIC or FPGA, throughout the coding phase (RTL development) of the design it is important to follow good coding practices. This includes writing RTL code (typically VHDL or Verilog) that is easy to understand, verify and maintain. As with conventional software that runs on a processor, it is important to apply a good coding standards or a set of rules for how to write good quality code, as well as to document the code through comprehensive formatting, as well as meaningful and unambiguous signal, variable, function and module naming. Clear and intuitive code is preferred by auditors over incomprehensible compact coding. The key is to keep the RTL simple and clean!

This is where a good RTL Linting tool can play a major role in the development process. Tools such as Blue Pearl Software's Visual Verification Suite, with advanced RTL Linting can examine the coding style of the RTL and report back on issues such as hard coded constraints, upper and lower case names and constants, clock prefixes, line length, unused parameters and variables, unconnected inputs and outputs, clock gating, reset conditions and more. With close to 300 individual checks that can be grouped into company-specific checklists, the tool can help enforce clean and readable code while also pointing out potential issues such as bus contention and improper use of clocks and resets prior to simulation where they may be more difficult to find and fix.

Recommend RTL safety checks include:

- Avoiding FSM state unreachability, terminal states, and coding issues
- Avoiding differences between simulation and synthesis semantics
- Avoiding operations with expensive implementation costs
- Following naming and RTL coding conventions
- Enforcing RTL modeling clarity and reducing complexity
- Enforcing checks for clock bundles (clocks, enables, resets) and control signals
- Enabling testability and traceability of the code
- Avoid long ITE chains
- Confirm all IP ports are registered

SAFE FINITE STATE MACHINES

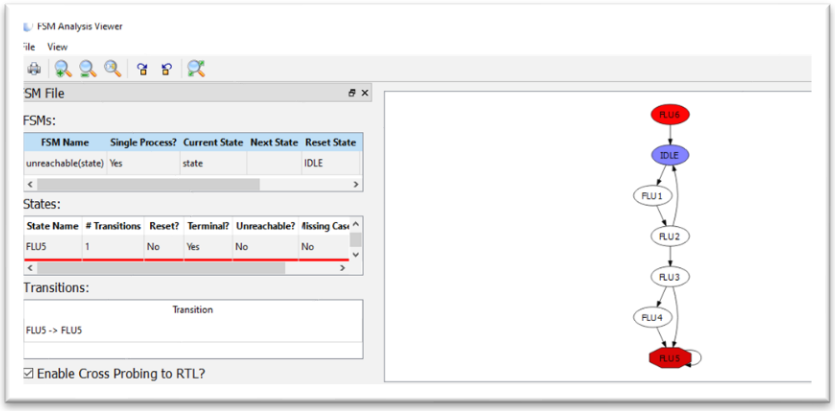
Finite state machines are widely used in the design of medical devices. Generally, when designing a state machine using RTL, the synthesis tools will optimize away all states that cannot be reached and generate a highly optimized circuit. Sometimes, however, the optimization is not acceptable. For example, if the circuit powers up in an invalid state, or the circuit is in an extreme working environment and a glitch sends it into an undesired state, the circuit may never get back to its normal operating condition.

The decision to implement a Moore or a Mealy machine will depend upon the function the state machine is required to perform along with any specified reaction times. The major difference between the two is in how the state machines react to inputs. A Moore machine will always have a delay of one clock cycle between an input and the appropriate output being set. This means a Moore machine is incapable of reacting immediately to a change of input. This ability of the Mealy machine to react immediately to the inputs often means that Mealy machines require fewer states to implement the same function as a Moore implementation would need. However, the drawback of a Mealy machine is that when communicating with another state machine, there is the danger of race conditions, which occur when the output is unexpectedly and critically dependent on the sequence or timing of other events. It is also possible to create a hybrid state machine that uses both styles to deliver a more efficient implementation of the function required.

The number of states in a state machine is always a power of two. This means that some states will not be used within the design. The designer is responsible for ensuring these unused states are correctly handled within the design.

Not all state machine designs are "unsafe." "Safe" depends on how many states are in a design and how you define the state encoding styles. If the number of states (N) is a power of 2, and you use a binary or gray-code encoding algorithm, the state machine is "safe". This ensures that you have M number of registers where N = 2M. Because all of the possible state values (or register statuses) are reachable, the design is "safe."

"Unsafe" State Machines can occur if the number of states is not a power of 2, or if they do not use a binary or gray-code encoding algorithm, e.g. one-hot, the state machine is "unsafe."



FSMs:

FSM Name	Single Process?	Current State	Next State	Reset State
unreachable(state)	Yes	state		IDLE

States:

State Name	# Transitions	Reset?	Terminal?	Unreachable?	1issing Cas
FLU5	1	No	Yes	No	No

Transitions:

Transition
FLU5 -> FLU5

☑ Enable Cross Probing to RTL?

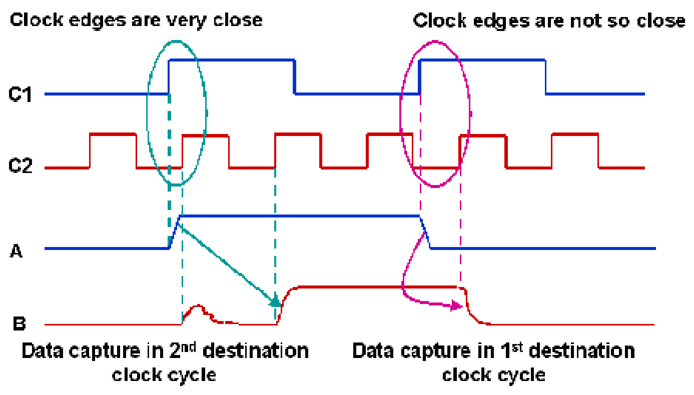
FSM Analysis with Unreachable State

Blue Pearl's Visual Verification Suite offers additional checks to ensure the FSM states are analyzed for dead and/or terminal states and a visual representation of each FSM is generated which includes the states and transitions. Leveraging a tool to help visualize the state interactions can help avoid unsafe state machines which may slip through undetected in a standard code review as well as add to the documentation of the design for design reviews and audits.

CROSS DOMIAN CLOCKING ISSUES

Metastability refers to signals that do not assume stable 0 or 1 states for some period of time at some point during normal operation of a design. In a multi-clock design, metastability cannot be avoided but the detrimental effects of metastability can be neutralized. This is especially important when designing medical devices as metastability can cause dire consequences.

A metastable output that traverses additional logic in the receiving clock domain can cause illegal signal values to be propagated throughout the rest of the design. Since the Clock Domain Crossing (CDC) signals can fluctuate for some period of time, the input logic in the receiving clock domain might recognize the logic level of the fluctuating signal to be different values and hence propagate erroneous signals into the receiving clock domain.


Clock edges are very close
Clock edges are not so close
C1
C2
A
B
Data capture in 2nd destination clock cycle
Data capture in 1st destination clock cycle

Clock Domain Crossing with Unsynchronized Signal

Simulation requires the generation of appropriate test vectors and is an accepted traditional method for functional verification during the design creation phase, however it may not catch CDC issues. While it is possible to find a CDC and then create a test vector, finding CDC issues using simulation is virtually impossible.

Modern Clock Domain Crossing tools point out where synchronizers are required to allow signals to cross clock domain boundaries. Correct operation of circuits crossing clock domain boundaries cannot be guaranteed by simulation because the timing between different clock domains can vary arbitrarily. This would require an infinite number of simulations to verify.

The Visual Verification Suite's Advanced Clock Environment (ACE) and CDC Analysis provide graphical representations summarizing data paths between clocks, and can make recommendations for grouping of clocks into clock domains. With ACE, designers can identify clocks to better understand how they interact with synchronizers in the design. This allows users to quickly identify missing synchronizers or improper clock domain groupings. The Visual Verification Suite CDC Analysis incorporates over a decade of experience to find errors other tools may fail to identify.

INDUSTRY'S ONLY STATIC VERIFICATION TOOL OPTIMIZED FOR FPGA AND ASIC

The Visual Verification Suite offers the only Linting and CDC verification environment optimized for both ASIC and the unique requirements of FPGA design. Specific for FPGA are checks to analyze for routing congestion, reset configurations and even estimate critical timing paths prior to synthesis. Grey Cell modeling supports the analysis of FPGA vendor and 3rd party provided protected IP cores with asynchronous clock domains. The Visual Verification Suite supports Xilinx Vivado™ Design Suite with built in UltraFast™ Design Methodology design rules, Microsemi Libero Design Suite and Intel Quartus® Prime Design Software and is the only static verification environment that runs on Windows – the preferred FPGA environment – as well as Linux operating systems.

For more information about Grey Cell Modeling see **RTL Analysis for Complex FPGA designs using a Grey Cell Methodology to Improve QoR**

For more information on Blue Pearl's Software's reliability testing see: **How Can We Build More Reliable EDA Software Whitepaper**

BLUE PEARL SOFTWARE

Blue Pearl Software, Inc. is a provider of design automation software for ASIC, FPGA and IP RTL verification. Its Analyze RTL™ linting and debug, Clock Domain Crossing analysis and Synopsys Design Constraints (SDC) generation solutions are proven to improve quality-of-results (QoR), reduce risk and decrease development time. The Visual Verification Environment complements RTL simulation solutions provided by EDA and FPGA vendors by ensuring code and SDC quality along with clocking integrity. Engineered to maximize RTL find/fix rates, the Visual Verification Suite uniquely provides easy setup, consistent results, Management Dashboard for complete push-button analytics, and runs on both Linux and Windows. The Visual Verification Suite is designed, tested and supported in the United States of America.

Revision History

Feb 20, 2017
New