DFT strategy for IPs
chintan panchal - January 08, 2016

Abstract

IPs (Building blocks of ASIC/SoC e.g. CPU, GPU) build and sign off in a wider sense. It doesn’t always mean Chip Timing, Design Rule Constraints & Power Closure of a block, but also refers to the creation of a layout/partition around the delivered build/sign off IPs/blocks. This would further refer to flows that are planned, library composition, various floorplan styles and shapes. The conflicts encountered in DFT and its equivalent solution, experienced during IP Block build (Hardening) are discussed in this paper. CPU - IP Block build (Hardening) accumulates many new DFT Flows & Methodologies for this purpose.

1. Introduction

1.1 SoC Methodology in Large Chip Design

Semiconductor companies typically face situations of unpredictable complexity during large chip design work which forces design engineers to switch to a fire-fighting mode rather than focusing on core design activities. While there are several methodologies used to tackle the challenges, System-on-chip (SoC) design, as a tried-and-tested methodology, has seen great success in large chip design. Simplicity is the key here: The SoC methodology involves the usage of predesigned and preverified blocks which are sourced from in-house teams or/and outside contractors which when combined together, becomes semiconductor intellectual property (IP). This greatly reduces development test cycle time thus, ensuring faster time-to-market for the final product.

1.2 What is IP Build (Hardening)

Over the past one decade, as integrated circuits have augmented in costs and complexity, system-on-chip (SoC) design has gained greater currency. Nowadays, progressive IP vendors provide preverified software and hardware IP for the required environment. Thus, integrating and reusing them into Soc Design saves considerable design and verification effort. Reusing proven designs is the core idea behind IP build - also known as “hardening” whose aim is to greatly reduce fabrication risks in hardware. Large productivity gains can be achieved using the IP build approach. Indeed, rather than implementing each and every component independently, it’s more
effective to combine them together on a chip to derive intricate functions in a relatively short amount of time.

1.3 How is IP Build (Hardening) Achieved

The integration process in IP Block build (hardening) involves connecting the sub blocks to the communication network data path, and implementing Design-For Test (DFT) techniques which can be used at SoC level across existing test platform procedures in order to save on time and costs.

Essentially, IP build – hardening is a parallel development process along with SoC/Chip design. Once IP build is done, it can be used as a hard macro to be placed on die as per floor plan. Hard macros are silicon tested and proven block designs which are generally in the form of hardware IPs. There is no need to open this build IP (hard IP) from the perspective of layout/timing/test.

1.4 Core Types

The actual form of a reusable IP core can vary depending on the manner in which IP developers/vendors decide to provide integrating cores to the chip designer. There are three main categories here: Soft Core, Firm Core & Hard Core.

1.4.1 Soft Core

It starts with soft IP blocks being described using RTL (Verilog etc.) or other higher level languages. They are more appropriate for inner cores since a Hardware Description Language (HDL) is process-independent. The basic advantages of RTL include *flexibility, portability, and reusability*, with the disadvantage of not having definite timing or power capabilities, since any deployment in different processes and applications produce performance variations. Also, HDL is obtained from outsourcing vendors in an encrypted form which does not allow it to be customized. This makes it harder to be adapted in unforeseen circumstances, while preventing designers from introducing any new errors into the block. Soft core also contain other soft core, hard core or standard cells.

1.4.2 Firm Core

These blocks are in the synthesized gate level netlist form. Provided as parameterized circuit descriptions, designers can optimize cores for their specific design needs. These flexible parameters allow the designers to make the performance more predictable. Firm IP offers a compromise between soft and hard cores as it’s more flexible and portable than hard IP and yet more predictable than soft IP. Firm IP blocks are less flexible than soft IP.

1.4.3 Hard Core

These blocks have set layouts and are exceedingly optimized for a known application as per a
precise process at the physical design stage. They target specific IC manufacturing technologies and more predictable in terms of power, area and timing. They provide fixed configuration and do not offer as much flexibility as firm core or soft core. They have the individual benefit of having expected performances. This, however, comes with extra exertion and costs, and lack of portability that may to a great extent, limit the areas of application. This form of IP is usually pre-qualified, which means the provider has already tested it in silicon.

![Figure 1.1 Core Types](image)

2. **IP Build - Hardening Flow**

For creating SoC designs, we need an array of reusable IP blocks that sustain plug-and-play integration. IP blocks are the top level building blocks of an SoC. A library of reusable IP blocks with various **timing**, **area** and **power** configurations is the key to success in SoC design. SoC integrators can apply the swapping that is most appropriate for the needs of end user applications. The process of creating a reusable IP block, nevertheless, differs from the traditional ASIC design approach. Normally, it may require five times as much work to generate a reusable IP block in contrast to a single-use block.

### 2.1 Reusable IP Formation

#### 2.1.1 Digital IP

Digital IP blocks are the most popular form of reusable IP in industry today. The three stages in the design process are:
Following specifications and documentation of the reusable IP;
- Implementation using standardized coding practices;
- Full verification including code coverage and behavioral (or functional) coverage.

The first step involves the generation of suitable documentation and specification for the IP block. The second step involves code design, synthesis, and design for test (DFT). The third step involves full IP verification with a goal of negotiated 100% code coverage and functional coverage. In fact, depending on the size and type of the IP, the third step of verification may take up to 50-70% of the total time.

### 2.1.2 Analog/ Mixed Signal IP

When analog and digital blocks coexist on the same substrate, the analog portion of the design can be more protracted to develop even though it may represent a smaller percentage of the chip area. It is anticipated that more than 70% of all SoC designs will have some form of analog/mixed-signal blocks. This increase is consistent with the expected growth of the wireless industry in the same period.

### 2.1.3 Programmable IP

As SoC Size becomes larger and larger, the design costs are so high that it becomes vital to fit in programmability within the SoC to allow for reuse at the chip level. This programmability can come into view in a number of forms: hardware programmability using programmable logic cores and software programmability using an embedded processor. The key to programmable SoC designs is to offer some form of flexible hardware and/or software infrastructure, often called the programmable fabric. We differentiate between these two types of flexibility: pre-fabrication and post-fabrication.

### 2.1.4 On Chip Communication Infrastructure - Core Types

Figure 2.1 shows the typical SoC structure which contains digital, analog and mixed-signal IPs communicated through different bus protocols depending upon the application. It contains memory controller, watchdog, GPIO IPs in the form of soft core and memory module in the form of hard core. Generally, memory modules are used in form of hard core macro in SoC.

Now, all the Soft IPs are potential candidates for Hard IP creation, in order to build SoC effectively.
2.2 DFT Strategy - IP Build (Hardening) Stages

Here, we discuss one of the most efficient DFT flows used for IP development at eInfochips.
Sign Off: Quality checks

- Name Rule Check on Netlist Instance, Netlist Modules and Signal name. Length of name/string should be within specified characters
- Tie Cell Insertion for Power/ground Ports
- Electric Rule Check by Parsing netlist to find floating nets and high fanout nets and existence of clock delay cells

MBIST: EDA tool specific

- ET_Checker – Parse netlist with compatible library
- ET_Planner - Memory grouping done.
- ET_Assembler - Generates specific database - programmable soft core controller, Built in algorithms, (covered fault models), synthesize it
- ET_Sign_Off - Verifies Per Controller Pattern Sequences & Mapped Pattern Files from Chip Top

SCAN: Adding controllability and observability

- Advanced Test Compressor for reducing test application time, reduce vector size per pattern, favoring ATE Memory.
- Check design rule constraints before scan insertion and after scan insertion to meet technology specific restriction.

ATPG: Test vector generation

- Generates structural patterns depend upon the fault model.
- Basic Scan Sequential Engine & Ram Sequential Engine & Latch Transparency engine with Single pulse Per capture cycle & multi pulse (~up to 10) for TR
- OCC Enabled two clock pulse supported test proc.
- Also generates Verilog test bench through which we can simulate generated patterns on scan inserted netlist.

Spare: Layout Timing closure

- 0.2% of spare cells, which includes all std cells & memory cells for post metal eco purpose

Timing Analysis: DFT mode timing closure

- Zero Wire Load Model with Idle Clock tree setup calculation & Hold under Control.

Simulation: Vector validation

- Simulate generated ATPG patterns on scan inserted netlist
- Perform zero delay simulation for Flush, SSA and TR patterns (compressed & uncompressed) on pre-layout netlist
- Perform unit delay simulation for Flush, SSA and TR patterns (compressed & uncompressed) on post-layout netlist.
2.3 Hard IP Integration Platform

Figure 2.3 shows pictorial view of DFT architecture implemented around IPs in SoC/ASIC. The Input and Output switch box provides hook up test mode signals to top module pins. Memory Tessent MBIST tool of Mentor Graphics provides memory testing and repair functionality. EDT/ATPG scan control provides the scan insertion in IP and generate ATPG patterns test different faults in IP.

2.3.1 Test Integration Platform Components

Test Integration Platform is a Switch Box like a Controller, which is customized based on Architect Implementation/Service provider perspective. This can be further accessed via testers to access chip from the outside for different modes of testing.

2.3.1.1 Combinational Mode decoder
We need to count the total no. of Digital/Analog/Outsource IPs prepared for testing and based on total grouping.
We need to decide the total no of Test Mode pins that would be used to select particular DFT IP Modes for testing.

2.3.1.2 Sequential Mode decoder

On the basis of grouping, we need to count the total no. of Digital/Analog/Outsource IPs prepared for testing.
We need to decide Clock & test Mode two pins to start selecting particular modes during testing. Here, the sequence of Test Mode registers will be available for selection purpose.

2.3.1.3 TIP Input Switch Box

TIP input switch box controls all signals that enable us to perform operations.
It provides hook up for all test input signals of each IP all the way to the top module pins of SoC to control it from the top level.
It facilates the user to run IPs in either normal mode or testmode.
IPCON separate tesing is performed to check access controls/observe from top module.

2.3.1.4 TIP Output Switch Box

TIP output switch box is initiated normally from the top of module & observes all signals that enable us to perform operation.
It provides hook up for all test output signals of each IP to top module pins of SoC to observe it from the top level.
IPCON separate tesing is performed to check access controls/observe from top module.

2.3.1.6 Memory Tessent MBIST

Logic Vision TAP is generated along with generation/insertion of BIST and embedded in Netlist.
Planner, Assembler & Simulation sign_off stages all ports are hooke_up with switch input box & switch output box.
Corresponding Test Control Signals & Test observe signals from MBIST Mode to ASIC Top Signals are hooked up.

2.3.1.7 EDT : Scan

EDT related ports (edt_update, edt_shift, edt_clock, edt_bypass, edt_reset, edt low power shift controller & edt low power controller enable) are hooked up from top (shared with IO pins).
Scan Channels (Input & Output Mode are also shared with top Ports)
Scan Reset, Scan Test, Scan Enable, Scan Clock & DFT Fix Clock all needs to be shared from top.
Check Design rules constraints before and after scan cell insertion.

2.3.1.8 EDT : ATPG

ATPG Test mode needs to be defined for putting design under Pattern generation Mode.
Set different fault models for which we need to generate ATPG pattern.
Generate Patterns from Top Scan Channels (port io sharing) to ATPG engine.
Write generated ATPG pattern in to defined format (i.e. STIL, WGIL or verilog).

2.3.1.9 Wrapper Chain Selection
• Wrapper chain facilitates users to test individual core logic in SoC from the top level of design.
• Wrapper chain needs to be stitched around hardened sub blocks to be accessed via test mode with individual testings.
• Wrapper test with Intest mode & Extest mode needs to be tested from the top Architecture mode.

2.3.2.0 General Hardened Integration Issues

• Proper Pad Selection is the main goal of hook_up/ports with particular test based actions.
• Tri-State control selection needs to be seriously taken care of for the purpose of direction control while sharing.
• Between Modes, Port selection sharing needs to be done with limited actions.
• Package Pin Access file needs to be taken in to account for selection of sharing purpose, which would help in greater routing congestion reduction.
• IP connectivity testing needs to be done, for each test modes before we start FTIP Access from tester or during any FDK stage.

2.4 Summary

This paper has provided a perspective on the hardening issues during integration associated with mixed-signal SoC design. Reusable forms of digital, analog/mixed signal, and programmable IP components were described. Integration issues associated with interconnect, ASIC Top Hook_UP, Test Mode Connectivity Testing(IPCON) and verification were presented. The author believes that almost all designs in the future will make use of reusable IP and that commercial tool vendors will continue to advance their tools to address the more challenging issues of system level hardware/software codesign and co-verification.

2.5 Acknowledgement

The author wishes to thank his team & research engineers for the significant contributions to the work at the DFT Team, eInfochips, Ahmedabad. The authors also wish all viewers for invaluable discussions regarding SoC design and reuse.