INTEGRATION, the VLSI journal 49 (2015) 125–136

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Synthesis-based design and implementation methodology of high-speed, high-performing unit: L2 cache unit design

Mozammel Hossain [a,*], Chirag Desai [a], Tom Chen [b], Vikas Agarwal [a]

[a] International Business Machine (IBM) Corporation, Austin, TX 78757, USA
[b] Department of Electrical and Computer Engineering, Fort Collins, CO 80523, USA

ARTICLE INFO

Article history:
Received 12 June 2014
Received in revised form
29 September 2014
Accepted 2 October 2014
Available online 14 October 2014

Keywords:
LBS
RLM
Physical design
Soft hierarchy
Interior pin

ABSTRACT

We propose a physical design methodology for synthesis using soft hierarchy, interior pin placement, pre-placing critical logic, and routing techniques on a very timing- and area-challenged unit, the L2 cache, with ~20 million synthesizable transistors. In any past and present standard design at IBM, this test case would stretch all front- and back-end design tools by two to three times due to data volume, congestion and timing criticality, which opens a new avenue to explore for Large Block Synthesis flow. The results confirm the ability to deliver a design in the shortest possible schedule at ~50% of Physical Design cost while still maintaining best-of-breed quality.

1. Introduction

Today's high-speed and high-performing L2 cache unit design plays a crucial part in the microprocessor industry. Due to the physical limits to its area and speed requirements, a high-speed and area-efficient L2 cache design demands custom macros, which require not only highly-skilled custom circuit designers but also time to design these macros. Fig. 1 shows a typical L2 cache unit micro-architecture which contains mainly four functional areas as follows: 1) cache array macros, 2) address macros, 3) data path macros and 4) control macros. The widths and depths of these array, address and datapath macros depend on the micro-architecture of the cache unit. Traditionally, array macros are designed using full- and semi-custom approaches where every transistor is tailored for its specific application. Typically, the effort is in the order of 1–2 person-years (PY) per array. Similarly, most of the peripheral address and data path macros are hand-crafted custom or semi-custom macros. Although these data flow macros are not as critical as arrays, due to area and timing pressure, these macros are done with custom design flow.

However, all of the random logic macros, especially for control areas, are built using automated tool flow, such as synthesis, in a typical microprocessor design methodology.

In the L2 cache unit, 'Array Banks' parts are repetitive cache macros tiled together to build a cache function. This design demands about one-third of the highly skilled man power for array design and two-thirds of the resources for the rest of the unit. This means that major reengineering is required when moving from one technology/design point to the next. This work aims to find a design methodology to improve the two-thirds of the unit area. In this paper, we present a Large Block Synthesis (LBS) design methodology and walk through step by step our proposed physical design methodology. We show how development cycle and custom-circuit-design resources can be cut by ~50% while maintaining the best-of-breed quality. We discuss possible elimination of timing and integration resources that are needed in today's design methodology to deliver a high-quality routing with timing-closed unit to the chip. We also explain how synthesis-based design methodology improves power efficiency by looking at logic gates and drive strength across the macro boundary. We show how localized macro congestion can be resolved at the unit level by sharing routing layer resources. While we study the design methodology for the L2 unit, we explore, develop and recommend how pre-routing, pre-placement and 'soft hierarchy' (SH) techniques can be used as aids to resolve critical timing and routing issues that are commonly encountered.

* Corresponding author. Tel.: +1 512 286 6708; fax: +1 512 973 4286.
E-mail address: mozammel@us.ibm.com (M. Hossain).

http://dx.doi.org/10.1016/j.vlsi.2014.10.001
0167-9260/

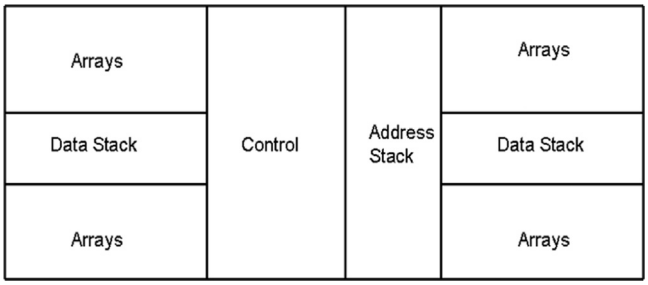

Fig. 1. Generic L2 cache unit architecture.

We lastly present data on different macro design topologies and show how macro design methodology has evolved over the last 16 years, as well as how LBS methodology is becoming the mainstream design flow at IBM.

2. Problem definition

2.1. Custom macro design approach

To meet design constraints such as area, timing and power, data flow macros most often need hand-crafted schematics and layout design. The custom design approach is overwhelmingly time consuming and thus requires as much as four times the development time of the synthesis flow. In general, a custom designer has to plan every detail, including physical placement of the design, up front. The problem with a custom design approach is incorporating any late design change may occasionally cause a major rip-up of a previously built and timed macro. Fig. 2 shows a typical custom macro design flow and its dependencies for all phases of design cycles. In this flow, almost each step requires a good amount of hand holding and manual intervention and thus becomes very time consuming.

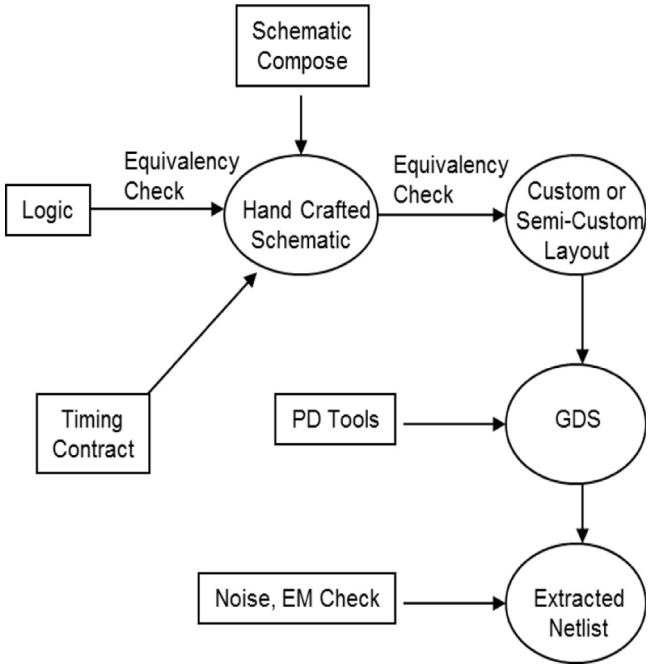

Fig. 2. Custom macro design flow.

2.2. Fully synthesized macro design approach

Compared to custom macros, synthesis-based design is primarily for non-structured control logic and is automatic tool based (Fig. 3) with faster turnaround time. Late changes in the design can easily be incorporated in an automated way, keeping the rest of the design mostly unaffected. Today's synthesis engines do a very good job in closing timing, electrical and noise constraints for low-latency and random-logic macros. Synthesis does not do a good job for structured, high-frequency and timing-critical logic that needs to be packed in a tight floorplan. In a typical synthesis flow, each macro is individually tuned to make timing and Physical Design (PD) rule checks clean for unit level work, still requiring a fair number of PD resources for macro closures. This methodology still needs the same unit integration and timing support as the custom macro design flow requires.

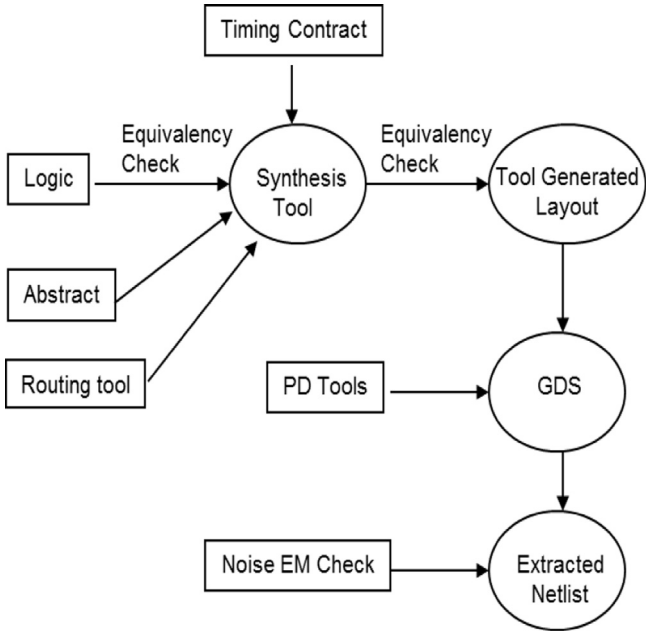

Fig. 3. Fully synthesized macro design flow.

2.3. Design optimization and physical design (PD) resources

Both the custom and the synthesis flows at lower level macros are constrained by logic partitions and timing windows. In both design flows, each macro is tuned individually, which may not necessarily be a good solution for overall unit design. Most often, macros are designed and developed by a group of people who lack understanding of each other's macros or timing constraints, which can lead to a sub-optimal design. For example, redundant logic or unnecessary drive power at macro boundaries could end up missing timing arcs and wasting power. Bug fixes and late changes in an individual macro design most often become very expensive in terms of turnaround time. Units like L2 cache could have ~500 physical partitions (approximately 10 arrays, 40 custom/semi-custom and 40 synthesizable unique macros, some of which are being used multiple times) with many timing-critical paths that require more physical design resources. These problems can be efficiently resolved by developing the design methodology for Large Block Synthesis (LBS), as discussed in later sections of this paper. Proposed techniques for LBS methodology can take advantage of breaking hierarchies for all non-array macros to optimize logic gates and their placement to close highly timing challenged paths. LBS methodology enables downsizing of the PD resources for both custom and synthesizable macros and eliminates the need for unit integration and timer resources.