TOOLCHAIN INTEGRATION

William Bricken
March 2002

BILD Products Tool-Chain Integration

This memo addresses the ways in which the BILD reconfigurable chip and its associated optimizing compiler software would fit into conventional EDA tool-chains and design flows. Two example scenarios are considered: substitution for existing programmable components, and the use of BTC products in the design of new semiconductor products. Three Figures representing conventional and BILD design flows follow the narrative.

Substitutes for Existing Components

We assume that our customer already has product that uses reconfigurable chips (SPLDs, CPLDs, or FPGAs) for which we had designed standard competitive products that are available in appropriate packages and pin configurations. Another possible scenario is the use of our products as replacements for non-reconfigurable ASICs and for slow microprocessors.

Upon providing identifying information, the customer would receive authorization to download the BILD optimizing compiler software from our website. The software will easily install in any supported OS environment (most likely versions of Windows or Unix). The customer would initiate a "compile" action.

Interactively, the BILD Tools would

1) identify the location and format of the existing netlist or HDL specification,
2) identify a location to store the generated CCA configuration and the standard netlist,
3) query for setting compiler flags,
4) estimate the processing time, and
5) construct and store the CCA configuration and its netlist.

During their manufacturing process, the customer would replace the competition's programmable chip on the PCB with the BILD hardware. We assume physical compatibility. Depending upon the customer's choice for configuration file storage, the CCA configuration would be loaded into the BILD hardware on the PCB. Possible configuration file storage techniques include a PROM on the PCB or a dataline into the PCB.

For customers who use third-party model testing and verification suites, the timing, capacity and performance characteristics published in the BILD

product specification sheets can be entered directly into their abstract model. The BILD architecture assures that these specifications can be met. For tool compatibility and upon request, the BILD optimizing compiler would also return a timed or untimed netlist with pin-level backannotation and a NOR-gate technology library. This integrates BILD product performance models with third-party, system-level design and verification tools.

During PCB design testing with the BILD product, timing and performance would be assessed in the context of the rest of the board. The BILD chip can be run at those various clock speeds that integrate with other board components. It is expected that the BILD chip will support clock speeds in excess of other board components, and thus will not be a design bottleneck. Since the BILD product will always provide more real, useable gates than existing PLDs, fitting the functionality into the BILD part is expected not to be an issue.

Should a BILD chip already be installed, to modify its functionality a designer would regenerate the BILD configuration file from the new HDL specifications, and load it into the BILD chip. Timing and other performance characteristics would not change.

Design of New Semiconductor Products

- *Design of single clock functions*: Functionality would be specified in HDL format as if it were to be submitted to any PLD configuration software. HDL files would be submitted to the BILD optimizing compiler as above. During initial design iterations, while functionality is being specified and debugged, BILD hardware development boards would be available. Design iterations would be resubmitted to the BILD compiler, however the performance characteristics of the BILD chip would not vary. BILD has no place and route steps and no physical layout steps; instead, similar to SPLDs, its performance characteristics would remain fixed during design iterations.

- *Design for logic integration*: During design of a PCB which contains several single clock chips, a designer would estimate the total gate requirements for all chips, with the intention of purchasing a BILD chip that would accommodate them all. PBC circuitry and bussing that supported the physical integration of the several separate chips and their respective clocks could be deleted from the design. The single BILD chip running under the single clock would replace them all. The BILD chip clock could be expected to support rates in excess of other board components.

- *Design for microprocessor replacement*: The functionality which was previously specified in a microprocessor assembly language, or in a high-level language such as C, would be specified instead in the synthesizable but untimed subset of an HDL. In many cases, this would simply be the high-level specification of the FSM or the logic equations which define functionality

and which themselves would be used to guide the unnecessary C implementation. Third-party C-to-HDL conversion tools might also be used. The BILD chip would replace the microprocessor on the PCB, with accompanying increases in processing speed.

• *Design for IP cores*: IP cores available through future IP partners would be purchasable as BILD configuration files. Exact performance specifications would accompany the BILD/IP-partner product.

DESCRIPTION OF FIGURES

Figure I shows two generic, simplified design flows, one for standard PLD products and the other for BILD products. Design capture and simulation tools are the same for both design flows. The BILD optimizing compiler manages design constraints, technology libraries, static timing analysis, and physical fitting and optimization for the BILD chip architecture. Unlike other PLDs, BILD chips do not complicate design by imposing device-specific constraints or library requirements. The static timing of a given BILD architecture product is set in advance and does not vary since logic fitting or mapping into fixed resources and signal routing are not required.

Figure II shows a more detailed generic design flow, comparing each step to the BILD design flow. Design specification and capture remain the same, although the BILD optimizing compiler can accept any abstract formal behavioral description, such as an FSM or a set of Boolean equations. Existent design logic and specifications that may have been used to define expected functionality at a higher-level than the RTL description characteristic of HDLs need not be converted into HDL.

BILD synthesis, optimization, and formal verification tools are built into the BILD optimizing compiler. They provide the same functions as do standard tools, with three significant differences. First, the BILD compiler is completely formal, and cannot introduce design error given a golden standard specification. Second, BILD tools and transformations are much simpler than standard tools since the BILD chip architecture does not require physical layout across fixed capacity logic blocks and labyrinthine interconnect. Finally, the behavior of the BILD hardware is stable and predictable during design changes. *Available on demand* functions are generated automatically by the BILD optimizing compiler and are included for compatibility with existing design practices.

BILD tools can return a backannotated timed or untimed netlist specification which could be reentered into a traditional design flow. The BILD technology library, prior to generation of the CCA configuration file, is conventional NOR gates, providing complete cross-compatibility with conventional tools. BILD tools provide all standard verification steps, assuring at each step that the design meets expectation.

Figure III provides a comparison between BILD tools and those advertised for sale by Altera for use in programming the Altera FLEX and MAX product lines (this information is dated, but still provides a useful generic comparison). Design entry tools are available for both companies' products from third-party vendors. Some Altera tools are not required for BILD products, many are replaced by the BILD optimizing compiler. The BILD tools functions are interactive. One significant comparison is that we intend to provide its optimizing compiler without cost to its chip customers, thereby probably reducing software costs to customers by thousands of dollars.

Figure I: Generic Design Flow Comparison, Standard PLDs to BILD Products

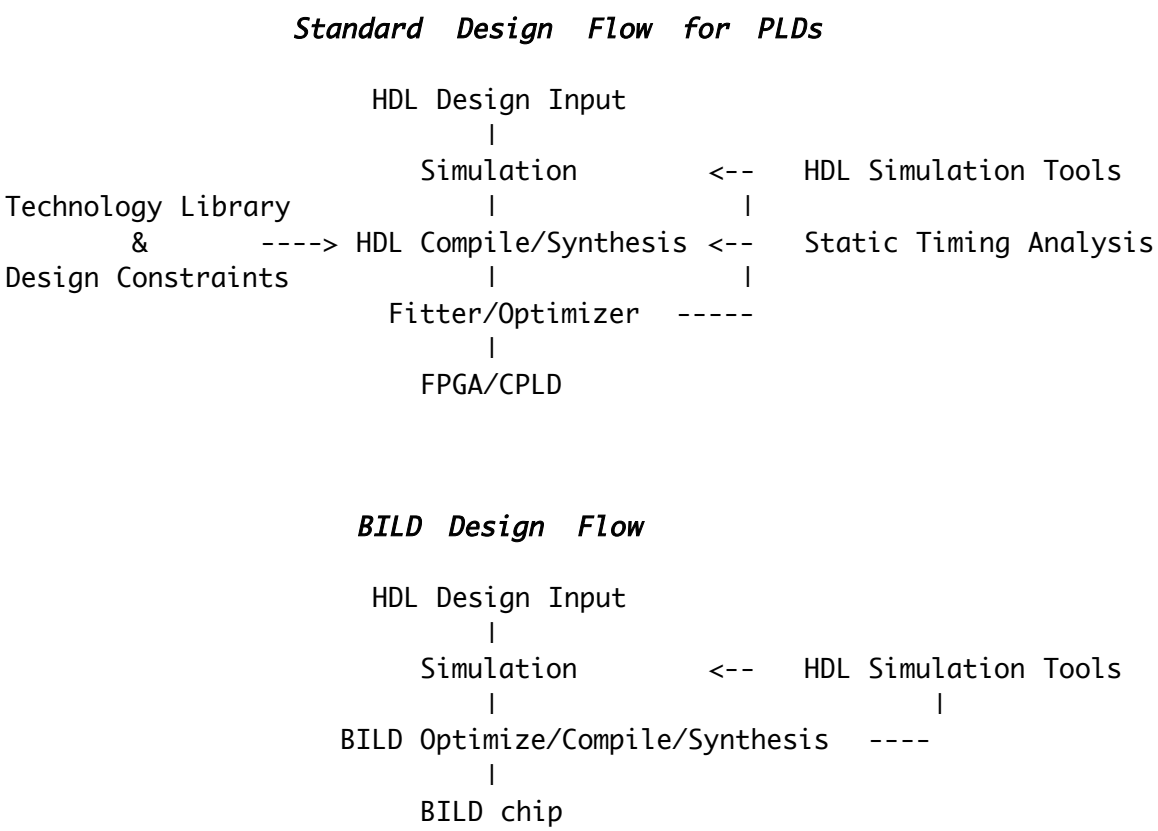

Figure II. BILD Tool-chain Integration

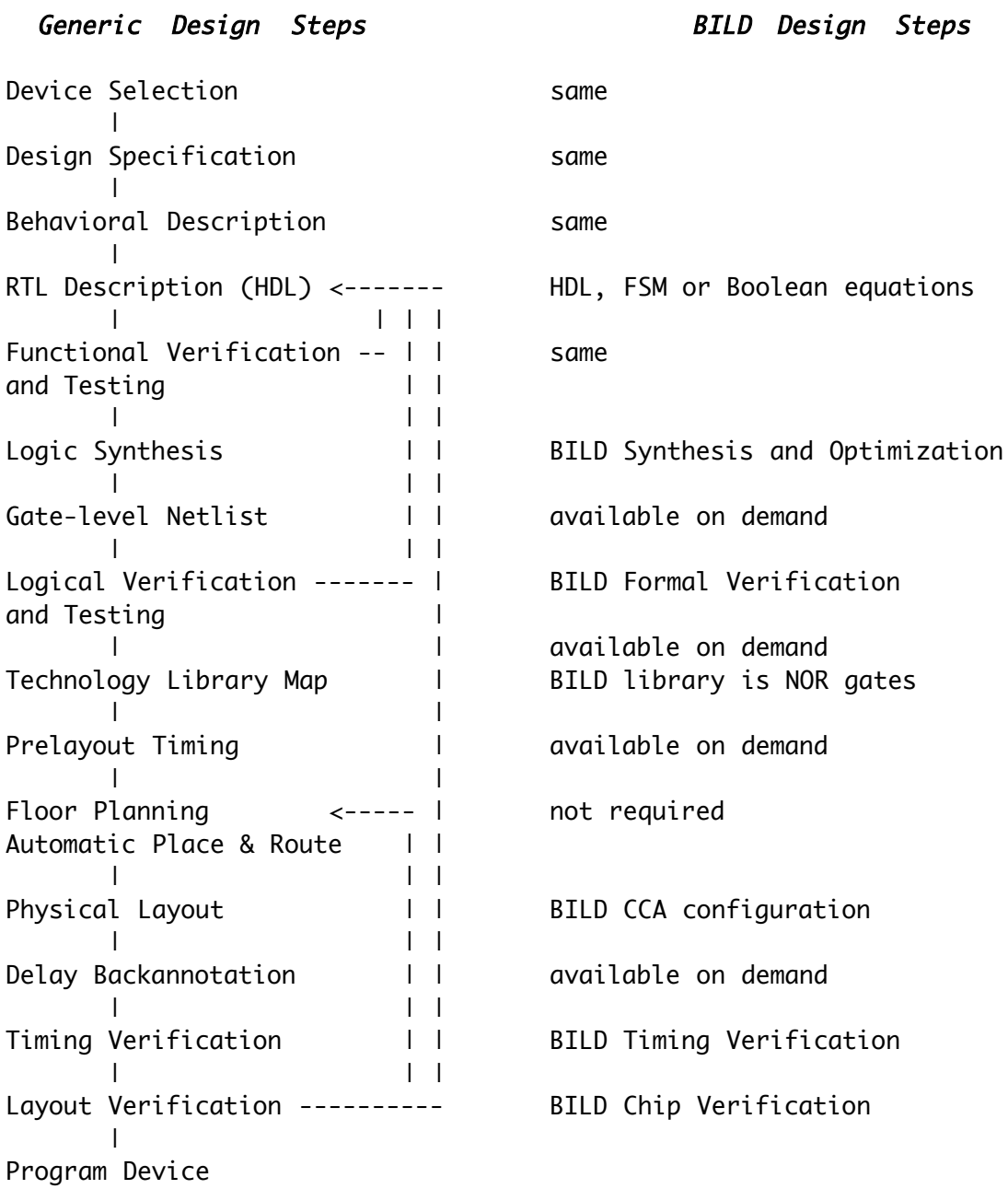

Figure III. Software Design Tool Comparison (Altera vs. BILD)

	Altera	*BILD*
Design Entry		
	Schematic Design Entry	*Third-party*
	HDL Design Entry	*Third-party*
	Waveform Design Entry	*Third-party*
	Netlist Interfaces	BILD Optimizing Compiler
	Floorplan Editor	*not required*
	Hierarchical Design Management	BILD Optimizing Compiler
Design Compilation		
	Timing-driven Compilation	BILD Optimizing Compiler
	Logic Synthesis/Fitting	BILD Optimizing Compiler
	Automatic Error Location	*not required*
	Design Rule Checking	BILD Optimizing Compiler
	Multi-chip Partitioning	BILD Optimizing Compiler
Design Verification		
	Timing Analysis	BILD Optimizing Compiler
	Waveform Editing	*not required*
	Functional Simulation	BILD Optimizing Compiler
	Timing Simulation	BILD Optimizing Compiler
	Multi-chip Simulation	BILD Optimizing Compiler
	On-line help	BILD Optimizing Compiler