

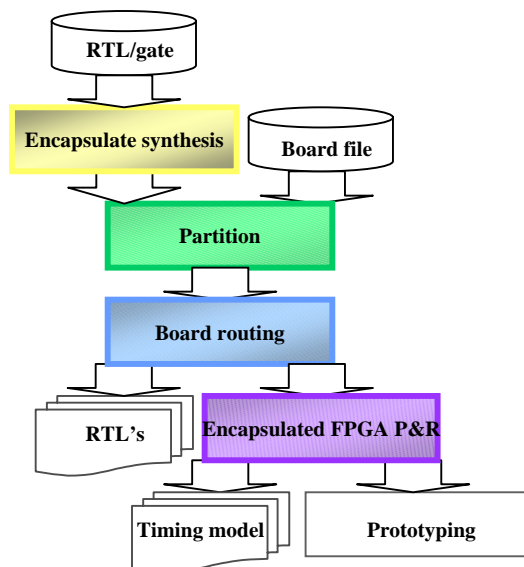
## Overview

Mapping today's complex ASIC or SoC onto multiple-FPGA prototype presents the challenge of finding a partition that

- (1) meets the constraints of the target prototyping hardware such as FPGA capacity, clock distribution, and limited board traces in-between FPGA's.
- (2) achieves the target prototype performance.

## Mapping flow

**ACE Compiler**, with its hierarchical timing-driven technology, produces solutions which not only efficiently map big designs into the target prototyping platform, but also run at top speeds.



### **ACE Compiler flow**

The design is imported modularly through the encapsulated synthesis with the commercial FPGA synthesis tool. The partition step breaks down the design into multiple FPGA's with the optimized timing, while honoring the platform constraints. The board routing then assigns inter-FPGA signals to traces, cables, switches and consequently fixes the FPGA pin locations. The encapsulated FPGA P&R generates the programming bit-streams.

FPGA RTL's may be exported as the other option in order to further optimize timing on individual FPGA's with the FPGA synthesis.

The whole process can be completely automatic or done manually on a portion of the design for reasons such as anticipation of future design changes, target system interface, or preferred grouping and then followed with the automation to finish the mapping.

## Automatic partition

**ACE Compiler** distributes clocks across the prototyping platform with the minimum skew. Schemes of loop-back, generator duplication or isolation are applied on internally generated clocks in order to balance the skew reaching every FPGA.

**ACE Compiler** can also limit the number of clock domains partitioned into each FPGA to ensure sufficient clock buffers to drive every clock in FPGA's.

Gated, latch-generated, and divided clocks will be automatically converted. The conversion can be cascaded or crossing the hierarchies.

Combinatorial hops crossing FPGA's will be minimized by the auto-partition in order to achieve the highest prototype performance.

To achieve the overall target prototype performance, the timing budget tool will calculate and prepare timing constraints on every FPGA to be synthesized, placed and routed.

## Wire-sharing

**ACE Compiler** inserts wire-sharing logic to relieve FPGA pin limits. The higher performance can be achieved with the domain-based wire-sharing and the exclusion of the combinatorial hopping signals. A variety of custom wire-sharing schemes are supported. Wire-sharing signals are automatically prepared.

## Hierarchical approach

**ACE Compiler** adopts hierarchical approach on all its operations in order to manage the design complexities. The ability to work at the gate-level for the timing and logic optimization purpose without flattening the design is one of the major features of **ACE Compiler**. **ACE Compiler** is capable of handling designs over 100M gates.

## Automatic Board routing

**ACE Compiler** will automatically route through FPGA's to complete the connections for those signals without the direct paths to the target FPGA's.

**ACE Compiler** is capable of assigning board connections through cables, field-programmable interconnect devices or bus switches.

If the design has more tri-state buses than the hardware platform supports, **ACE Compiler** implements each tri-state bus in a focal FPGA after routing all the tri-state and enable signals from driving partitions to this FPGA.

Clocks will be assigned to low-skew traces automatically. LVDS board traces are automatically identified and assigned to those LVDS-pair signals.

## Probe

**ACE Compiler** is able to bring out any design signal for observation. Probes can be brought to the connector pins or to the instrumentation module inserted by **ACE Compiler** if instructed to do so.

## Parallel execution

Several time-consuming steps, such as synthesis, partition optimization and FPGA P&R, in **ACE Compiler** can be sped up with the parallel execution on multiple machines across the network.

## Features

- Automatic partition and board routing.
- RTL in Verilog or VHDL, gate-level, or mixed language.
- Encapsulated synthesis with commercial FPGA synthesis tools.
- Parallel synthesis, partition and FPGA P&R.
- Hierarchical approach handles very big designs.
- Timing-driven partition produces low-skew clock distribution, converts gated clocks, minimizes the combinatorial hops crossing FPGA's and selects signals for the high-performance wire-sharing.
- Detailed reports on the timing characteristics of the FPGA prototype including clock distribution, combinatorial hops and static TA report.
- Timing budget to generate FPGA timing constraints to achieve the target prototyping performance.
- Optimized partition find solutions with the lowest FPGA pins and minimal combinatorial hops.
- Global logic trimming.
- Fast router embedded in partition to find the solution best matched with the board interconnect.
- Wire-sharing signals are automatically prepared.
- Automatic routing through FPGA's, cables and programmable bus switches.
- Probing any internal signal.
- Simulatable at every step.
- GUI or script execution.
- Work for custom or commercial boards.

## Integration

The encapsulated synthesis is integrated with commercial synthesis tools from Altera, Mentor Graphics, Synopsys, and Xilinx.

The encapsulated FPGA P&R is integrated with Altera and Xilinx P&R tools.

**ACE Compiler** supports FPGA's from the Altera StratixII/StratixIII/StratixIV/StratixV families and Xilinx Virtex4/Virtex5/Virtex6/Virtex7 families.

## Platform

Linux, Windows and Solaris