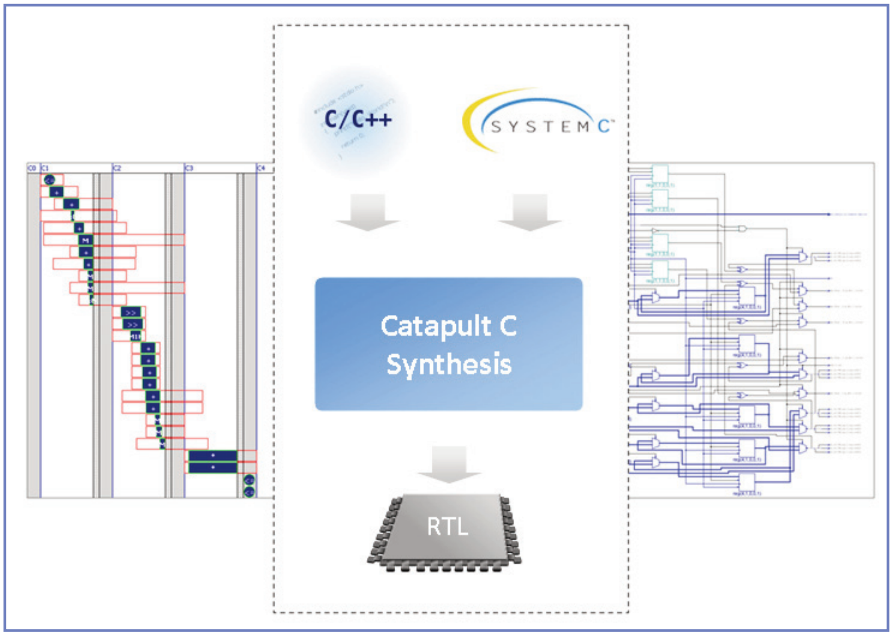
Catapult C Synthesis

C-Based Design

DATASHEET



Catapult C Synthesis produces high-quality RTL implementations from abstract specifications written in C++ or SystemC, dramatically reducing design and verification efforts.

Tackle Complexity, Accelerate Time to RTL, Reduce Verification Effort

Traditional hardware design methods that require manual RTL development and debugging are too time consuming and error prone for today's complex designs. The Catapult® C Synthesis tool empowers designers to use industry standard ANSI C++ and SystemC to describe functional intent, and move up to a more productive abstraction level. From these high-level descriptions Catapult generates production quality RTL. With this approach, full hierarchical systems comprised of both control blocks and algorithmic units are implemented automatically, eliminating the typical coding errors and bugs introduced by manual flows. By speeding time to RTL and automating the generation of bug free RTL, the Catapult C Synthesis tool significantly reduces the time to verified RTL.

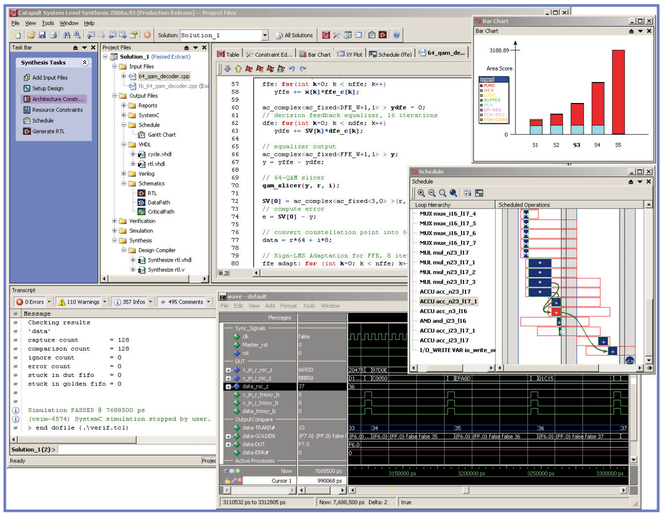
Catapult's unified flow for modeling, synthesizing, and verifying complex ASICs and FPGAs allows hardware designers to fully explore micro-architecture and interface options. Advanced power optimizations automatically provide significant reductions in dynamic power consumption. The highly interactive Catapult workflow provides full visibility and control of the synthesis process, enabling designers to rapidly converge upon the best implementation for performance, area, and power.

The Catapult solution has been used in the successful tape out of hundreds of ASICs and FPGAs by major companies around the world, with over 170 million

Major product features:

- Mixed datapath and control logic synthesis from both pure ANSI C++ and SystemC
- Multi-abstraction synthesis supports untimed, transaction-level, and cycle-accurate modeling styles
- Full-chip synthesis capabilities including pipelined multi-block subsystems and SoC interconnects
- Power, performance, and area exploration and optimization
- Push button generation of RTL verification infrastructure
- Advanced top-down and bottom-up hierarchical design management
- Full and accurate control over design interfaces with Interface Synthesis technology and Modular IO
- Interactive and incremental design methodology achieves fastest path to optimal hardware
- Fine-grain control for superior quality of results
- Built-in analysis tools including Gantt charts, critical path viewer, and cross-probing
- Silicon vendor certified synthesis libraries and integration with RTL synthesis for predictable backend timing closure
- ASIC and FPGA technology aware scheduling for high-performance hardware
- Broadest C++ language support including classes, templates and pointers
- Maximize IP and reuse potential with C++ object-oriented encapsulation

ASICs shipped by the end of 2009. Catapult was recognized by Gary Smith EDA as the high-level synthesis (HLS) leader for three years running.



```
ffe: for(int k=0; k < nffe; k++)
   yffe += x[k]*ffe_c[k];

ac_complex<ac_fixed<DFE_W+1,1> > ydfe = 0;
// decision feedback equalizer, 16 iterations
dfe: for(int k=0; k < ndfe; k++)
   ydfe += SV[k]*dfe_c[k];

// equalizer output
ac_complex<ac_fixed<FFE_W+1,1> > y;
y = yffe - ydfe;

// 64-QAM slicer
qam_slicer(y, r, i);

SV[0] = ac_complex<ac_fixed<3,0> >(r,
// compute error
e = SV[0] - y;

// convert constellation point into 6
data = r*64 + i*8;

// Sign-LMS Adaptation for FFE, 8 ite
ffe adapt: for (int k=0; k < nffe; k+
```

Transcript

0 Errors | 110 Warnings | 357 Infos | 495 Comments

```
#  Message
#  Checking results
#  'data'
#  capture count        = 128
#  comparison count     = 128
#  ignore count         = 0
#  error count          = 0
#  stuck in dut fifo    = 0
#  stuck in golden fifo = 0
#
#
   Simulation PASSED @ 7688500 ps
   (vsim-6574) SystemC simulation stopped by user.
#  > end dofile {.\verif.tcl}
Solution_1{2}>
Ready
```

Catapult integrates a fully automated verification flow, including linting and code coverage of the input source code and push-button simulation of the generated RTL, for higher quality results.

High-Level Synthesis from ANSI C++ and SystemC

Catapult offers support both for pure untimed ANSI C++ and for SystemC, the two major standard languages for high-level design and synthesis. This dual-language support is ideal for engineers, letting them choose the language most suited to their design needs or company culture, including using both languages in a single design flow. The untimed nature of C++ makes it the best choice for architectural design and verification at the most abstract level, and SystemC is effective for the finer control desired for synthesis of complex control-logic, such as bus interfaces and SoC interconnects. Moreover, by exclusively relying on standards, Catapult allows companies to leverage their legacy designs regardless of whether they are in ANSI C++ or SystemC.

Full-Chip High-Level Synthesis

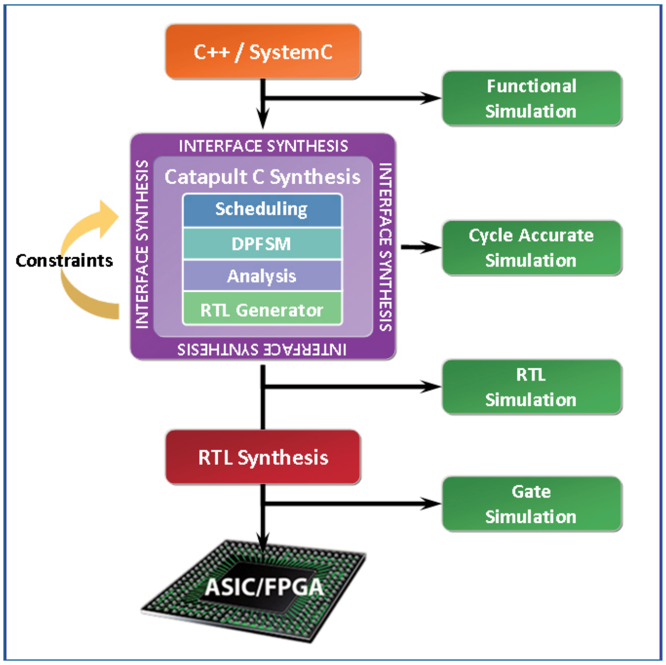
Catapult C Synthesis synthesizes multi-block pipelined and concurrent hierarchical designs from pure sequential ANSI C++ and from SystemC. When beginning a new design, Catapult C Synthesis identifies and analyzes possible hierarchies, allowing the designer to interactively select the optimal structure. Catapult then uses its hierarchical engine to synthesize each function to concurrent hierarchical blocks with autonomous FSMs, control logic, and datapaths. Simultaneously, the tool's robust channel synthesis capability optimizes inter-block communication, supporting streamed channels with FIFOs, ping-pong memories, shared memories, channel depth, and channel width. Catapult also performs top-level pipelining, automatically building concurrent and pipelined sub-blocks to satisfy top-level throughput constraints.

Front-to-Back, Fully Automated Verification

Catapult leverages Mentor Graphics' leadership in functional verification and integrates a robust and fully automated verification flow, helping designers confidently synthesize their designs and easily validate the correctness of the generated RTL.

Starting with the input model, Catapult automatically performs linting and static code checks to catch potential errors and improve the model. Catapult also provides code coverage reports and performs runtime checks, helping designers achieve a higher degree of confidence in their C++ and SystemC code.

After the RTL has been synthesized, Catapult automates a complete verification infrastructure reusing the original C++ or SystemC testbench to exercise the generated RTL. Leveraging automatically-built transactors and comparators, the RTL outputs are matched against the golden results. This flow can replay the cycle-accurate behavior of the RTL back into the original design, so designers can analyze timing



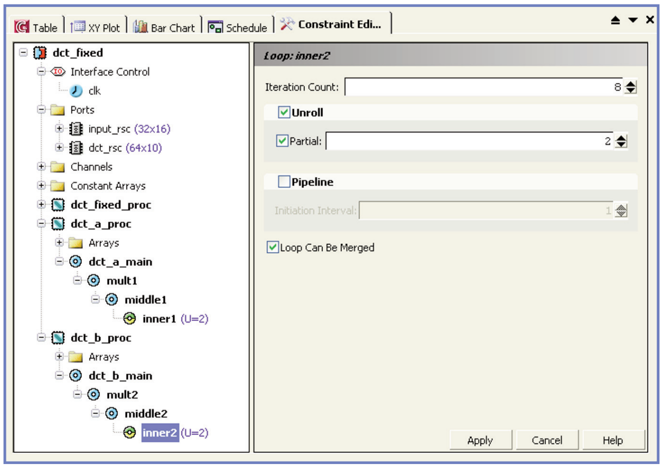
The Catapult design flow enables the evaluation of a wider range of micro-architectures than possible using hand-coded methods.

related aspects directly in the untimed source. This push-button solution automatically produces all the needed files and scripts and offers an effortless solution to verify the generated design.

Micro-Architecture Analysis and Optimization

Catapult combines automation with specific high-level constraints so designers can precisely control the hardware implementation and interactively converge on significantly better quality designs in less time. The architectural constraints editor presents a graphic view of all ports, arrays, and loops in the design and allows any or all of the following high-level constraints to be applied:

- Loop merging, unrolling, and pipelining
- Relative cycle-by-cycle timing
- RAM, ROM, or FIFO array mapping
- Memory resources merging
- Memory width re-sizing



```
dct_fixed
    Interface Control
        clk
    Ports
        input_rsc (32x16)
        dct_rsc (64x10)
    Channels
    Constant Arrays
    dct_fixed_proc
    dct_a_proc
        Arrays
        dct_a_main
            mult1
                middle1
                    inner1 (U=2)
    dct_b_proc
        Arrays
        dct_b_main
            mult2
                middle2
                    inner2 (U=2)
```

Loop: inner2

Iteration Count: 8

☑ Unroll

☑ Partial: 2

□ Pipeline

Initiation Interval: 1

☑ Loop Can Be Merged

Optimize hardware by applying architectural constraints to unroll, merge, and pipeline loops, map arrays to RAM, and control resource allocation.

Interface Synthesis

Interfaces and their properties have a determinant impact on the performance and quality of a design. For that matter, Catapult is fully geared to help designers make the best interface decisions and build them optimally. The tool accommodates the needs of both architects seeking the best timing and bandwidth and designers needing to implement them most efficiently.

At the most abstract level, Catapult does not require interface protocols to be embedded in the source description. Rather it accepts a pure ANSI C++ description as its input and uses patent-pending interface synthesis technology to control the timing and communications protocol on the design interface. This enables interface analysis so designers can explore a full range of hardware interface options such as streaming, single- or dual-port RAM, handshaking, FIFO, and many other custom or built-in I/O components.

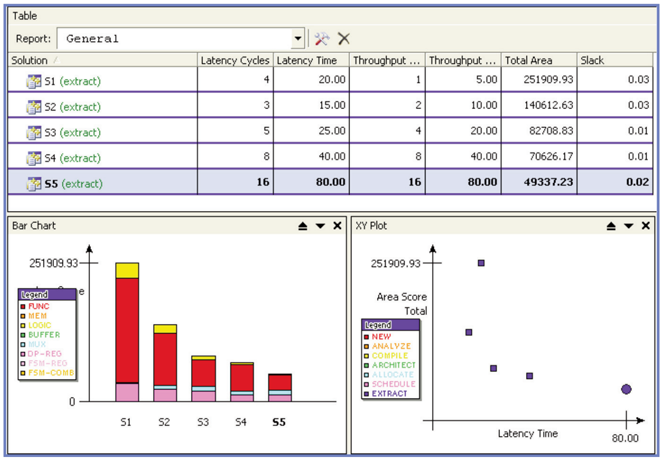
Alternatively, Catapult lets designers fully specify their interface requirements using cycle-accurate descriptions in SystemC. Using a transaction-level modeling style, interface timing and behavior is separated from the core functionality. This enables designers to tightly control the implementation of the design interfaces and still benefit from abstract modeling for the rest of the design. This approach is particularly well suited to efficiently model, verify, and synthesize complex bus interfaces and SoC interconnects.

Low-Power Exploration and Optimization

Catapult C Synthesis fully automates highly efficient, low-power design techniques; such as multi-level clock gating, memory access optimization, intelligent resource sharing, and multiple clock domains. The tool thoroughly analyzes and optimizes RTL netlists to reduce power consumption, working on a per-register basis to maximize power savings. These powerful optimizations can be used in combination with Catapult's power exploration flow, letting designers instantly generate dozens of implementation candidates based on design parameters such as clock frequency, performance, and micro-architecture. This unique combination of power exploration and optimization delivers unrivalled results, dramatically lowering the power meter.

Predictable Timing Closure

Catapult features technology-aware scheduling and allocation heuristics to produce superior designs and predictable timing closure in the physical design stage. Catapult C Library Builder collects detailed characterization data from the downstream RTL synthesis tools with specific target technology libraries. This allows Catapult to precisely schedule hardware resources, chain operators, infer multi-cycle components, and quickly provide accurate area, latency, and throughput estimates without spending costly time and effort going through RTL synthesis. The tool maximizes design performance by structuring essential FSM control logic off the timing-critical datapath. Highly optimized datapaths are constructed by leveraging the native technology-specific operators used by the downstream RTL synthesis tools, such as DesignWare for Design Compiler. This methodology ensures precise knowledge of datapath delays, leading to correct-by-construction timing through RTL and physical synthesis.



Table

Report: General

Solution	Latency Cycles	Latency Time	Throughput ...	Throughput ...	Total Area	Slack
S1 (extract)	4	20.00	1	5.00	251909.93	0.03
S2 (extract)	3	15.00	2	10.00	140612.63	0.03
S3 (extract)	5	25.00	4	20.00	82708.83	0.01
S4 (extract)	8	40.00	8	40.00	70626.17	0.01
S5 (extract)	**16**	**80.00**	**16**	**80.00**	**49337.23**	**0.02**

Catapult offers a comprehensive set of analysis tools to debug, explore, optimize, and rapidly converge on the optimal solution in terms of power, performance, and area.

Interactive Design Analysis Tools

Automating RTL creation with Catapult allows designers to easily explore a wide range of alternative micro-architectures for a given design. Catapult offers superior control, generating solutions based on user constraints and graphically displaying the results in a choice of X-Y plots, bar charts, tables, and schematic views. Designers quickly make informed decisions in terms of power, area, and performance to deliver the optimal balance of these features. The hierarchical Gantt chart in Catapult provides information on critical paths, data flow, and component utilization. It gives designers immediate insight into hard-coded performance bottlenecks and inefficiencies; such as memory bandwidth limitations, loop dependencies that prevent parallelism, and data dependencies that limit optimal scheduling. Designers can quickly identify problem areas and cross probe back to the C++ code to understand and optimize both the source and hardware implementation.

Integrated ESL to RTL Flow

Leveraging SystemC and transaction-level modeling (TLM), Catapult unites two distinct domains — system-level and hardware design. When combined with Mentor Graphics Questa and Vista solutions, it lays the foundation for next-generation electronic system level (ESL) design. Catapult offers natural connections with ESL flows and practices either by synthesizing from TLM written in SystemC or by automatically generating TLM from a pure C++ model. Starting from the Mentor Graphics Vista SystemC modeling environment, designers can seamlessly transition to the Catapult environment, where they can explore different micro-architecture options and use the tightly integrated Vista debugger to fine tune the results. Once optimized and verified, the high-level models are synthesized to RTL. From transaction to cycle-accurate to RTL, Catapult provides support for all the models and abstraction levels needed to bridge the gap between ESL and implementation.

Certified and Integrated Flows

Catapult produces certified integration with third-party tools offering the smoothest flow and the highest quality of results in going from ESL to verified gates. Built-in flows and officially supported tools include, but are not limited to:

- Mentor Graphics:
  Vista, ModelSim, Questa, Olympus, Precision
- Synopsys: DesignCompiler, VCS
- Magma: Talus
- Cadence: RTL Compiler, NC-Sim
- Mathworks: Simulink, MATLAB
- Atrenta: SpyGlass, SpyGlass-Power
- Apache: PowerTheater
- Calypto: SLEC
- Xilinx: XPower
- Altera: PowerPlay

Platforms Supported

Windows NT/2000/XP, Linux Red Hat Enterprise,
and SUN Solaris 8

CALYPTO®

Visit our web site at www.calypto.com for the latest product news.