Company Introduction

- Mission: SoC nanometer timing and synthesis technology leader
- Channels

Direct Offices:
USA: Silicon Valley
Southern California
Taiwan: Hsinchu

Distributors:
Japan (Marubeni), India (ICON),
China (OnePass Solutions),
Korea (ED&C), Israel (AST)
Incentia Product Offering

◆ Logic, Low Power, DFT Synthesis solution
  ■ DesignCraft, PowerCraft, TestCraft

◆ Timing analysis solution
  ■ TimeCraft: World fastest Static Timing Analyzer (STA)
  ■ TimeCraft-LOCV: Location Based OCV
  ■ TimeCraft-SSTA: Statistical STA
  ■ TimeCraft-SI: Signal Integrity
  ■ TimeCraft-PCA: Power Analysis
  ■ ConstraintCraft: Constraint Management

◆ Design closure solution
  ■ ECOCraft-Timing: Hold-time & Setup-time ECO
  ■ ECOCraft-Power: Leakage power ECO
How Incentia Products Fit into Design Flow

**IC Design Flow**

- **Logic Synthesis**
  - Netlist
- **Placement & Route**
  - Netlist, DEF/GDSII
- **RC Extraction**
  - Netlist, SPEF
- **Delay Calculation**
  - Timing, Power Analysis
- **Meet Constraints?**
  - Yes: **OK**
  - No: **Hold-Time, Power ECO**

**Incentia Products**

- **DesignCraft**
  - (Logic, DFT, Low Power Synthesis)
- **Complete Timing Analysis**
  - TimeCraft (Static Timing Analysis)
  - TimeCraft–LOCV
  - TimeCraft-SI
  - TimeCraft–SSTA
  - TimeCraft-PCA (Power Analysis)
  - ConstraintCraft (Constraint Management)

**Post-layout Design Closure**

- ECOCraft-Timing (Hold-time & Setup-time ECO)
- ECOCraft-Power (Leakage Power ECO)
DesignCraft Highlights

◆ Complete solution
  ■ Logic, low power, DFT
  ■ Fast synthesis runtime: Up to 3X faster than other solutions!
  ■ Specialized for aggressive area & power reduction
◆ Support all the standard inputs
  ■ Verilog/VHDL
  ■ Synopsys .lib, CCS library
  ■ Timing constraints in SDC
  ■ Tcl scripts
◆ Easy to migrate and use!
  ■ Take your RTL code directly without any modifications!
  ■ Very similar script usage
◆ **Much more affordable solution with better results!**
◆ **Proven through many customer tape-outs**
Complete Integrated Logic Synthesis

DesignCraft

Built-in STA Unified database CREST

Data Path

Low Power

Leakage Power

DFT

Target for designs demanding aggressive area & power reduction
Contents

◆ Data-path synthesis
◆ Optimization strategy
◆ Low power synthesis
  ■ Multi-voltage designs
  ■ Dynamic power optimization
  ■ Leakage power optimization
◆ Power analysis
◆ DFT synthesis
Data-path Synthesis

Data-path (e.g. adder, multiplier, …) implementation architectures affect synthesis area / timing QOR

- Various architectures are available for area/timing tradeoff

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rpl</td>
<td>Ripple carry</td>
</tr>
<tr>
<td>bk</td>
<td>Brent-Kung carry look-ahead</td>
</tr>
<tr>
<td>fbk</td>
<td>Fast Brent-Kung carry look-ahead</td>
</tr>
</tbody>
</table>

- Data-path inference in RTL code
  - Automatic inference of data-path & architecture
  - Explicit instantiation through Incentia Parameterized Components

- Data-path resource sharing
  - Control through compiler derivatives
Incentia Parameterized Components (IPC)

◆ Support more than 40 pre-defined data-paths through IPC
  ■ add, sub, addsub, mult, divider, inc, dec
  ■ absval, ash, bsh, shifter, decoder
  ■ mac, prod_sum, square, sqrt, …
  ■ Also pipe-lined IPCs

◆ Data-path instantiations in RTL code
  ■ Using IPC notations
  ■ Can also take your existing synthesizable RTL code directly without any modifications!
Data-path Synthesis: Inference Example

◆ Automatic inference from RTL code
  ■ Automatic architecture selection and optimization

◆ Explicit instantiation
  ■ Instantiate pre-defined data-paths (adder, multiplier, etc)
  ■ Optionally specify the implementation architecture through Verilog/VHDL compiler directives
  ■ Also take DW*** from your existing RTL code


code:

```verilog
module top (a, b, ci, sum, co);
    parameter width = 16;
    input [width-1:0] a, b;
    input ci;
    output [width-1:0] sum;
    output co;

    // incentia ipc_implementation fbk I1
    ipc_add #(width) I1 (.A(a), .B(b), .CI(ci), .SUM(sum), .CO(co));
endmodule
```

```
module top (a, b, ci, sum, co);
    input [15:0] a, b;
    input ci;
    output {15:0] sum;
    output co;
    assign {co, sum} = a + b + ci;
endmodule
```

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rpl</td>
<td>Ripple carry architecture</td>
</tr>
<tr>
<td>bk</td>
<td>Brent-Kung carry look-ahead architecture</td>
</tr>
<tr>
<td>fbk</td>
<td>Fast Brent-Kung carry look-ahead architecture</td>
</tr>
</tbody>
</table>
Optimization Strategy

◆ Synthesis and optimization are done based upon constraints
  ■ Finding the best trade-off point for timing, area, and power

  optimize ?-mode mode_type? ?-incremental?
  ?{ -only_design_rule | -no_design_rule }?
  ?-only_fix_hold? ?-expand expand_type?
  ?-structure_effort structure_effort_type?
  ?-map_effort map_effort_type?
  ?-demote_timing? ?-scan? ?-boundary_opt?
  ?-leakage_power?

◆ Options to explore for best timing, area or power, and runtime
  ■ mode: timing vs. area optimization mode
  ■ structure_effort: boolean optimization for area mode
  ■ map_effort:
    ■ demote_timing: another option for
    ■ boundary_opt: allows for boundary optimization for better quality
Low Power Solution

- **DesignCraft**
  - Low Power Synthesis
  - Multi-vdd synthesis
  - Dynamic power optimization
    - Clock gating
    - Sentinel data-path
  - Leakage power optimization
    - Multi-thresholds
  - Power analysis
    - Dynamic power
    - Leakage power

- **Switching Activity**
  - (VCD, SAIF, FSDB)

- **Reports**
  - (VHDL, Verilog)

- **Netlist**
  - (SDF, SDC)

- **Constraints**
  - (SDC)

- **Libraries**
  - (LIBs)

- **Multi-Vth Libraries**
  - (LIBs) (optional)
Multi-vdd Synthesis

♦ Support of both top-down and bottom-up approaches
♦ Automatic level shifter insertion among different voltage domains
♦ Seamless integration with DFT
♦ Concurrent multi-vdd and multi-threshold optimization
♦ CPF support
♦ Support multi-bit sequential cells for area/power reduction
♦ Efficient low power cell (level-shifter, isolation/retention cell) insertion in early or integration stage
Automatic Level Shifter Insertion

(set target_lib {lib10})

(set design_target_lib {
  { voltage_island_1 lib12 } |
  { voltage_island_2 lib10 } |
  { voltage_island_3 lib08 } } )
Concurrent Multi-vdd Multi-threshold

set target_lib { lvt_lib10 hvt_lib10 }
set design_target_lib {
  { voltage_island_1 { lvt_lib12 hvt_lib12 } }
  { voltage_island_2 { lvt_lib10 hvt_lib10 } }
  { voltage_island_3 { lvt_lib08 hvt_lib08 } }
}

(1.2v) Voltage Island 1
LVT / HVT

(1.0v) Voltage Island 2
LVT / HVT

(0.8v) Voltage Island 3
LVT / HVT
Dynamic Power Optimization

 entra

 Clock gating

 - Do not depend on specific RTL coding styles: highest achievable rate for clock gating implementation

```
always @ (negedge clk)
  if (select)
    out <= in;
```

- Unique latch-based clock gating optimization: take full advantage of time borrowing to meet timing and reduce area
- Able to use integrated clock gating cells in the library
Dynamic Power Optimization (Cont)

◆ Sentinel data path for sleep mode operation
  ■ Generate guarding sentinel logic for data-paths
  ■ RTL compiler directive controls

```
a b
data_in select
```

![Diagram](attachment:image.png)
Dynamic Power Optimization: Customer Cases

- Clock gating can reduce about 30% dynamic power

<table>
<thead>
<tr>
<th>Designs</th>
<th>Power Consumption (mW)</th>
<th>Power Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No Clock Gating</td>
<td>Clock Gating</td>
</tr>
<tr>
<td>Design 1</td>
<td>330</td>
<td>220</td>
</tr>
<tr>
<td>Design 2</td>
<td>638</td>
<td>447</td>
</tr>
<tr>
<td>Design 3</td>
<td>736</td>
<td>489</td>
</tr>
<tr>
<td>Design 4</td>
<td>1120</td>
<td>765</td>
</tr>
</tbody>
</table>

Clock gating can reduce about 30% dynamic power.
Leakage Power Optimization

◆ Global leakage power optimization
  ■ Critical paths: low-vt cells
  ■ Non-critical paths: high-vt cells

◆ Flexible controls of optimization priority
  ■ Timing first: reduce leakage power while meeting timing constraints
  ■ Leakage power first: reduce leakage power as much as possible while keeping timing results in reasonable range

◆ End results
  ■ An average of 30% leakage power reduction
  ■ Up to 80% leakage power reduction using aggressive leakage power optimization
## Leakage Power Optimization

<table>
<thead>
<tr>
<th></th>
<th>Leakage Power Optimization Effort</th>
<th>Leakage Power Reduction</th>
<th>Timing (WNS)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Design1</strong> (0.13nm, 2M gates)</td>
<td>None</td>
<td></td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>30.8%</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Aggressive</td>
<td>68%</td>
<td>-0.02</td>
</tr>
<tr>
<td><strong>Design2</strong> (90nm, 4M gates)</td>
<td>None</td>
<td></td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>32.3%</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Aggressive</td>
<td>76.4%</td>
<td>-0.08</td>
</tr>
</tbody>
</table>
Power Analysis Highlights

◆ Inputs
  ■ Netlist or RTL
  ■ Activity file formats: VCD, SAIF, FSDB
  ■ Timing data: SPEF, annotated cap, wire-load model

◆ Reports
  ■ Dynamic power: Switching, Internal (SDPD), Glitch, X-transition power
  ■ Static/Leakage power (SDPD)
  ■ Average, peak, clock tree, interval power

◆ Peak power output for waveform viewer

◆ Very fast runtime: 2X to 5X faster than other solutions!
Power Analysis: Switching Activities

- Vectorless power analysis capability
  - No need of input switching activity
- Gate-level power analysis with RTL-input activity file
  - Skip long gate-level simulation for large gate-level activity file
- Analyze sub-design power with top-level design activity files

```
compile_fsdb --current_design current_subckt --strip_path xxxx
```

- Multiple activity files for each sub-design

```
compile_fsdb --multi_activity_file { { top/I1 I1.fsdb U1} { top/I1 out2.dump U2} }
```

- Multiple activity files for different time slots

```
compile_fsdb --strip_path top/I1 "file_1.fsdb file_2.fsdb file_3.fsdb"
```
Power Analysis: Reports

- Dynamic, static (leakage)
- Average, peak, clock tree
- Interval power
  - Report average power for interested intervals

**Single interval:**
```
set_pa_option –interested_interval “100 200”
compile_fsdb or compile_vcd
report_analyzed_power –interested_interval “100 200” > power_1.rpt
```

**Multiple intervals:**
```
set_pa_option –interested_interval “100 200 300 400 500 600 700 800”
compile_fsdb or compile_vcd
report_analyzed_power –interested_interval “100 200” > power_1.rpt
report_analyzed_power –interested_interval “300 400” > power_2.rpt
report_analyzed_power –interested_interval “500 600” > power_3.rpt
report_analyzed_power –interested_interval “700 800” > power_4.rpt
```

- Toggle rate report
Power Distribution Diagram

Module: sub1

- Switching (% Dyn.) : $1.624 \times 10^{-3}$ (50.70%)
- Internal (% Dyn.) : $1.579 \times 10^{-3}$ (49.30%)
- X-trans (% Dyn.) : $0.000 \times 10^{0}$ (0.00%)
- Glitch (% Dyn.) : $0.000 \times 10^{0}$ (0.00%)
- Dynamic (% Tot.) : $3.203 \times 10^{-3}$ (99.98%)
- Static (% Tot.) : $5.231 \times 10^{-7}$ (0.02%)
- Peak Power (Time) : $3.204 \times 10^{-3}$ (20345 ~ 20355)
- Total : $3.204 \times 10^{-3}$
Time-Base Power Waveform in nWave

- Output peak power info in VCD or FSDB format
## DesignCraft Benchmark Data

### Design Specifications

<table>
<thead>
<tr>
<th>Design</th>
<th>Logic Gates</th>
<th>Macros</th>
<th>Frequency</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design1</td>
<td>1.2M</td>
<td>8</td>
<td>133 MHz</td>
<td>.130nm</td>
</tr>
<tr>
<td>Design2</td>
<td>2.2M</td>
<td>12</td>
<td>250 MHz</td>
<td>90nm</td>
</tr>
<tr>
<td>Design3</td>
<td>3.6M</td>
<td>24</td>
<td>250 MHz</td>
<td>90nm</td>
</tr>
<tr>
<td>Design4</td>
<td>5.6M</td>
<td>22</td>
<td>350 MHz</td>
<td>65nm</td>
</tr>
</tbody>
</table>

### Normalized Runtime

![Normalized Runtime Graph]

- **DesignCraft**
  - Design1: 1.4X
  - Design2: 2.2X
  - Design3: 1.7X
  - Design4: 2.5X
Summary

◆ Complete integrated synthesis solution
  ■ Logic, Low Power
◆ Very fast runtime with big capacity
  ■ Up to 3X faster than other solutions!
◆ Most aggressive reduction in chip area and power consumption
  ■ Smaller synthesized gate counts: up to 20% less
  ■ Smaller power: up to 20% less
◆ Many customer tape-outs in different applications
  ■ Communication, mixed-signal, networking, wireless, consumer electronics, multi-media, graphics
  ■ From 250nm to 65nm
◆ Best performance-price synthesis solution in market!