What Is Your Innovation Platform?

Building Innovation Platforms For Consumer Electronic Product Development

Sankhya Technologies Private Limited

www.sankhya.com
Agenda

- CE Overview
- Convergence and Market Opportunity
- Imperatives for Success
- Inside a CE Device
- Development Challenges
- System Level Design
- CE with System Level
- Design Innovation Platforms for CE
- A Quick Example
Consumer Electronics

Embedded System;
A computer embedded into a larger system or device
Consume Electronic products are examples of embedded Systems
Opportunity

- Convergence of Functions Leading To Smarter Phones, Set Top Boxes, Mobile Internet Devices etc.
- Near Perpetual Demand for Smarter Devices
- Smart devices enable mobility, work and deliver news, entertainment and education
- Opportunity!
  - Over $150 Billion market in India alone by 2020
  - Indian Cable TV will go digital by 2014
  - GoI promoting Electronic System Design and Manufacturing (New Policy for 2012)
### Long Term Growth

- Companies in the space show consistent growth

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM</td>
<td>7.68 Millions of Pounds</td>
<td>100</td>
<td>232.4</td>
<td>406.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broadcom</td>
<td>16.33 Millions of USD</td>
<td>1100</td>
<td>2670</td>
<td>6818</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Texas Instruments</td>
<td>34.6 Millions of USD</td>
<td>9200</td>
<td>13392</td>
<td>13966</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Samsung</td>
<td>137</td>
<td>8940</td>
<td>17210</td>
<td>27834</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apple</td>
<td>365.83 Millions of USD</td>
<td>7980</td>
<td>13930</td>
<td>65220</td>
<td>108250</td>
<td></td>
</tr>
</tbody>
</table>
Imperatives for Success

- **Consistently** Deliver High Quality Effective **Timely** Solutions To Customers and Customer Groups Competitively

- => Build Differentiators => Improve Productivity

- Differentiate => Faster => Better => Cheaper
  - More Functions AND Intuitive Easy To Use
  - Lower Power (Last Longer on Battery Charge)
  - Globally Optimal Designs = Smaller chips -> Faster, Higher Yield

Improve Productivity =>

- Large Block Reuse Across Projects and Products
- Faster Design Cycles Through Automated Solution Development
- Collaborative Agile Development
  - => Create Innovation Platforms
How Do I Build An Innovation Platform?
Inside a CE Device

A Complex Embedded System

- **Processors**
  - RISC, CISC, DSP, VLIW, ASIP

- **Peripherals**
  - Memory, Serial, USB, WiFi, Buses

- **HW Accelerators**
  - H.264, JPEG2000, MP3, MP4

- **Software**
  - Kernel, Drivers, Middleware, Applications

**User Interface**
Development Activities

- Solution Specialist: Market Research and Product Specification
- System Architect: Product Prototyping, UI and UE and Analysis
- Processor and Tools Architect: Processor Modeling and Software Development Tools Like Compilers, Assemblers, Linkers, Debuggers, Simulators
- Peripheral Designer: Peripheral Modeling USB, Ethernet, Bluetooth, CAN, RFID, GPS, GSM, Sensors – Behavior, Hardware
- System Software Architect: OS and Driver Development / Porting; Linux, Android Etc.
- Media Architect: DSP, Audio/Video Codecs
- Hardware/Prototype Designer: FPGAs/ASICs
- Application Architect: UML, Developing UI, MVC, Web
- Circuits and Circuit Board Design: PCB Design, DFM
- Physical Design: “Crash Testing” …
## Way To Innovation Platform

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Benefit to Designer</th>
<th>Translating to Better Design</th>
<th>Competitive Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Level Design</td>
<td>Use Architectural Languages for Design</td>
<td>Globally Optimized Solution</td>
<td>Low Cost, High Performance</td>
</tr>
<tr>
<td>Architectural Approach</td>
<td>MultiDimensional Design Automation</td>
<td>Comprehensive Solution</td>
<td>High Quality Products, Reduced Support Costs</td>
</tr>
<tr>
<td>Globally Optimized</td>
<td>Integrated Processor and System Design, Successful First Design</td>
<td>Reduced Gate Counts Direct Connection Architectures</td>
<td>Reduced Power Usage, Enhanced Yields (Lower Manufacturing Costs)</td>
</tr>
<tr>
<td>Higher Abstraction Designs</td>
<td>Greater Design Reuse and Automation</td>
<td>Improved Quality</td>
<td>Reduced Time to Market, Quick to develop Product Variants for Global Markets</td>
</tr>
</tbody>
</table>
Application Driven Embedded System Level Design

STB, DVD Player, Camera, MP3

System Models

Processor, Peripherals Models

Technology

Idea

Market

Solution

ESL D-Flow: Specify => Model => Verify => Synthesize
Application Driven Embedded System Level Design
For Consumer Electronic Product Development
Specify => Model => Integrate
Virtualize => Verify => Synthesize

Faster to market translates to higher price points and greater market share.

Transparent Modeling Languages for Processor and System Design

Automatic component reuse spanning application, system, peripherals and processors

Reusable HW/SW Developed In Parallel Using Virtual Prototypes
Reduced TIME TO MARKET

Model Driven Tools For Managing Design Tradeoffs

3-9 Months!
## Benefits for CE Development

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Benefit to Designer</th>
<th>Translating to Better Design</th>
<th>Competitive Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Level Design</td>
<td>Use Architectural Languages for Design</td>
<td>Globally Optimized Solution</td>
<td>Low Cost, High Performance</td>
</tr>
<tr>
<td>Architectural Approach</td>
<td>MultiDimensional Design Automation</td>
<td>Comprehensive Solution</td>
<td>High Quality Products, Reduced Support Costs</td>
</tr>
<tr>
<td>Globally Optimized</td>
<td>Integrated Processor and System Design,</td>
<td>Reduced Gate Counts Direct Connection</td>
<td>Reduced Power Usage, Enhanced Yields (Lower Manufacturing Costs)</td>
</tr>
<tr>
<td></td>
<td>Successful First Design</td>
<td>Architectures</td>
<td></td>
</tr>
<tr>
<td>Higher Abstraction Designs</td>
<td>Greater Design Reuse and Automation</td>
<td>Improved Quality</td>
<td>Reduced Time to Market, Quick to develop Product Variants for Global Markets</td>
</tr>
</tbody>
</table>

Embedded System Level Design

Principles of System level Design
Design to Innovation

**Design Platform**
- Formal Architecture Driven
- Cross-Functional Automation
- Publish/Subscribe “Channel” Model To Share Changes, Concurrent Development
- Virtual Prototypes
- High Level Synthesis
- Globally Optimized Green Solutions “Low Power, High Performance”

**Innovation Platform**
- Focus Creative Forces
- Target Multiple Segments
- Facilitate a Business and Technology Play
- Enable Reuse Across Multiple Products and Projects
- Unity of Purpose
System Model – A Quick Example
System Modeling with SSDL

Component Types

- Memory Device Model
- CPU Model
- USART Model

Component Types

- Memory
- Processors
- USART

Components

Relationship

Read/Write

Set/Get
model processor DLX dlx.md
model device SRAM SRAM.dll
model device USART USART.dll

component DLX cpu1
component DLX cpu2
component SRAM ram size=0x2000 mode=rw
component USART usart

map memory cpu1:address.0x0 ram:STDRAM
map memory cpu1:address.0xfffff0000 usart:tx_register
Questions?

➢ Innovation is the specific instrument of entrepreneurship... the act that endows resources with a new capacity to create wealth. – Peter Drucker

E-mail gopi@sankhya.com
http://www.sankhya.com/contact.html
http://www.sankhya.com/info/products/teraptor/teraptor.html
• Specify several views of the processor like, Architectural view, Mnemonic View and Machine code

Instruction Set architecture
- Describes the Instruction Set Architecture of the target processor in, Behavioural, Mnemonic and machine code formats.

Assembler Interface
- Describes the mapping between external assembler format and the internal format used in the processor model.

Binary Interface
- Describes the Application Binary Interface. This includes information like stack layout and parameter passing convention.

Relocation Interface
- Describes the relocations supported by the target architecture.
SMDL Example – Operands

• Immediate Operand

// 5-bit unsigned immediate value
Operand o_uimm5 : Integer {
    range (i = 0, 31, 1) {
        cbcode = { "$i" };
        asm = { "$i" };
        mcode = { "$i" };
    }
};

• Register Operand

// 32 general purpose registers + 3 special purpose registers
Operand o_gprv : Register {
    range (i = 0, 34, 1) {
        cbcode = { "r$i" };
        asm = {"r$i"};
        mcode = {"$i"};
    }
};
SMDL Example – Addressing Modes

- E.g., PC Relative Offset

Operand o_i_pcrel {
    o_imm16 o;
    cbcode = { "+", "pc", "<<", o, "2" };  
    asm = { expr("<< $o 2") };  
    mcode = { o(0,16) };  
};
Here's an Example of SMDL Representation of a simple 'add' instruction of the DLX processor

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Opcode</th>
<th>Operands</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td>000001</td>
<td>rs1 rs2 rd 00000 00000</td>
</tr>
</tbody>
</table>

**Instruction i_rrr_add**
```plaintext
o_gprv rd, rs, rt;
```

**Common Behavior Code**
```plaintext
cbcode = { "{width=32}="}, rd, "{sign=u}+", rs, rt };  
```

**Mnemonic**
```plaintext
asm = { "add", rd, rs, rt };  
```

**Machine Code**
```plaintext
mcode = { or(0,7,0b00000000); or(7,4,0001); rd(11,5); rt(16,5); rs(21,5);or(26,6,0b000001});  
```
```plaintext
};
```
SMDL Example – Instruction Bundle

• Allows VLIW type instructions to be modeled.
• Example: A 128-bit VLIW

Bundle b_mii {
    i_mem i1;
    i_alu i2, i3;

    b_width = 128;
    cbcode = { i1, i2, i3 };
    asm  = { i1, i2, i3 };
    mcode = { i1(0,48); i2(48,40); i3(88,40) };
};
SMDL Example – Instruction Attributes

• Specify width as 32 bits for this instruction.
  
  \[
  \text{cbcode} = \{ \text{"width=32"}, \text{rd}, \text{"{sign=u}"+}, \text{rs}, \text{rt} \};
  \]

• Cycle Attribute - specifies number of clock-cycles to be used.
  \[
  \{\text{cycle=2}\}{++} \{\text{read-cycle=1,write-cycle=3}\}\text{r1}
  \]

• Effects Attribute – Specifies side-effects of a cbcode operation
  E.g. Subtract with Carry (sbc) instruction for two 32-bit values

  // SBC{cond}{S} Rd, Rn, <Op2>
  Instruction i_rro_sbc {
    o_gprv rn, rd;
    o_operand2 op2;
    cbcode = \{ "="\, \text{rd}, "-"\, ",\{!effects=1,}
              !effects-create-mask=\text{NZCVtttttttttttttttttttttttttttt},
              !effects-destination=\text{flags}\}"\, ",-"\, \text{rn}, \text{op2}, \text{"bf"}, \text{"29"}, \text{"1"},
              \"flags"
    };
    asm = \{ "sbc"\, \text{rd}, \text{rn}, \text{op2} \};
    mcode = \{ \text{or(26, 2, 0b00)}\; \text{or(21, 4, 0b0110)};\text{rn(16, 4)}; \text{rd(12, 4)}; \text{op2(0, 12)} \} ;
  \}

N, Z, C, V, t – Negative, Zero, Carry, Overflow, Transparent (not affected)
flags – Status register
SMDL – Key Features

- The SMDL model contains sufficient information to automate various tasks like CPU design, verification, system and tool development, code generation etc.

- Extensible Language:
  Rich Set of Language attributes enables to Model
  • Cycle-Accurate Processors
  • Complex Pipelining Architectures