Rapid Prototyping Using HDL Coder
Who Are We?

Esa-Matti Turtinen
R&D Manager, SoC Prototyping, Nokia Oulu
- M.Sc., Electrical Engineering
- 31 years old
- About 6 years of experience working on different roles related to SoC development

Joonas Järviluoma
Prototype Engineer, SoC, Nokia Oulu
- M.Sc., Electrical Engineering
- 26 years old
- Just graduated
- Currently working on FPGA lab testing
Expanding the human possibilities of the connected world
Nokia has been at the forefront of every fundamental change in how we communicate and connect.

<table>
<thead>
<tr>
<th>Telephony begins</th>
<th>Analog revolution</th>
<th>Digital revolution</th>
<th>Mobile revolution</th>
<th>The new connectivity through the Cloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bell Telephone Laboratories formed in 1925</td>
<td>Long distance voice communication</td>
<td>Voice, data, and video communication</td>
<td>Wireless communication</td>
<td>Intelligent and seamless connectivity through the Cloud</td>
</tr>
<tr>
<td>Copper networks</td>
<td>Laser</td>
<td>First ever calls on GSM and LTE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Circuit switches</td>
<td>Satellite communications</td>
<td>First car phone</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amplifiers</td>
<td>UNIX</td>
<td>Commercialization of Small Cells</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DWDM</td>
<td>MIMO</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100Gbps optical transport</td>
<td>Optical super channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>400G routers</td>
<td>Optical super channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G.Fast: 1Gbps over copper</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Terabit IP routing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datacenter infrastructure and applications for the Cloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Smart sensors for the Internet of Things</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A financially strong leader

<table>
<thead>
<tr>
<th>Category</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revenue*</td>
<td>€26.6bn</td>
</tr>
<tr>
<td>R&amp;D spend*</td>
<td>€4.5bn</td>
</tr>
<tr>
<td>Net cash*</td>
<td>€10.0bn</td>
</tr>
<tr>
<td>Employees</td>
<td>106,000</td>
</tr>
</tbody>
</table>

* Combined Nokia and Alcatel-Lucent 2015 numbers according to Nokia accounting policies, non-IFRS
R&D professionals ~40,000
Services professionals ~40,000
World leading intellectual property (patent families) ~31,000
Bell Labs
Nokia Technologies
Would it be possible to left-shift this and trial algorithms in HW earlier?
FPGA Prototyping Flow Timeline
Proportional Estimation in Generic HLS flow
HDL Coder Flow
From Algorithm to FPGA Programmable Model
Example Design for HDL Coder Flow
Scaling and Power Limitation Block

- Arithmetic logic (multipliers, adders etc.)
- Loop structures
- State-Machine
- Look-up tables for dB conversions
- Registers for state control and buffering
- Variable indexing
- Configurable parameters
Classic Division of Models
Algorithm and RTL

Algorithm Model

- Large Vectors
- Parallel Operations

Object-Oriented Programming

MATLAB operations optimized for maximized simulation performance

RTL Model

- Hand-written based on algorithm model
- ASIC optimized performance
- Thorough verification required
Division in HDL Coder Workflow
Algorithm and RTL

Algorithm Model:
- Written in MATLAB function blocks/System Objects and Simulink library components
- Has to be written from HW perspective to generate feasible RTL

RTL Model:
- Rapid generation from Simulink (or MATLAB) model
- Verification focus moves towards algorithm
- Cosimulation verificates RTL against algorithm model
- "Is as good as the algorithm"
Example 1: Algorithm without Data Type Definition

```plaintext
if (run)
    mk_tmp = Gk;

% Multiplied branches, format: 1.0.15 * 0.4.14 -> 1.4.29 (output formatting is set automatically by Matlab)
mul_I = data_I*mk_tmp;
mul_Q = data_Q*mk_tmp;
```

```
259 IF run = '1' THEN
260     -- '<S37>:1:44'
261     -- Multiplied branches, format: 1.0.15 * 0.4.14 -> 1.4.29 (output format is set automatically by Matlab)
262     -- '<S37>:1:44'
263     mul_temp := data_I_signed * signed(resize(Gk_unsigned, 19));
264     IF (mul_temp(34) = '0') AND (mul_temp(33) /= '0') THEN
265         mul_I := "011111111111111111111111111111111";
266     ELSIF (mul_temp(34) = '1') AND (mul_temp(33) /= '1') THEN
267         mul_I := "100000000000000000000000000000000";
268     ELSE
269         mul_I := mul_temp(33 DOWTO 0);
270     END IF;
271     -- '<S37>:1:46'
272     mul_temp_0 := data_Q_signed * signed(resize(Gk_unsigned, 19));
273     IF (mul_temp_0(34) = '0') AND (mul_temp_0(33) /= '0') THEN
274         mul_Q := "011111111111111111111111111111111";
275     ELSIF (mul_temp_0(34) = '1') AND (mul_temp_0(33) /= '1') THEN
276         mul_Q := "100000000000000000000000000000000";
277     ELSE
278         mul_Q := mul_temp_0(33 DOWTO 0);
279     END IF;
```
RTL Generation

Example 2: Algorithm with Data Type Definition

```plaintext
if (run)
    mk_tmp = fi(Gk, 1, 19, 14);
%Multiplied branches, format: 1.0.15 * 0.4.14 -> 1.4.29 (output format is set automatically by matlab)
    mul_I = data_I*mk_tmp;
    mul_Q = data_Q*mk_tmp;
```

```
IF run = '1' THEN
    '%<S37>:1:41'
    '%<S37>:1:42'
    mk_tmp := signed(resize(Gk_unsigned, 19));
    --Multiplied branches, format: 1.0.15 * 0.4.14 -> 1.4.29 (output format is set automatically by matlab)
    --'<'S37>:1:44'
    mul_I := data_I_signed * mk_tmp;
    --'<'S37>:1:45'
    mul_Q := data_Q_signed * mk_tmp;
```

Two multipliers
• Original hand-written model, targeted for ASIC, had slightly more signals and routing logic compared to generated model!

• Generated model tested successfully in FPGA-in-the-loop configuration
 ASIC Optimization
Area and Timing Results

Further timing optimization could have been performed (Work focused on FPGA prototyping)
FPGA Prototyping Flow Timeline
Proportional Estimation in HDL Coder Flow
Conclusion
Benefits and Shortages

Benefits:

• Human readable HDL output
• Design work and verification focus moves on higher level
• Good synthesis results in both FPGA and ASIC cases
• Distinct GUI
• Support for 3rd party tools and FPGA boards

Shortages:

• For feasible HDL generation and FPGA prototyping, algorithms have to be written strictly from HW perspective
• No trivial way to generate generic variables to create scalable IPs (due to Model-Based Design flow)
Future Work

Algorithm design work change towards RTL design style required
  • Close co-operation with algorithm and RTL designers is vital
  • Algorithm simulation speed might be critical

IP generation with generic interfaces
  • Was left out of scope in this study
  • Needs to be verified

Projects ongoing
Q & A