NOKIA

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

Nokia has been at the forefront of every fundamental change in how we communicate and connect

Telephony begins

Bell Telephone Laboratories formed in 1925

Analog revolution

Long distance voice communication

- Copper networks
- Circuit switches
- Amplifiers

Digital revolution

Voice, data, and video communication

- Laser
- Satellite communications
- UNIX
- DWDM
- 100Gbps optical transport
- 400G routers

Mobile revolution

Wireless communication

- First ever calls on GSM and LTE
- First car phone
- Commercialization of Small Cells
- MIMO

The new connectivity

Intelligent and seamless connectivity through the Cloud

- 5G
- G.Fast: 1Gbps over copper
- Optical super channels
- Terabit IP routing
- Datacenter infrastructure and applications for the Cloud
- Smart sensors for the Internet of Things

A financially strong leader

Revenue*	R&D spend*	Net cash*	Employees
€26.6bn	€4.5bn	€10.0bn	106,000

* Combined Nokia and Alcatel-Lucent 2015 numbers according to Nokia accounting policies, non-IFRS

NOKIA

R&D professionals

~40,000

Services professionals

~40,000

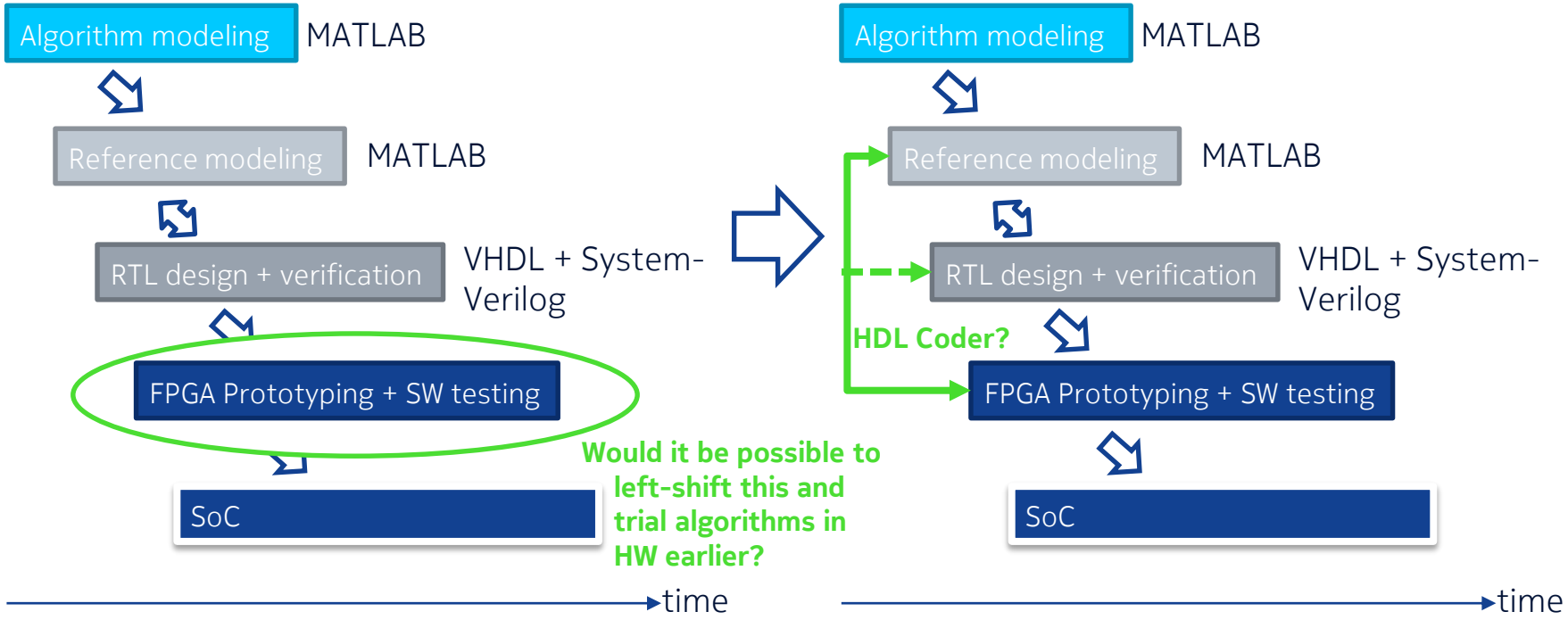
World leading intellectual property (patent families)

~31,000

Bell Labs

Nokia Technologies

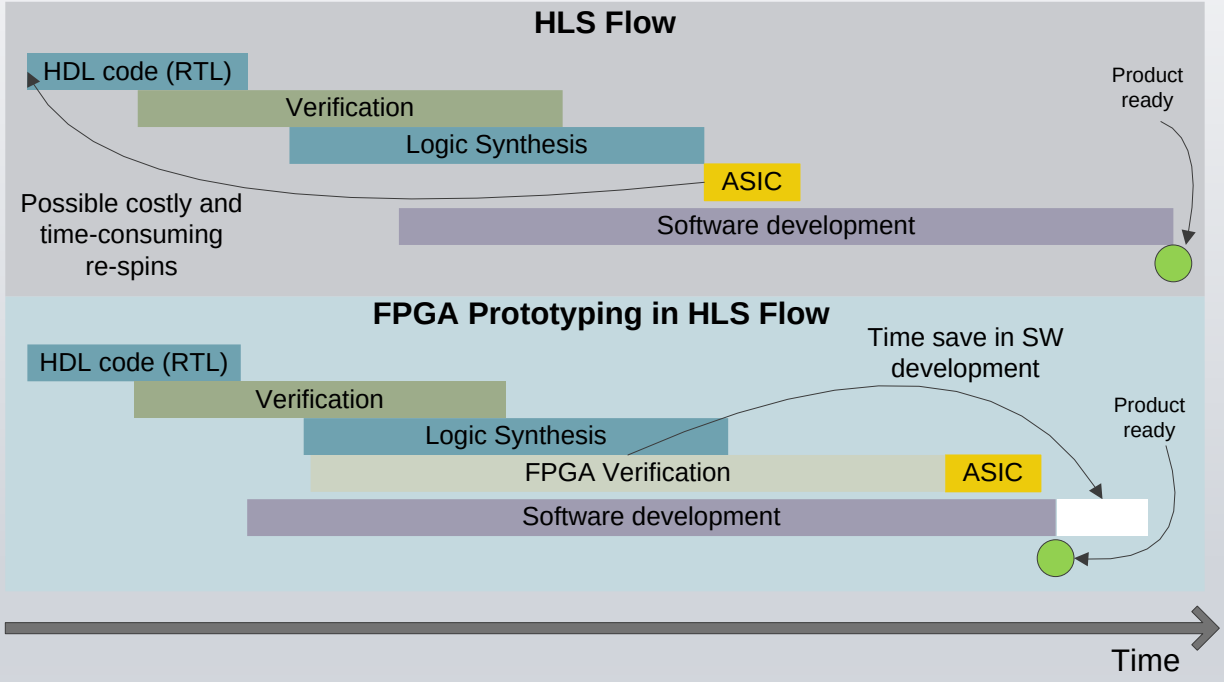
Challenge



NOKIA


NOKIA

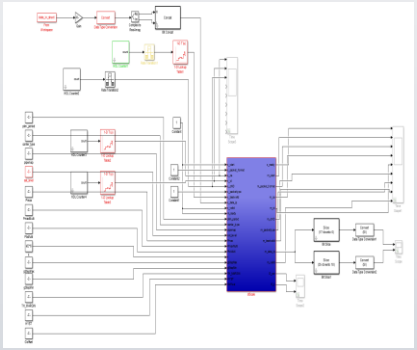
FPGA Prototyping Flow Timeline
Proportional Estimation in Generic HLS flow



HDL Coder Flow

From Algorithm to FPGA Programmable Model

HLS



```
24  %add and saturate, format: 1.5.5 + 1.5.5 -> 1.6.5 -> 1.5.5
25  if (start)
26      tmp_add = fi((G_cor + G_offset(channelID+1)), 1, 11, 5, fm);
27      tmp_sat = tmp_add;
28      valid_reg = true;
29  end
```

MATLAB comments

MATLAB

VHDL

Link to MATLAB design

Line number in MATLAB

```
154     --add and saturate, format: 1.5.5 + 1.5.5 -> 1.6.5 -> 1.5.5
155   IF start = '1' THEN
156   --'<S24>:11'
157   --'<S24>:1:24'
158     add_temp := resize(channelID_unsigned, 13) + to_unsigned(16#0001#, 13);
159     sub_cast := signed(resize(add_temp, 32));
160     add_temp_0 := (resize(G_cor_signed, 13)) + (resize(G_offset(to_integer(sub_cast - 1)), 13));
161     tmp_sat_next <= add_temp_0(10 DOWNTO 0);
162   --'<S24>:1:27'
163   --'<S24>:1:28'
164     valid_reg_next <= '1';
165   END IF;
```

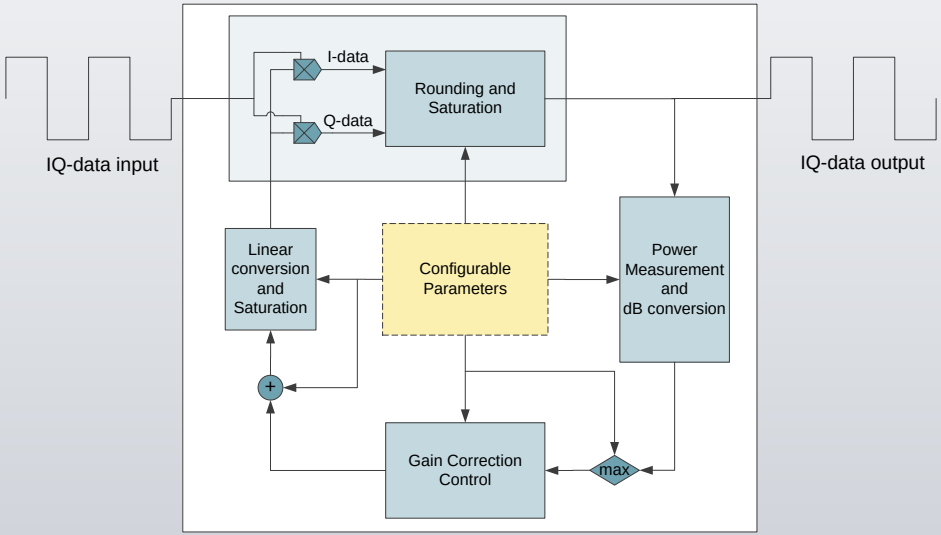
Logic Synthesis

Co-simulation

FPGA-in-the-loop

NOKIA

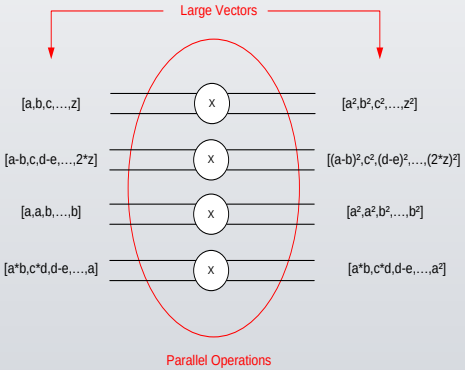
Example Design for HDL Coder Flow
Scaling and Power Limitation Block



- 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

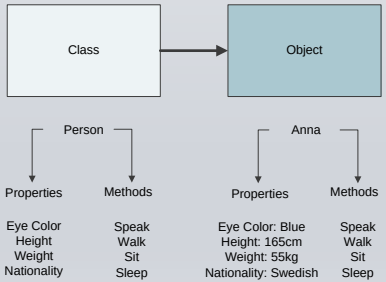
Classic Division of Models
Algorithm and RTL

Algorithm Model



MATLAB operations optimized for maximized simulation performance

Object-Oriented Programming



RTL Model

- Hand-written based on algorithm model
- ASIC optimized performance
- Thorough verification required

NOKIA

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”

NOKIA

RTL Generation

Example 1: Algorithm without Data Type Definition

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

```
259     IF run = '1' THEN
260       --'<S37>:1:41'
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 := "01111111111111111111111111111111111";
266       ELSIF (mul_temp(34) = '1') AND (mul_temp(33) /= '1') THEN
267         mul_I := "10000000000000000000000000000000000";
268       ELSE
269         mul_I := mul_temp(33 DOWNTO 0);
270       END IF;
271       --'<S37>:1:45'
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 := "01111111111111111111111111111111111";
275       ELSIF (mul_temp_0(34) = '1') AND (mul_temp_0(33) /= '1') THEN
276         mul_Q := "10000000000000000000000000000000000";
277       ELSE
278         mul_Q := mul_temp_0(33 DOWNTO 0);
279       END IF;
```

Two multipliers

Two multiplexers

NOKIA

RTL Generation

Example 2: Algorithm with Data Type Definition

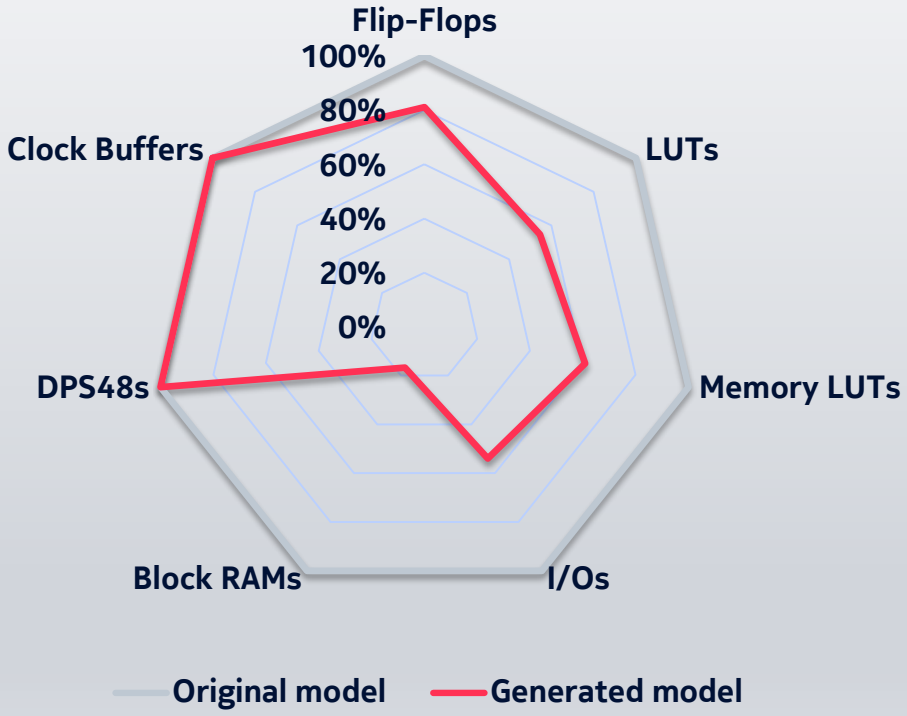
```
41 -  if (run)
42 -      mk_tmp = fi(Gk, 1, 19, 14);
43        %Multiplied branches, format: 1.0.15 * 0.4.14 -> 1.4.29 (output format is set automatically by matlab)
44 -      mul_I = data_I*mk_tmp;
45 -      mul_Q = data_Q*mk_tmp;
46
```

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

Two multipliers

RTL Resource Utilization Comparison

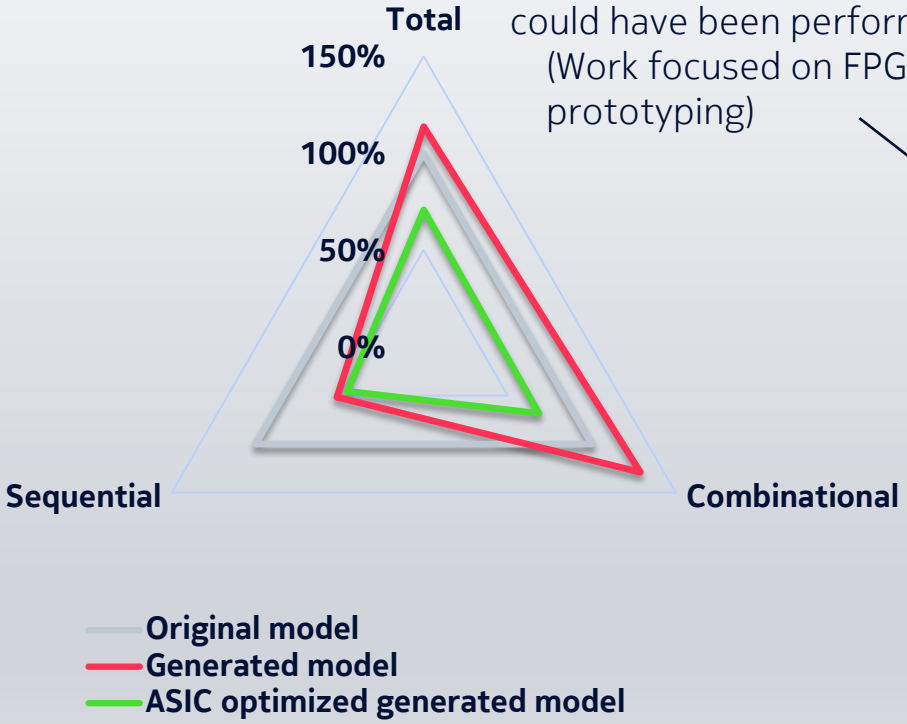
FPGA Prototype Vs. Original ASIC Targeted Model



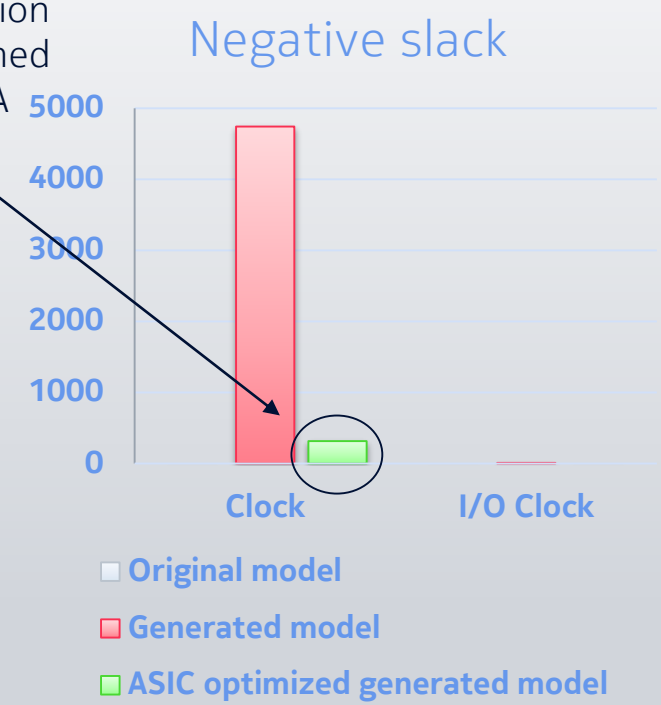
- Original hand-written model, targeted for ASIC, had slightly more signals and routing logic compared to generated model!
- Generated model tested succesfully in FPGA-in-the-loop configuration

NOKIA

ASIC Optimization
Area and Timing Results

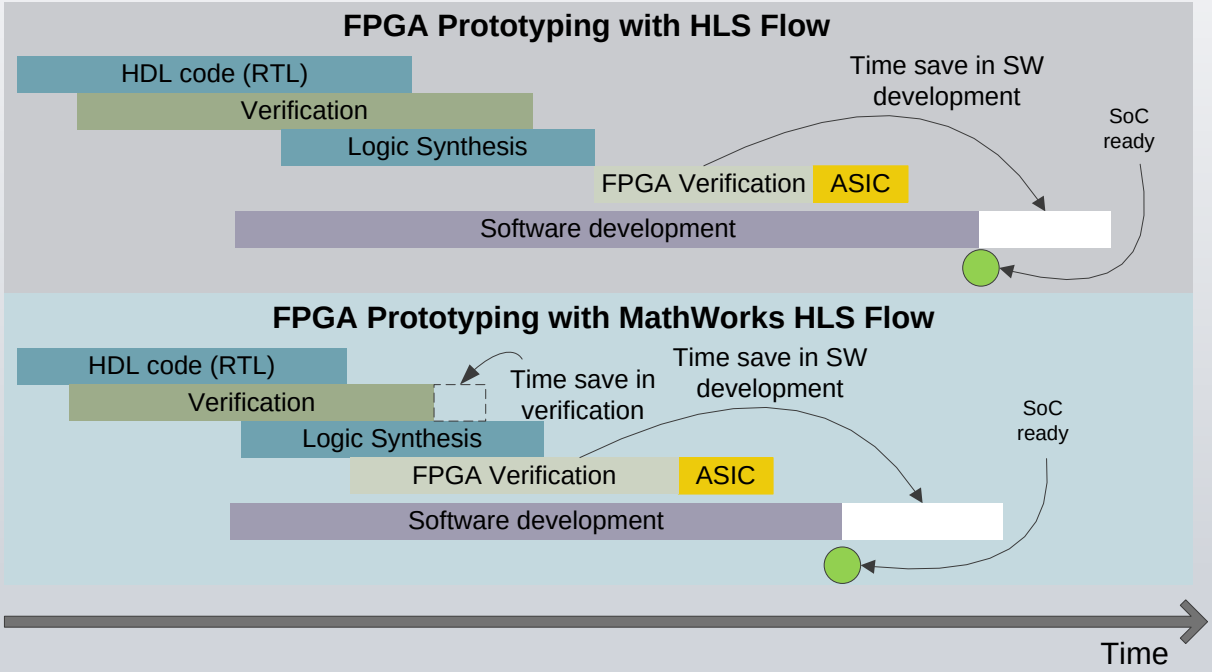


Further timing optimization could have been performed (Work focused on FPGA prototyping)



NOKIA

FPGA Prototyping Flow Timeline
Proportional Estimation in HDL Coder Flow



NOKIA

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)

NOKIA

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

NOKIA

Q & A


NOKIA