Codegeneratie voor FPGAs

Herman Roebbers

TASS

Onderwerp: Code generatie voor FPGAs
Datum: 4 October 2011
Aanwezigen: System Architecture Study Group

Overzicht presentatie

Over TASS
Code generatie voor FPGA
2 projecten

- OOTI Workshop System/Software Engineering
- Post-Game Analysis in a Mixed HW/SW environment

Wie is TASS B.V

TASS is een dienstverlener op het gebied van technische en Embedded software.

- Opgericht 1978 door Philips
- Onderdeel Total Specific Solutions sinds 2007
- Top drie speler in de Benelux voor technische software dienstverlening
- Ruim 200 medewerkers Nederland en Belgie
- Kennis van technische software ontwikkeling ontsluiten naar produkten en oplossingen

Missie

Als top-3 dienstverlener voor technische en embedded software, levert TASS een belangrijke bijdrage aan de ontwikkeling van intelligente en innovatieve producten.

Onze software professionals zetten hun kennis, kunde en toewijding in voor betere oplossingen, binnen de beschikbare time-to-market.

Zo helpt TASS haar klanten, om kwalitatief hoogwaardige producten nog efficiënter en sneller in de markt te zetten.

Visie

- Technische en embedded software speelt een steeds grotere rol in onze samenleving. Hoewel zelden zichtbaar voor de consument, vormt software vaak het hart van het product. De mate waarin het product zich onderscheidt van de concurrentie, wordt dan ook mede hierdoor bepaald.

Competences

Programming Target
DSP
Intel/Motorola
MIPS
ARM
Trimedia
FPGA:
Other

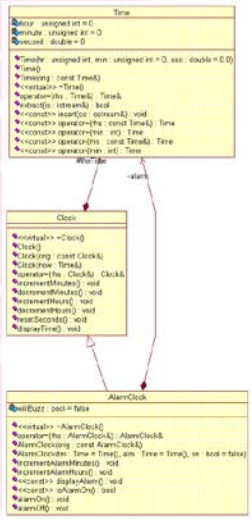
OS/API
Unix
Linux/RT
pSOS
VxWorks
Win NT/CE
MFC
Other

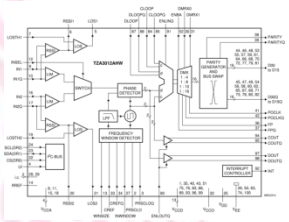
Languages
C/C++/C#
HTML/XML
VB-Scripts/VBA
Java/J2EE
COM/ATL
.NET
DCOM

Methods
SA/SD
OMT/UML
OORT
RMA
Agile (SCRUM, …)
Other

Dev'ment Tooling
ASD (Verum)
Matlab / Simulink
Labview

Special
Project mngmnt
QA/QO
Consultancy
Training





Interfaces/protocol
GSM/GPRS/UMTS
WiFi
Bluetooth
Zigbee

Interfaces/protocol
ODBC
WAP
JDBC
TCP/IP/Ethernet
UPnP
Webservices

Applications
Drivers, OS SW
MM Engines
Production comm control
Algorithm
Dig. A/V processing

Marktgebieden

- Mobility & Automotive
- Healthcare & Cure
- Mechanics & Control
- Consumer Lifestyle
- Safety & Security

Klanten


ASML


NXP
founded by Philips


océ


tomtom


BARCO
Visibly yours


nedap


BOSCH


PHILIPS


CATENA


ERICSSON


TNO

Partners


Fontys
Hogescholen


verum
Tools for building
mathematically
verified software


DSP
VALLEY


TU/e


Green Hills
SOFTWARE, INC.

Partners


Fontys
Hogescholen


verum
Tools for building
mathematically
verified software


DSP
VALLEY


TU/e


Green Hills
SOFTWARE, INC.

Introduction

Philips TASS sees possibilities of FPGA, a.o. SW ➔ system
In Handel-C (Celoxica DK), not in VHDL

Since 2000 many graduation projects on FPGA with various uni's (OOTI / UT)
Also projects (Philips Research / first tier Automotive supplier / …)

Handel-C as HDL

- Developed by Celoxica (1990-2008)
- C based specification language
- ANSI-C “compliant”
- Simple implicit timing model (1stmt == 1 clk)
- Accessible to HW and SW engineers
- One language for system design

(Philips) TASS: expertise in Handel-C

- HW IP in automotive industrie
- Demonstrators
- Board support package
- PixelStreams
- FPGA debugging
- MATLAB®/Simulink® ➔ FPGA conversion

OOTI = Ontwerpers Opleiding Technische Informatica, 2 yr. post-Master, TU/e

Two projects of 3 months each

- Workshop System / Software Engineering
- Goal: gain experience in industrial project

Goals

- More cooperation OOTI / TASS
- Offer inspiring assignments to students
- Gain experience, a.o. Model Based Design

Implement real-time control on FPGA

Non-trivial case

Automatic MATLAB®/Simulink® and Excel ➔ Handel-C

Via DK to RC203 (Xilinx XC2V3000)

First SW-in-the-loop, then HW-in-the-loop validation before connecting

Scanning Section

Automatic Document Feeder (ADF)

Stakeholders

OOTI Software Technology program

TU/e technische universiteit eindhoven / stan ackermans institute

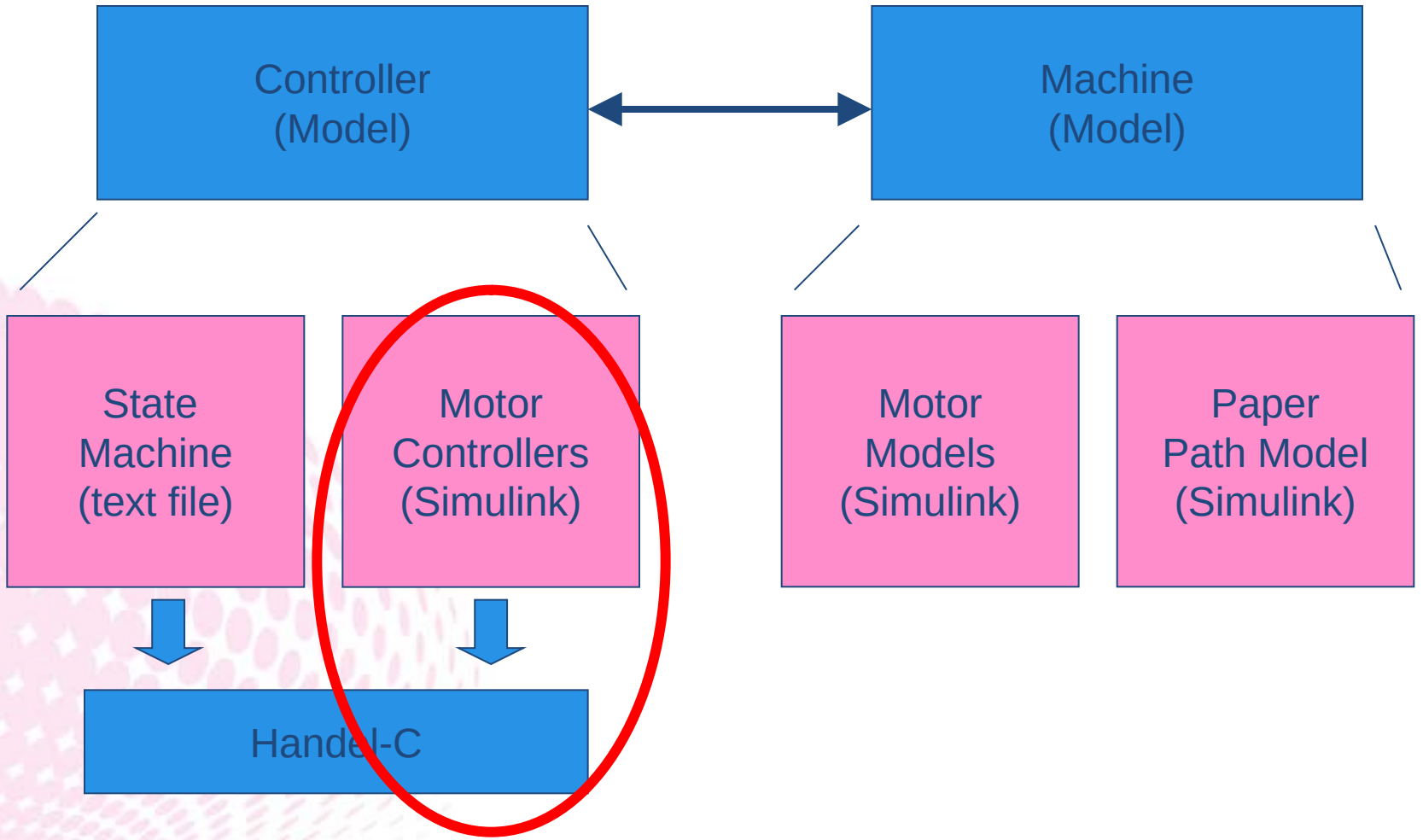
TASS software professionals

TASS

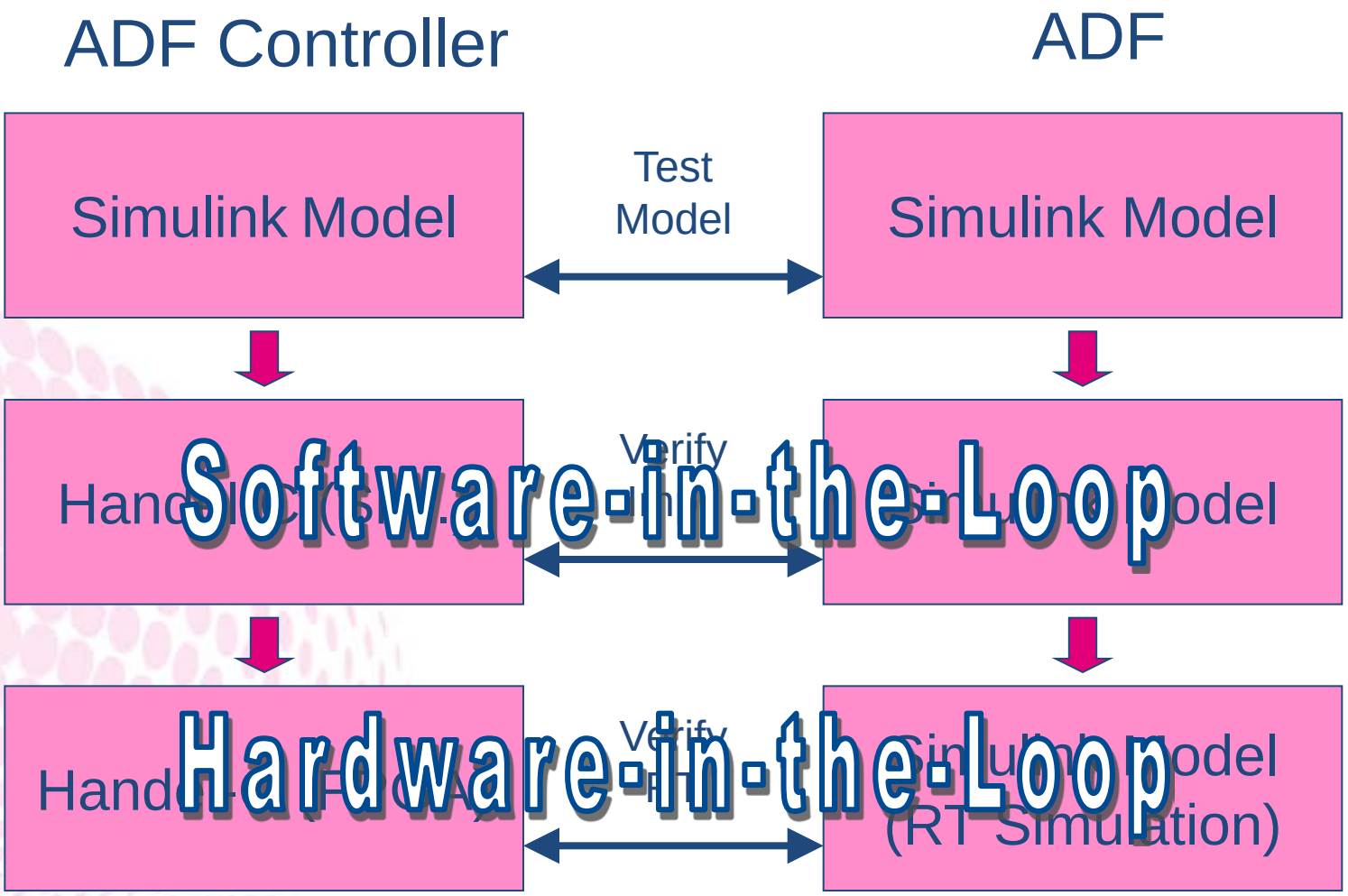
océ

Océ Technologies B.V.

ADF model

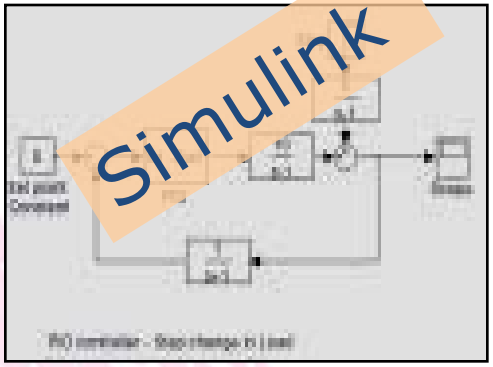

Controller
(Model)
Machine
(Model)
State
Machine
(text file)
Motor
Controllers
(Simulink)
Motor
Models
(Simulink)
Paper
Path Model
(Simulink)
Handel-C

Verification moments



2004 -> 2005: Size reduction

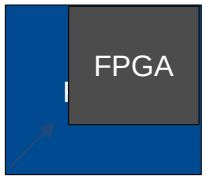
Motor control



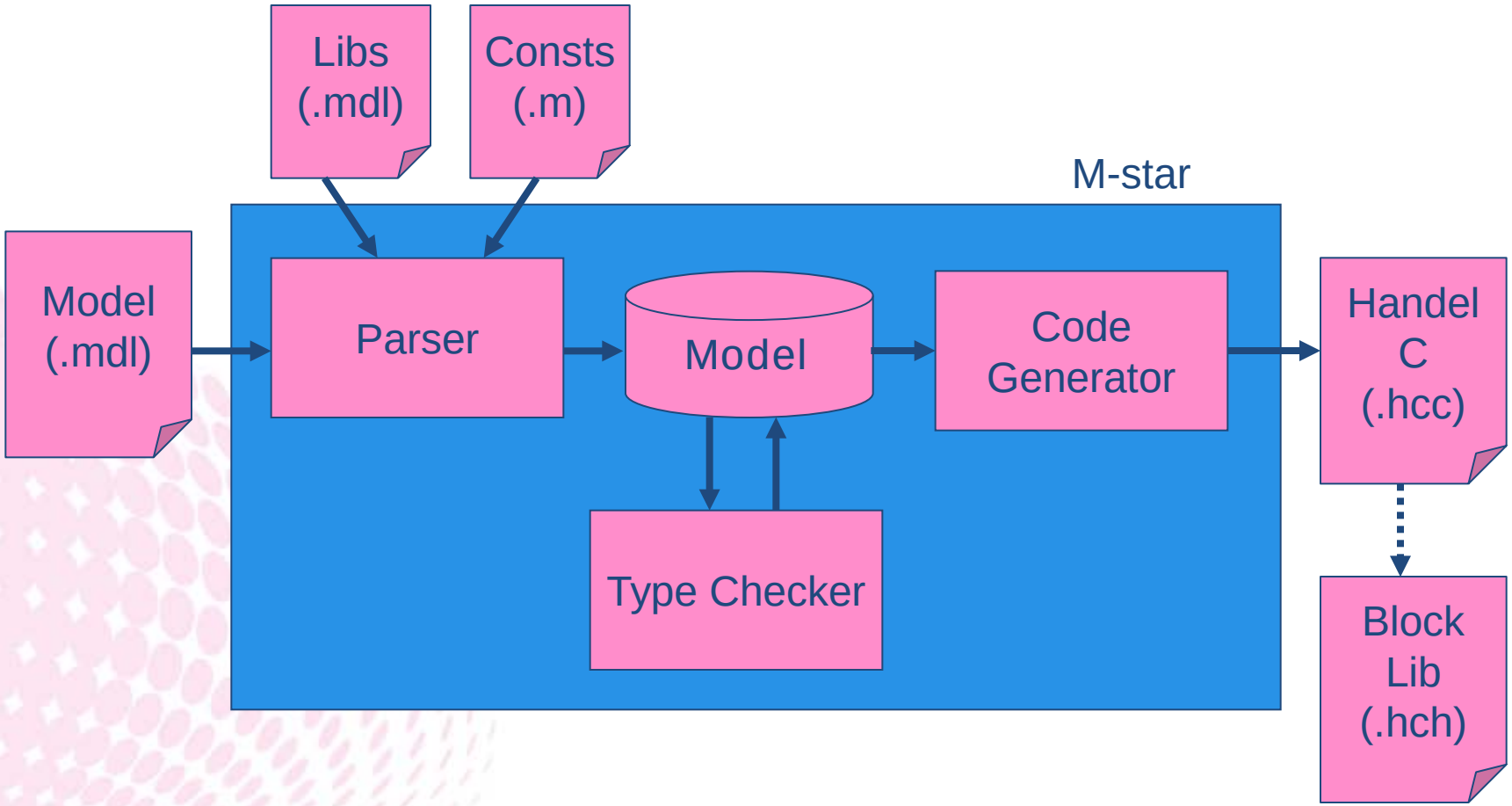
M-star

Celoxica / Xilinx

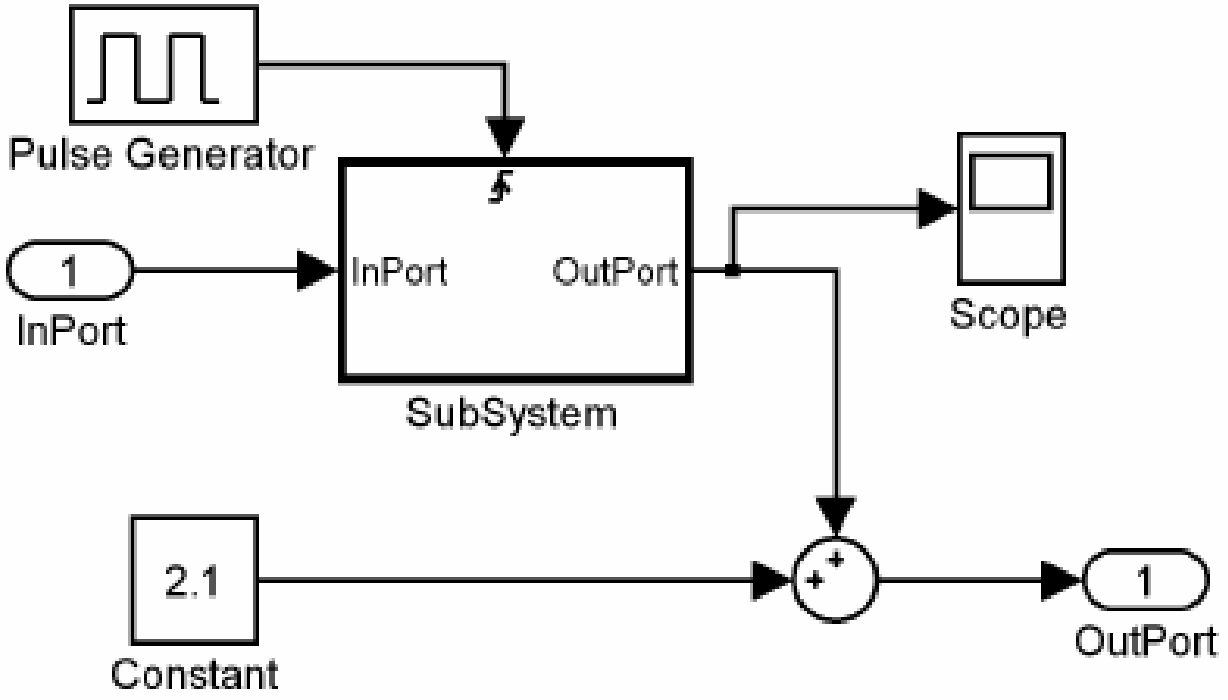
Size reduction



Simulink® to Handel-C



Simulink® to Handel-C
Example system



Simulink® to Handel-C Philosophy:

Simulink® block == Handel-C process

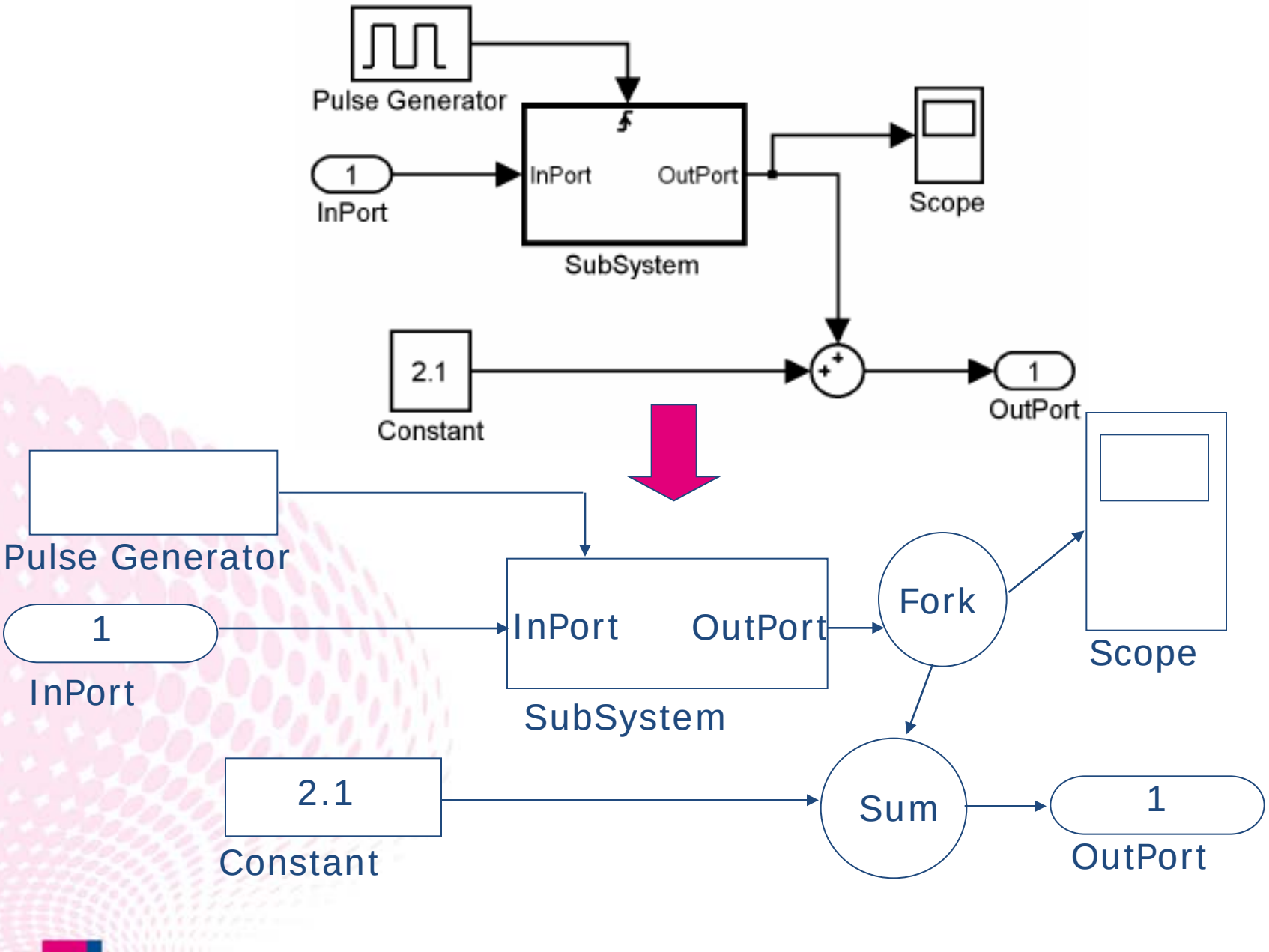
Processes communicate via channels

Size, type of channels derived from model

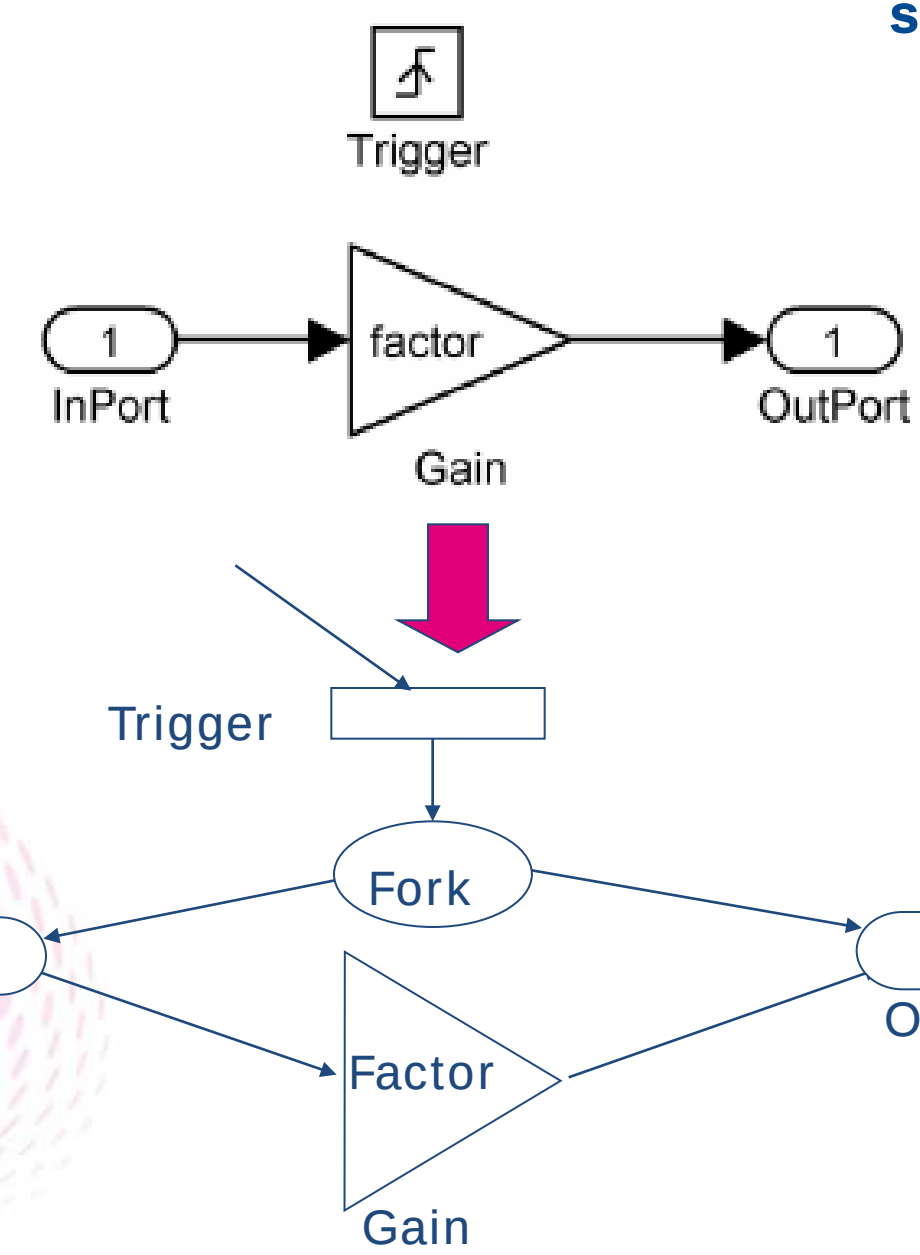
Processes run in parallel

Communicating Sequential Processes!

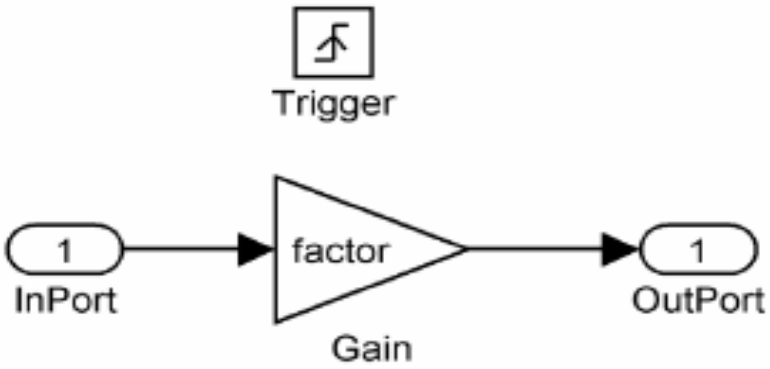
Simulink -> Handel-C



Simulink® to Handel-C subsystem_1



```
macro proc SubSystem_1(inputs, outputs, trigger_channel){
  chan signed 16 channel_0;
  chan unsigned 16 channel_1;
  chan unsigned 1 trigger;
  chan unsigned 1 triggerFork[2];
  par  {
    Fork(trigger, uvalue, triggerFork, 2);
    TriggerPort(trigger_channel, trigger, RISING);
    SubSystemInPort(*inputs[0], uvalue, channel_1, {&triggerFork[0]}, 1);
    Gain(channel_1, uvalue, UNSIGNED, 0, channel_0, SIGNED, 8, 896,
     SIGNED, 16, 8);
    SubSystemOutPort(channel_0, svalue, *outputs[0], {&triggerFork[1]},
     1);
  }
}
```



WSSE'04 results used in assignment UT at TASS

ACSIF

"Automatic Conversion from Simulink to FPGA"

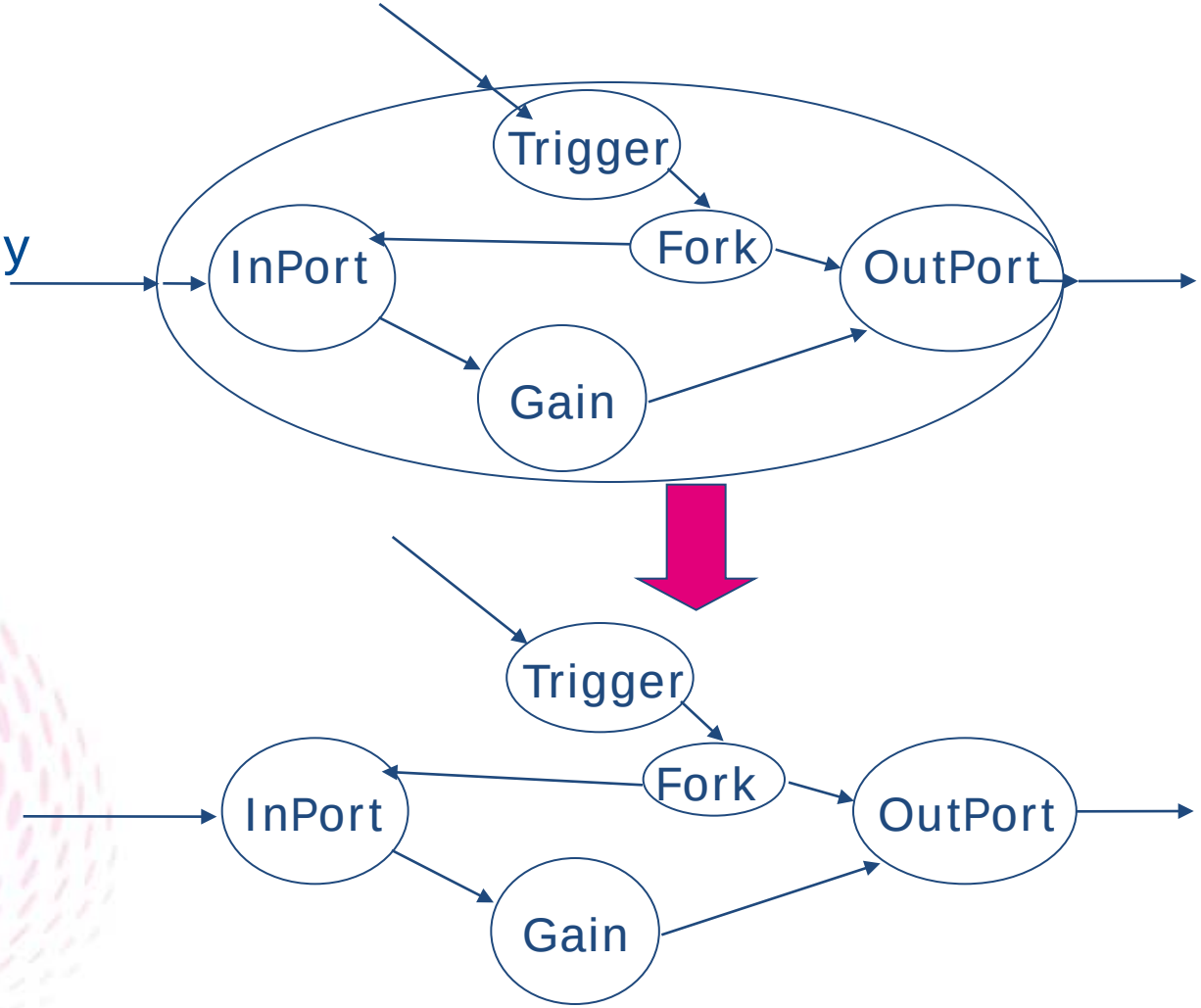
Additional blocks defined

Corrected errors

Starting point for WSSE'05 project

Simulink® -> Handel-C optimization ('05)

Remove
process hierarchy
(recursively)



Replace communication by assignment

```
par
  {
    a ! C;        ➔    D = C;
    a ? D;
  }
```

Simulink® to Handel-C optimization (‘05)

```
while (1)
{
  in ? Var;
  .....
  out ! Var;
}
```

➔

```
{
  Var = in;

  Out = f(Var);
}
```

Simulink® to Handel-C optimization (‘05)

|| + channels ➔
Scheduling implicit (no worry)

Remove || + channels ➔
Own scheduling to keep semantics
Where necessary add buffering (loops)

Simulink® to Handel-C optimization (‘05)

Use shared macros

One hardware block ➔ reused.

Macro only invoked 1 at a time.

Compiler does not check this.

Generator does check.

Simulink® to Handel-C optimization (‘05)

Now less ||

- Less, but sufficient performance (MHz)

Less gates

Results

Projects successful:

Via SIL and HIL to ADF in 3 months

ADF functions according to specification

Reduction of FPGA resource with 40 à 50 %

Code generation from MATLAB® / Simulink® for simple blocks

Approximately 40 blocks implemented

Article about WSSE'04 in Financiële Telegraaf

Paper about WSSE'04

- **Automatic Handel-C Generation from MATLAB® and Simulink® for Motion Control with an FPGA** (Communicating Process Architectures 2005)

Artikel in Bits & Chips

- OOTI studenten genereren FPGA code vanuit Matlab en Simulink (2007)

Support

Celoxica
Software-Compiled System Design

University Programme, Handel-C course, FPGA board, DK licences

Embedded Systems INSTITUTE

(BodeRC project) MATLAB® / Simulink® models

The MathWorks

MATLAB®/Simulink®, code gen.

XILINX®

University Programme, FPGA software

Spread

This work has influenced further work done in several STW projects at UT and TU/e

- ViewCorrect: Broenink (UT), Corporaal(TU/e)
  - HW/SW Design Space Exploration on the Production Cell Setup (Communicating Process Architectures 2009)
  - FPGA based Control of a Production Cell System (Communicating Process Architectures 2008)

Conclusions

- Using Handel-C it is fairly easy to generate code from process graphs (CSP heritage).
- Need to have static typing, don't want dynamic typecasting
- Floating point is big, non-trivial but can be done
- Implementation is fast and efficient
- Need library code for each building block.
- Like to have control over amount of transformation (serialization applied by optimization process) to control performance.

Epilog

2008: Celoxica sells Handel-C to Catalytic;
Catalytic merges with Agility Design Solutions

2009: Mentor Graphics buys Handel-C.

- No promotion
- Just enough development to allow existing customers to continue using it under Mentor's licensing mechanism
- Mentor much rather sell their own Catapult-C which is 10x more expensive, but does not have parallelism in the language.

Alternative C to hardware

Very promising in terms of Quality of results:

AutoPilot from AutoESL.

Recently (31-1-2011) acquired by Xilinx.

Basically “C/C++/SystemC with #pragma’s”

For more information contact:

TASS B.V.
Larixplein 6
5616 VB Eindhoven
T +31 40 250 3200
info@tass.nl
www.tass.nl