



# Esterel Studio

## Update

Kim Sunesen

Esterel EDA Technologies

[www.esterel-eda.com](http://www.esterel-eda.com)

**Synchron, November 2007, Bamberg Germany**



# Agenda

- ▶ Update on Esterel Studio
- ▶ Architecture Diagrams
- ▶ Formal Verification
- ▶ IEEE standardization
- ▶ Early Performance Estimation





# Industries Served



## Esterel Studio™

*Front-end design and verification suite for control-intensive hardware*

**Rigorous & unambiguous executable specifications**

**ESL synthesis path to efficient RTL / SystemC code**

**Formal Verification**



## SCADE Suite™

*De-facto Standard for Safety-critical avionics embedded software*

**DO-178B Level A certified systems**

**Automatically-generated C code**



## SCADE Drive™

*Safety-critical automotive embedded software*

**Code generator certified by TUV - IEC 61508 standard**

# Esterel Customer Consortium

- ▶ Best practice sharing
- ▶ Design flow integration
- ▶ Collaborative roadmap
- ▶ Implemented on projects
- ▶ **First Silicon 2005**





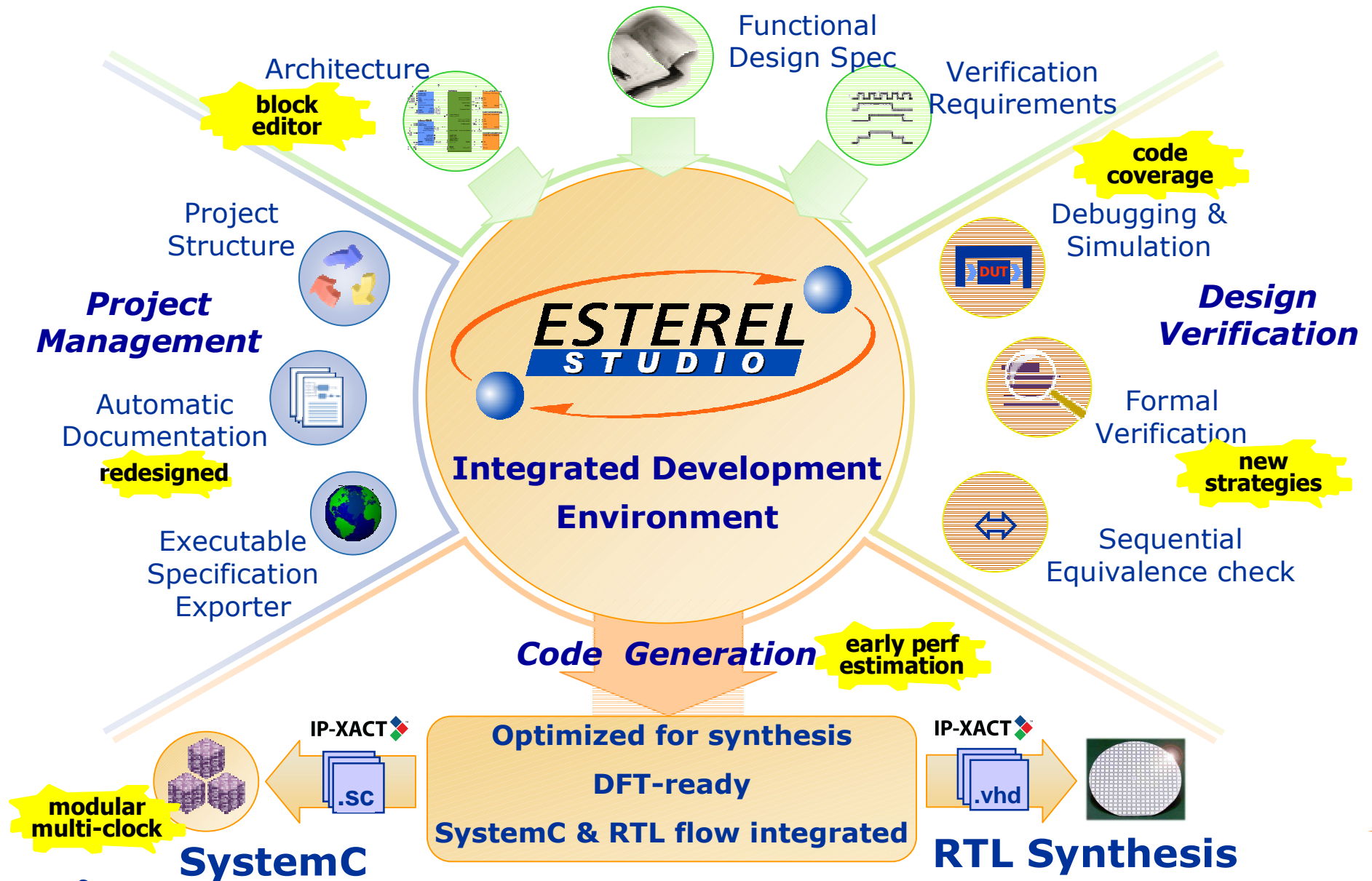
## ***Esterel Consortium Industrial Designs***

- ▶ 50+ industrial designs, some with ECOs
- ▶ Interoperability with 16 EDA tools from ESL to DFT
  - ✓ AHB, APB, OPB, OCP, AXI bus bridges, interfaces, converter, transactors
  - ✓ Cache coherence implementation and verification
  - ✓ Processor architecture validation
  - ✓ Processor & co-processor
  - ✓ Traffic controller, SD/MMC Card controller
  - ✓ Memory architecture
  - ✓ External Memory controller
  - ✓ Serial ATA
  - ✓ DMAs
  - ✓ Power management
  - ✓ Video streaming
  - ✓ Battery controller
  - ✓ UMTS/GPRS protocol specification and verification
  - ✓ Smart Cards security formal verification
  - ✓ Video controller

# Esterel Studio: the Front-end Design and Verification Suite

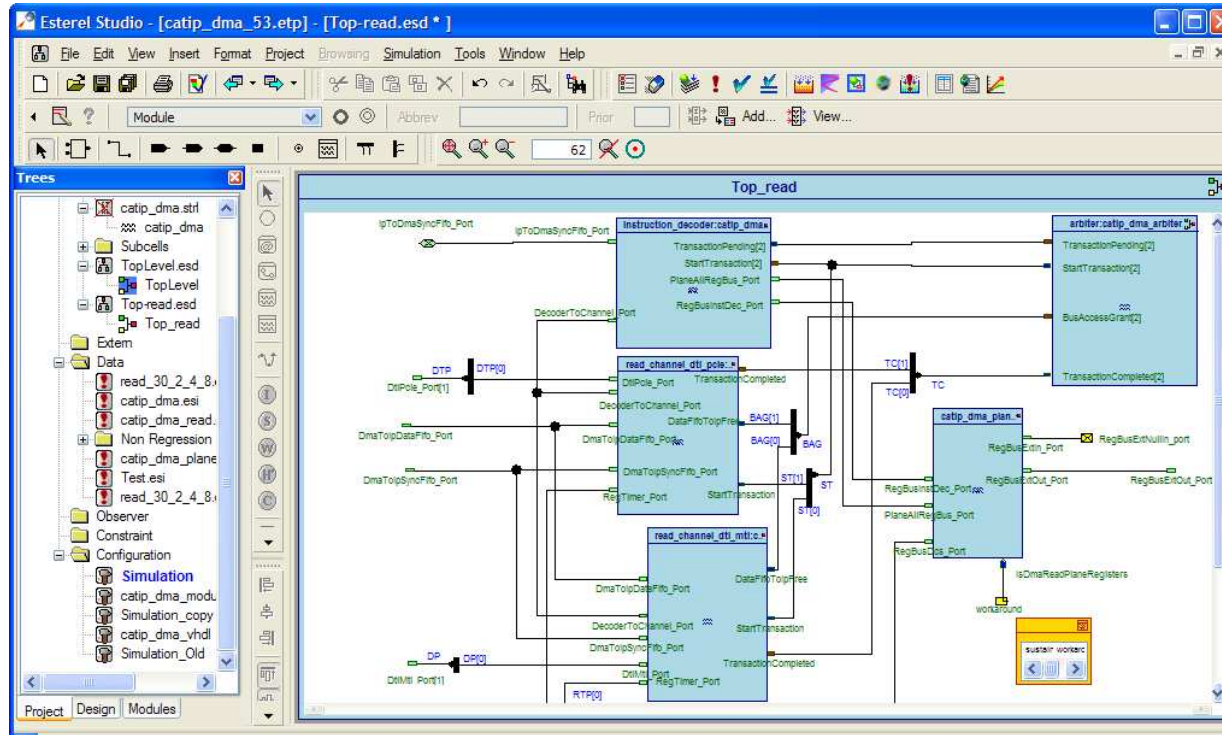
## Design Specification Capture

new in 6.0



new in 6.0

# Architecture Diagrams Editor



- ▶ Fully integrated in the workspace (navigation, tree, menu)
- ▶ Dedicated view
- ▶ Architecture diagram modules are hierarchical like SSM

# Architecture Diagram Components

Connection Bar

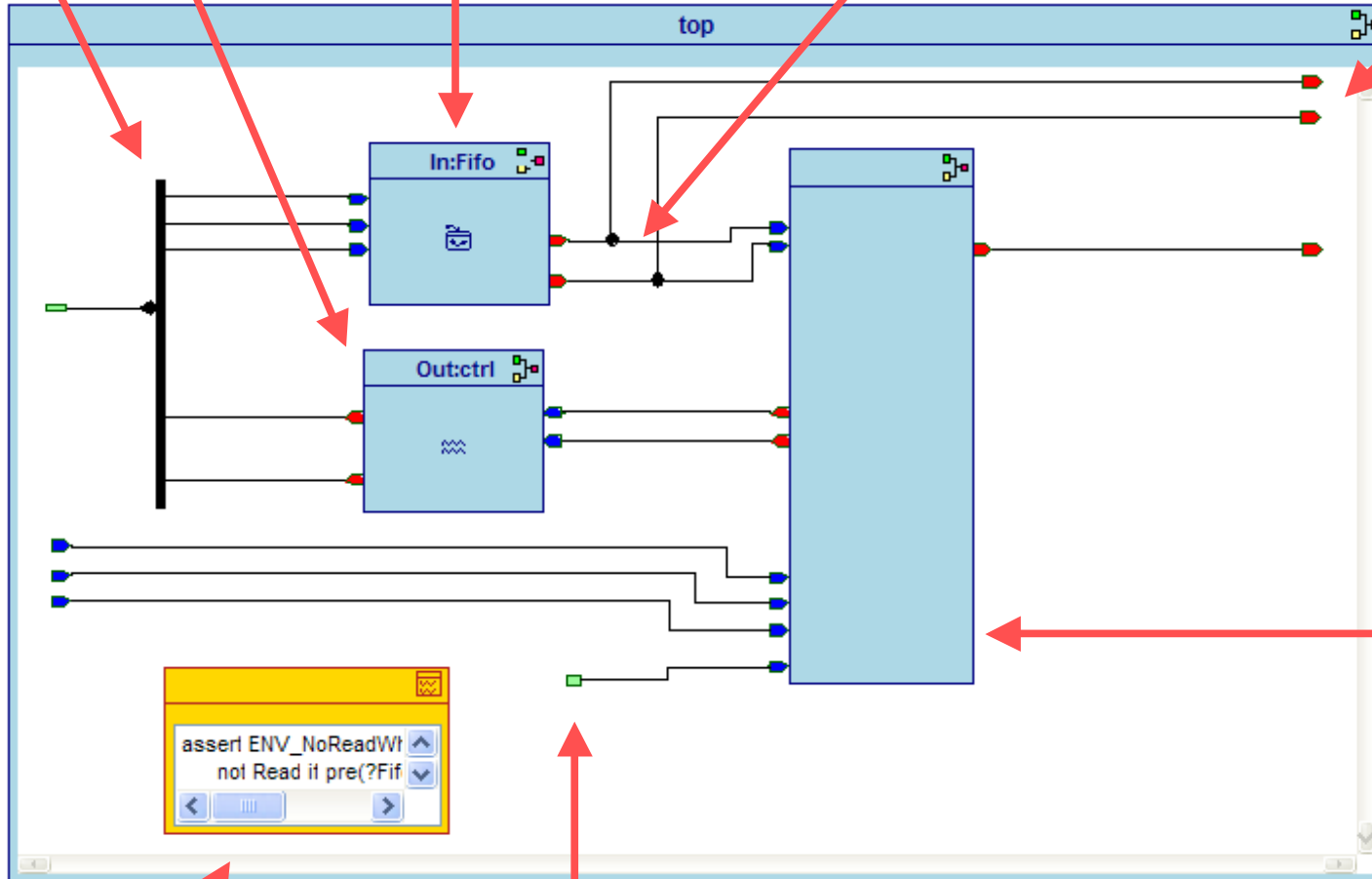
Bar

Textual Instance

SSM Instance

Connection point

Output Ports



Hierarchical Architecture Diagram

Textual block

Local Port






# Formal Design Verifier

## Auto-Asserts enable quick start


- ▶ Auto-asserts are automatically extracted from the design
- ▶ Auto-asserts prevent common mistakes

 Out-Of-Range

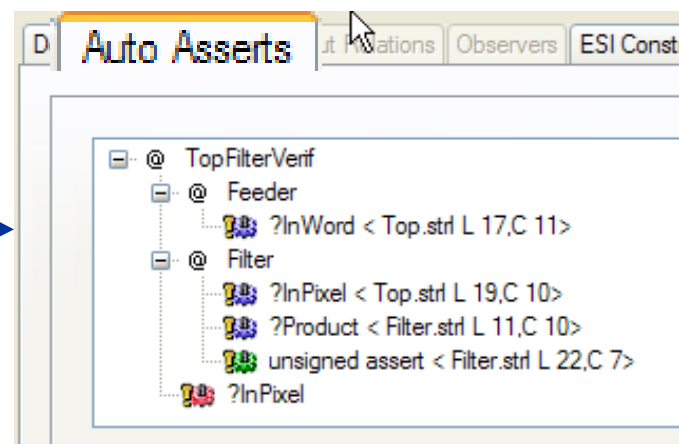
- Overflow errors, Array index errors, Division by zero

 Read-Before-Write

- Reading uninitialized signals before writing them

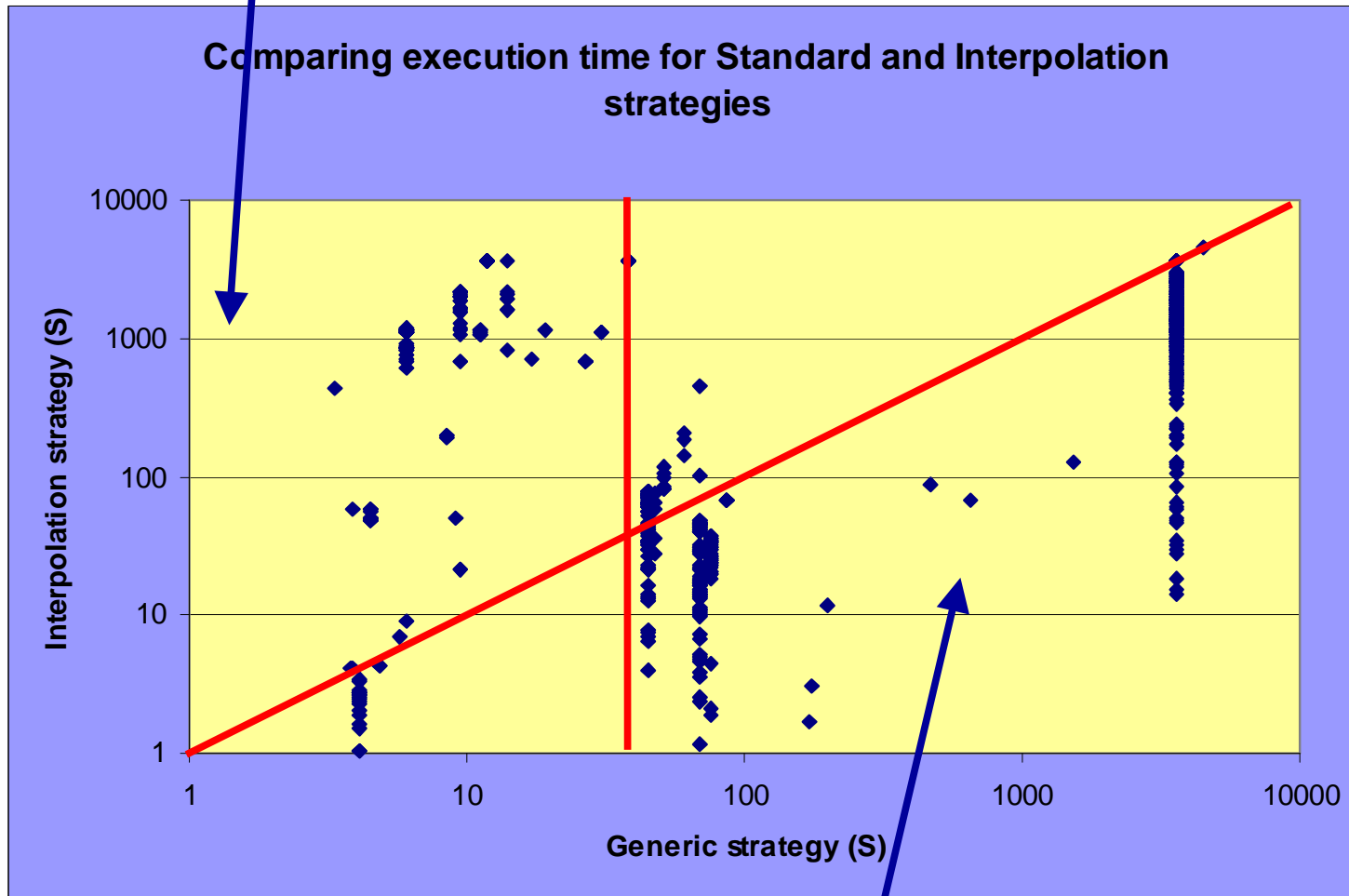
 Single Signals

- Multiple emissions (write / write races)





Standard strategy verifies very quickly “simple” properties



Interpolation more effective on  
“complex” properties



# ***Esterel language IEEE standardization***

- ▶ DASC (Digital Automation Standards Committee) and IEEE group started standardizing the Esterel language
  - Language is public and Language Reference Manual will become property of IEEE
  - started March 2007
  - output : the **Language Reference Manual (LRM)** in 2009
  
- ▶ Industry participants
  - General Motors, IBM, Intel, Microsoft, NEC, NXP, Orange, ST Microelectronics, Synopsys, Texas Instruments
  
- ▶ Academic participants
  - Columbia Univ., E.I. Geneva, INRIA, Kaiserlautern Univ., Technion Israel, Tübingen Univ, etc.





## Current Hot Subjects

- ▶ Data definition and data expressions
  - **struct**, tuples
  - simplified u2bin / bin2u conversions
  - **functional module calls**
- ▶ Signals, Interfaces, and modules
  - **declaration simplification** (*temp value* the default)
  - struct = values of ports
  - port emission by struct assignment
  - **module behavior extension** (= code in interfaces)
- ▶ Genericity
  - Improved and generalized mechanism





# Early Performance Estimation

## The Problem

- ▶ From a high-level Esterel model, designers wish to
  - Identify created operators
  - Evaluate the size and speed of the resulting circuit,
- ▶ But running synthesis is too costly
  - Slows down the optimization loop
- ▶ Quick approximate estimation is needed!

## The Solution

- Same type of information as synthesis:
  - actually allocated registers
  - estimated area consumption
  - max estimated delay path
  - operator count
- Estimation is quicker than synthesis (> 6x)
- Tracing back costly constructs is made easier



## HTML Screenshots (1/3)

### Estimation Summary

| Combinational Area | Max Delay | Register |
|--------------------|-----------|----------|
| 3112.14            | 38.04     | 79       |

- General summary for the entire design

### Max Delay Path

**Startpoint:** register PauseReg\_WaitForWord\_38\_1\_L1D1

**Endpoint:** output OutPixel\_data

| Point                            | Module      | Incr  | Path  |
|----------------------------------|-------------|-------|-------|
| PauseReg_WaitForWord_38_1_L1D1   | Feeder      | 0.00  | 0.00  |
| Then1_7_1_L1D1_and_x_506_507     | Feeder      | 1.08  | 1.08  |
| Status_InPixel_S8_0_or_x_602_603 | Filter      | 2.16  | 3.24  |
| Then2_13_2_and_604_605           | PixelFilter | 0.95  | 4.19  |
| sv_DelayLine_V11[0]__mux_607     | PixelFilter | 1.00  | 5.19  |
| factor_685                       | PixelFilter | 11.00 | 16.19 |
| sv_Product_V9[0]__mux_686        | PixelFilter | 1.00  | 17.19 |

- Max estimated delay path
- With source module info



## HTML Screenshots (2/3)

### Input - Output Combinational Paths

| Output        | Inputs                      |
|---------------|-----------------------------|
| OutPixel_data | (InWord, InWord_data[31:0]) |
| Ready         | (InWord)                    |
| OutPixel      | (InWord)                    |
| OutEndOfLine  | (InWord)                    |

Tells whether design is Mealy vs. Moore

### Module List

| Module      | Instanciated | Area    |
|-------------|--------------|---------|
| Filter      | 1            | 45      |
| Feeder      | 1            | 99      |
| PixelFilter | 1            | 2968.14 |

Module instance sizes





# HTML Screenshots (3/3)

## Operator Count

### Bool operators

| Operator  | BitWidth1 | BitWidth2 | Area | Number | Total Area |
|-----------|-----------|-----------|------|--------|------------|
| or        | 5         |           | 5.00 | 7      | 35.00      |
| and       | 3         |           | 2.00 | 44     | 88.00      |
| dszdyni_1 | 2         |           | 1.00 | 38     | 38.00      |

Operators count sorted by signal source types

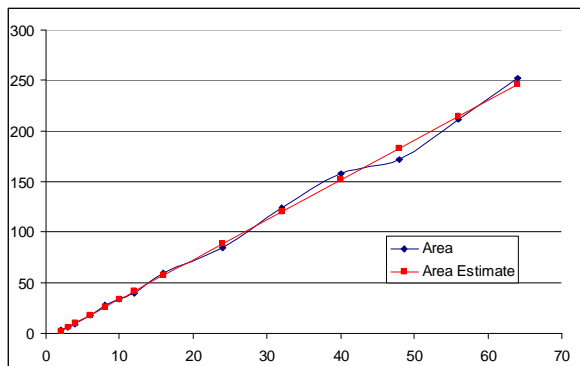
### Unsigned operators

| Operator | BitWidth1 | BitWidth2 | Area   | Number | Total Area |
|----------|-----------|-----------|--------|--------|------------|
| factor   | 8         | 3         | 406.45 | 5      | 2032.25    |
| add      | 12        | 11        | 359.95 | 1      | 359.95     |
| add      | 11        | 11        | 342.47 | 1      | 342.47     |



## How Early Perf is calibrated

1. **Measures:** every operator was synthesized with a variable bit-size and fanin
2. **Calibration:** using those previous synthesis result, one determines an operator sizing formula or table



| Fanin | Area | Perf Area | Av_Error%  | Delay | Perf Delay | Av_Error%   |
|-------|------|-----------|------------|-------|------------|-------------|
| 2     | 2    | 2         | <b>8.9</b> | 0.87  | 0.95       | <b>8.77</b> |
| 8     | 5    | 5         |            | 1.4   | 1.75       |             |
| 10    | 8    | 7.47      |            | 2.07  | 2.02       |             |
| 12    | 9    | 8.81      |            | 2.24  | 2.29       |             |
| 16    | 10   | 11.51     |            | 2.18  | 2.53       |             |
| 24    | 18   | 16.9      |            | 3.01  | 2.66       |             |
| 32    | 38   | 22.3      |            | 2.53  | 2.79       |             |
| 40    | 35   | 38.58     |            | 3.01  | 2.92       |             |
| 48    | 46   | 40.68     |            | 3.33  | 3.06       |             |
| 56    | 43   | 42.77     |            | 3.12  | 3.19       |             |
| 64    | 43   | 44.87     |            | 3.12  | 3.32       |             |

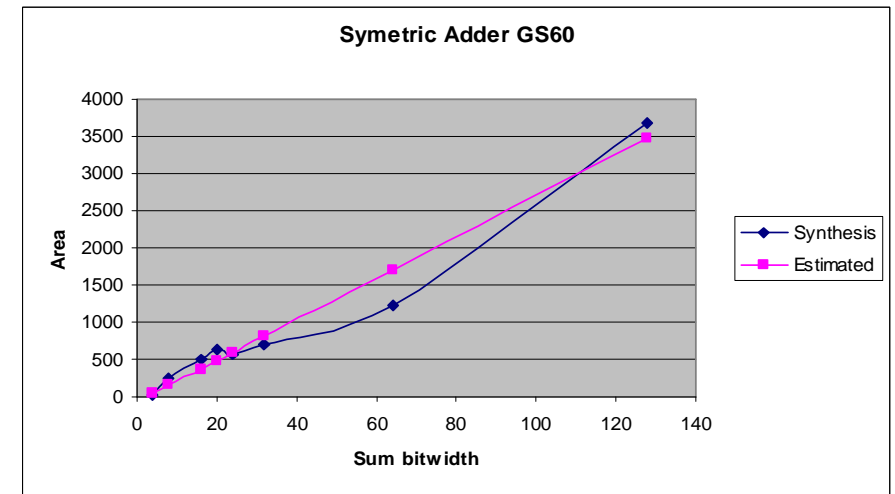
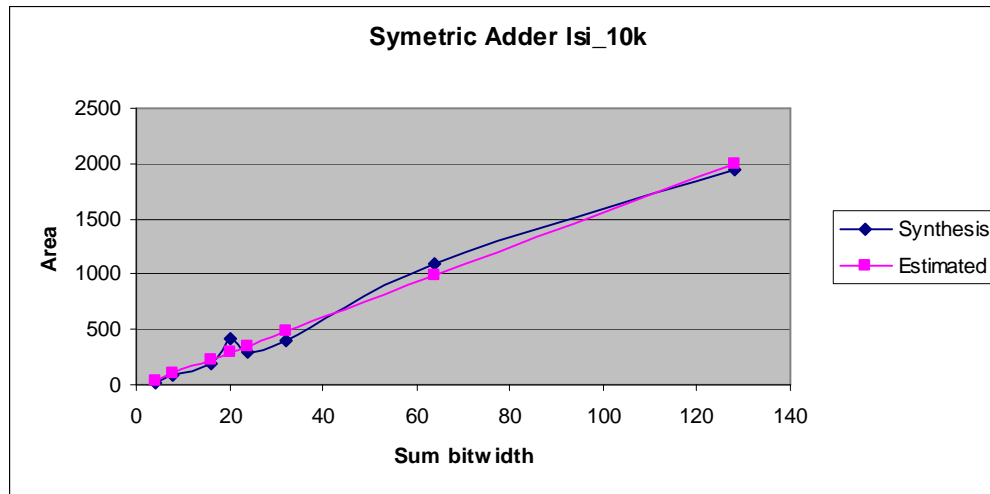


## Results on Industrial designs

| Design type              | Area (syn) | Area (perf) | Area error rate | Speed (syn) | Speed (perf) | Speed error rate | time gain  |
|--------------------------|------------|-------------|-----------------|-------------|--------------|------------------|------------|
| Write posting bus module | 9394       | 11744       | 25%             | 18.42       | 20.6         | 12%              | 7x faster  |
| Line filter              | 9235       | 10694       | 16%             | 47.2        | 51.77        | 10%              | 57x faster |
| DMA 1                    | 8515       | 14387       | 69%             | 22.87       | 23.06        | 1%               | 12x faster |
| Transactor AXI           | 34757      | 69810       | 101%            | 44.22       | 72.83        | 65%              | 6x faster  |

# Technology & Timing dependency

- Current areas / frequencies are based on the Isi\_10k library
- They can be adapted to other libraries by re-calibration



- Remark : early performance estimation does not consider timing constraints



# Thank you!

