OccN

A Methodology for NoC

AST Grenoble

Marcello Coppola

Outline

- Toolip overview
- SoC today
- NoC
- OCCN
- Test study
- Conclusion

TOOLIP Focus

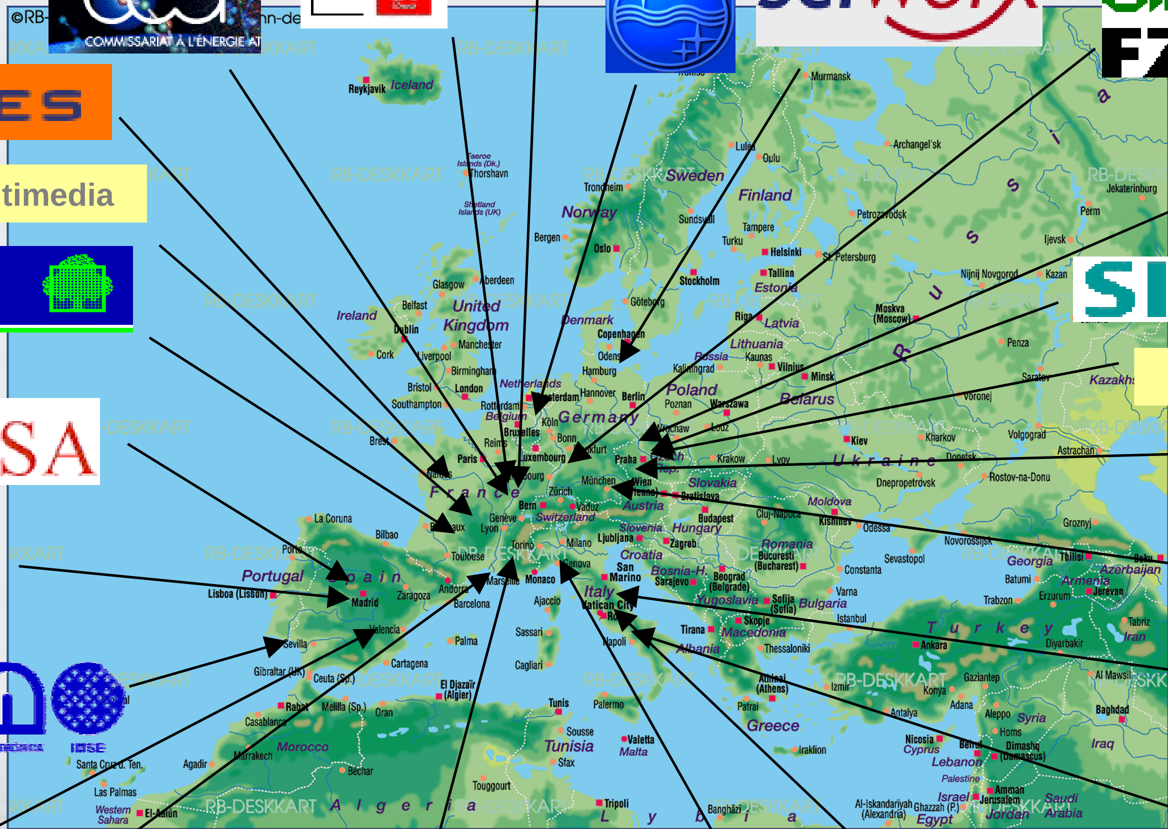
- Residential and home applications are becoming increasingly complex systems and application
- Industry requires a high level of integration of various functions
- Integration of whole systems on a single chip

Manage complexity by
- Parametric and reusable IP cores
- System-level modelling
- Verification techniques
- Design flow
- IP Qualification

Objectives:
- Shorten design times cycles
- First time silicon success
- Reducing design complexity
- Ease simulation, verification and test
- Make „reuse“ a feasible reality

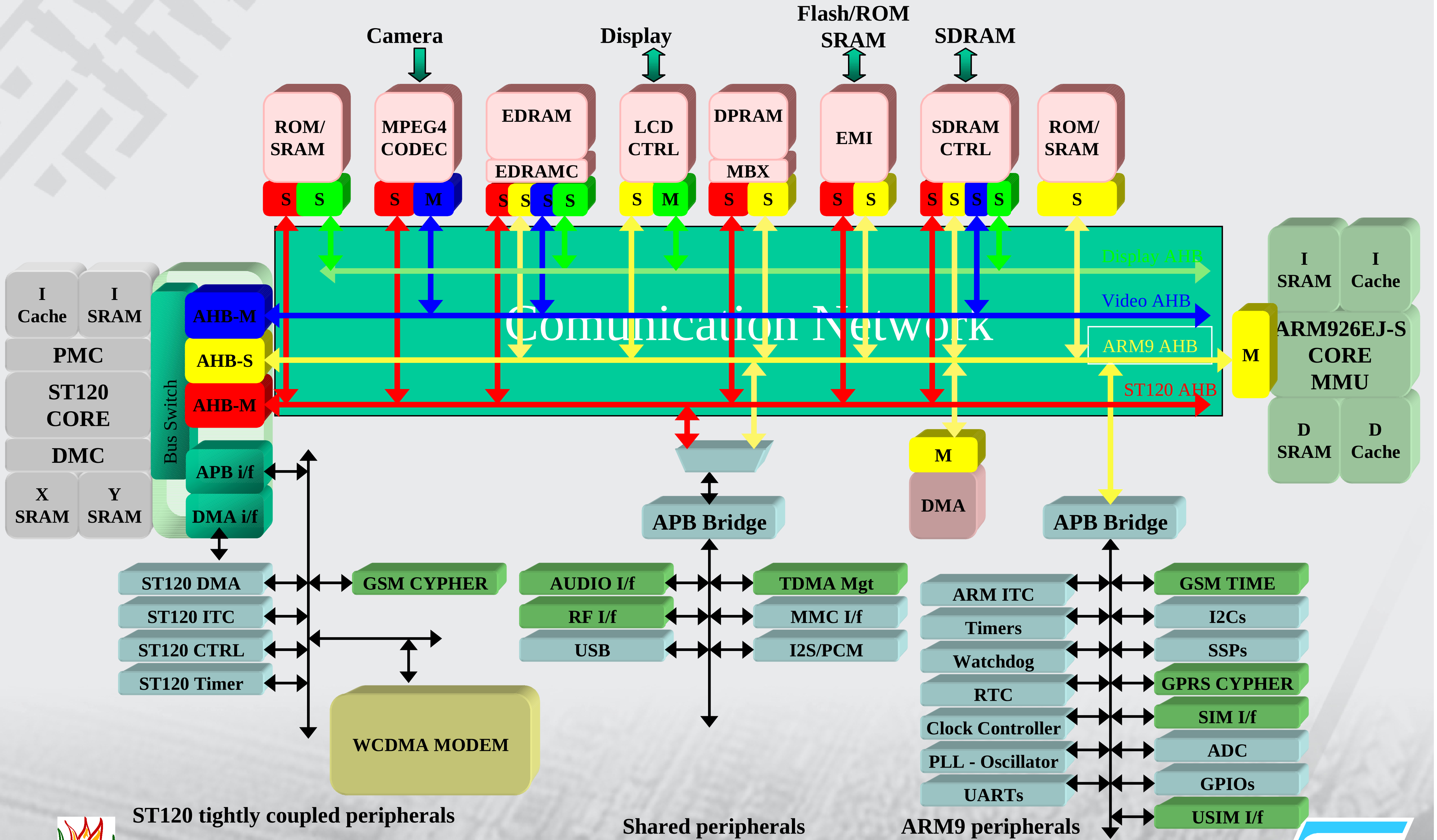
TOOLIP Partners

Thomson multimedia



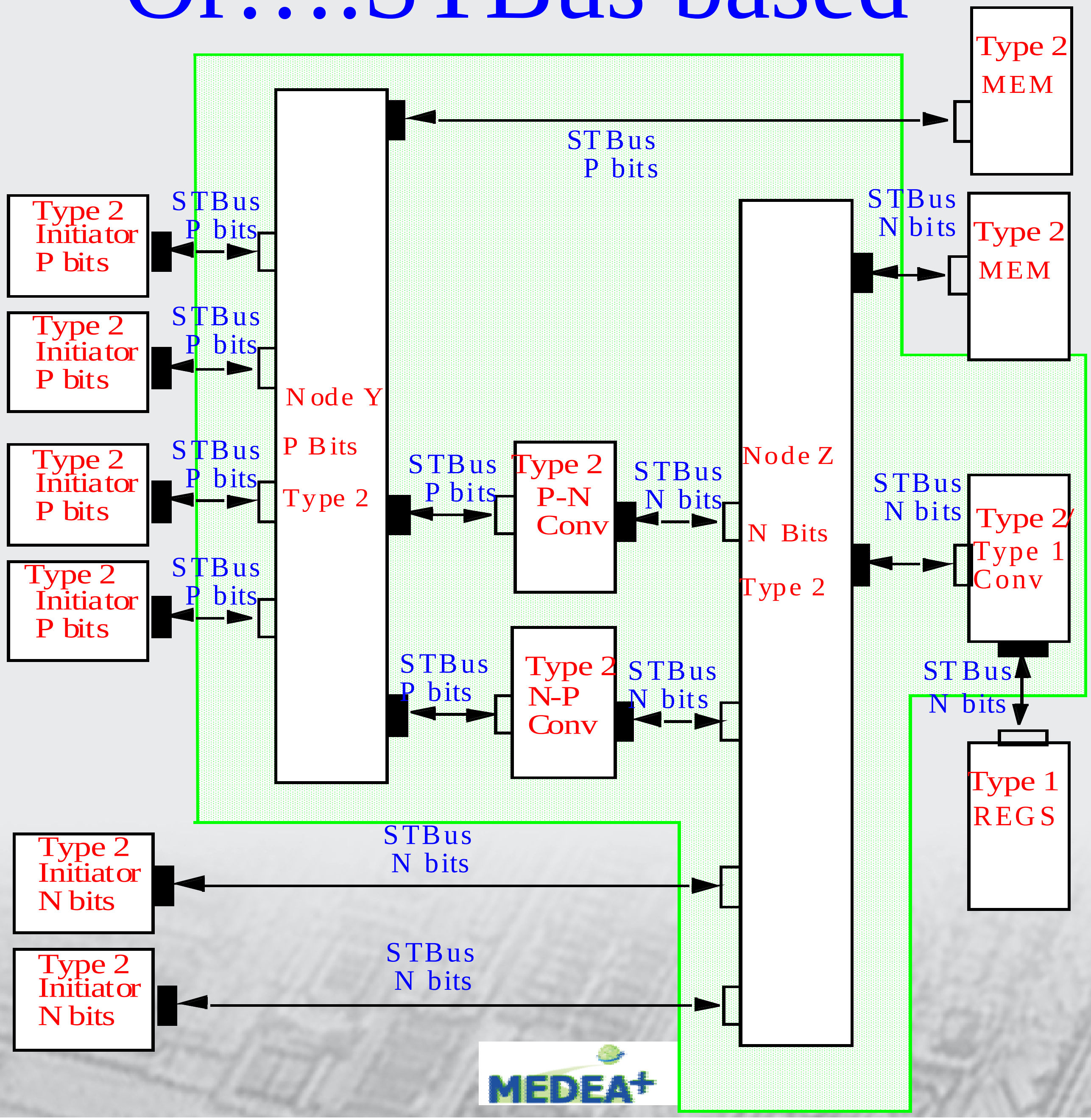
Thomson MM DTB

MP-SoC architecture: AMBA based



MEDEA+

Or….STBus based



STBus Interfaces

Interface Type	Initiator	Target
Type 1 : Peripheral - Simple synchronous handshake - Limited transaction set	ST20-C1	Peripherals (UART, timer) On-chip SRAM ROM
Type 2 : Basic System - Supports split, pipelined accesses	ST20-C2 core customer ASICs	Flash EMI SDRAM EMI
Type 3 : Advanced System - Supports split, pipelined accesses - Supports out of order execution - Shaped packets	ST40 / ST50 Core multi-channel dma	PCI master DDR LMI

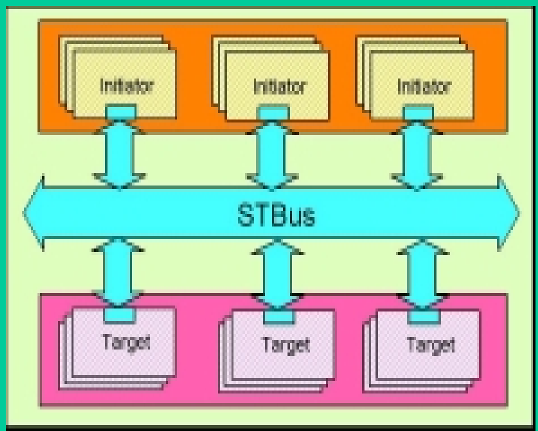
STBus Building Blocks

- Node
  - Performing arbitration and routing
- Buffer
  - Performing retiming
- Size Converter
  - Allowing the communication between two blocks having different bus sizes
- Type Converter
  - Allowing the communication between two blocks following different STBus protocols

From SoC to NoC

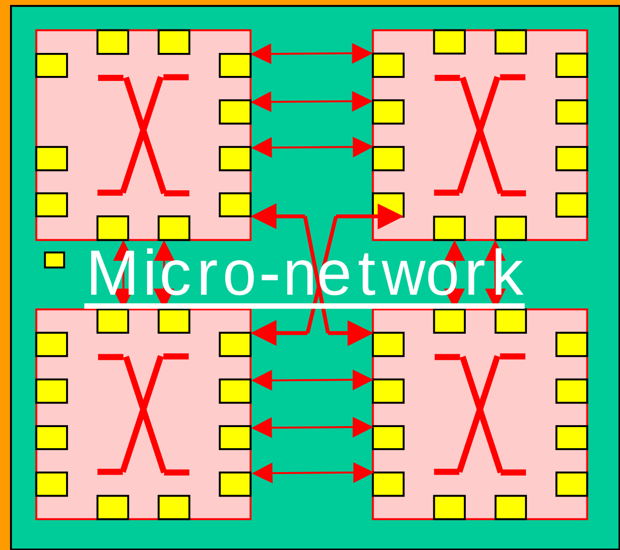
System on a chip

Programmable computation
Hardwired interconnectivity
Partially distributed storage



Network on a chip

Programmable computation
Programmable interconnectivity
Fully distributed storage

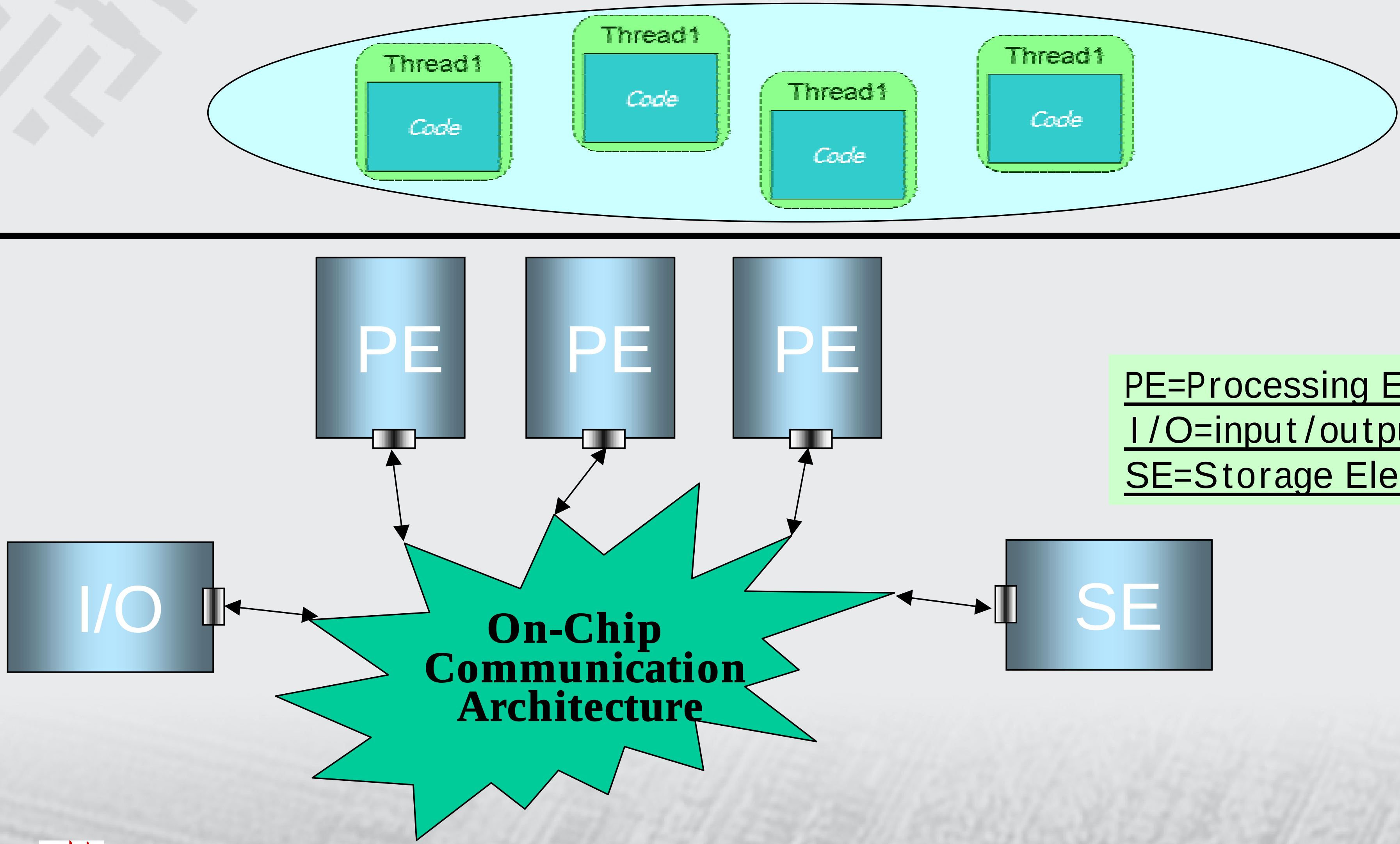


Some definitions

- NoC is a future view as a micro-network of components [Benini and De-Micheli]
- NoC is a parallel computation platform with a task/process level of parallelism; suitable only for high-volume products [J.P Soininen and H Heusala]
- NoC is a set of computation node connected via sophisticated on-chip communication network [A.A Jerraya et alt.]

MEDEA+

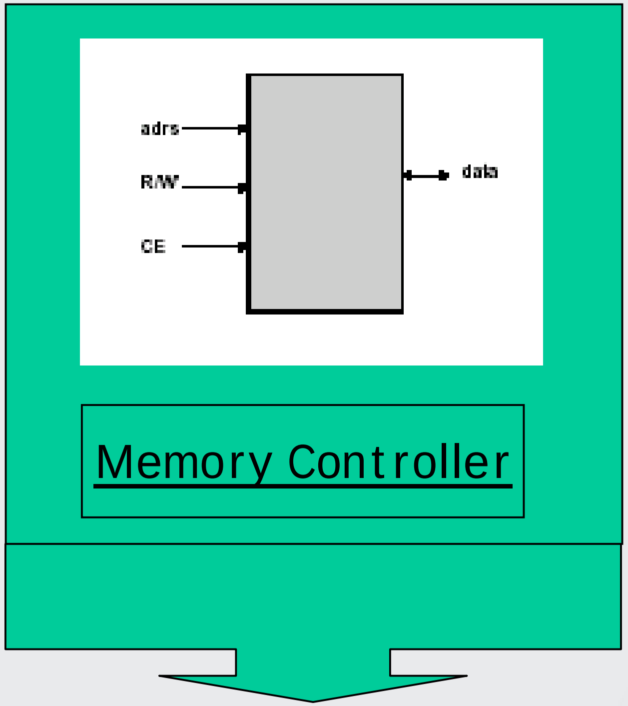
NoC



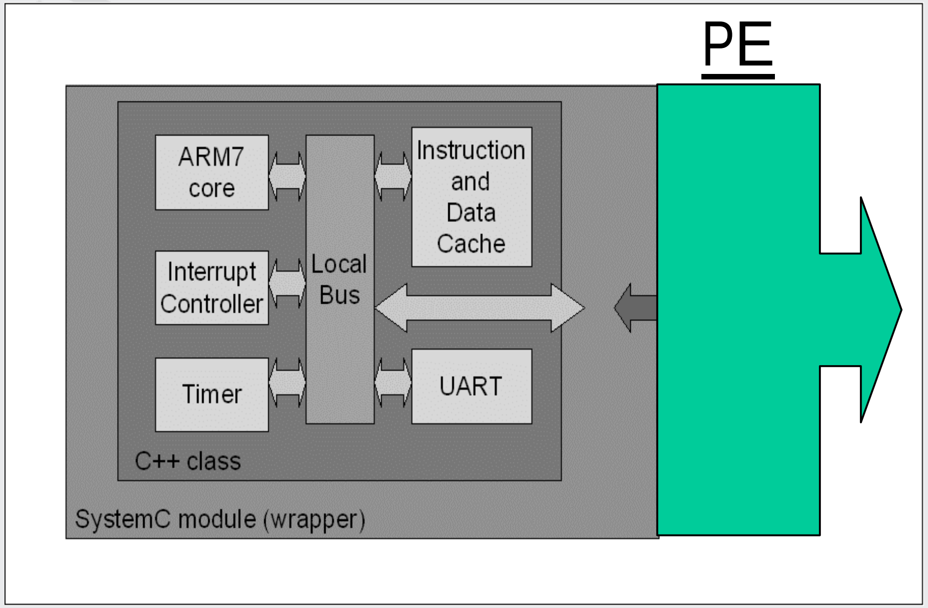
PE=Processing Element
I/O=input/output
SE=Storage Element

PE,I/O,SE

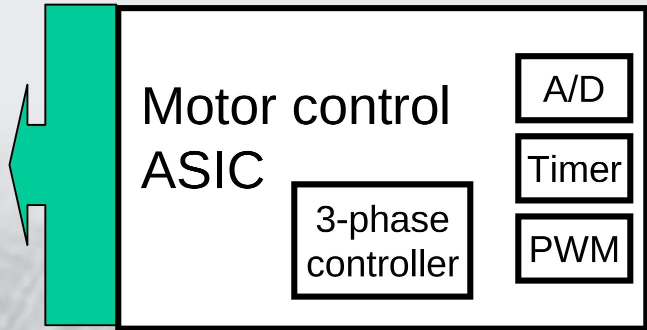
SE



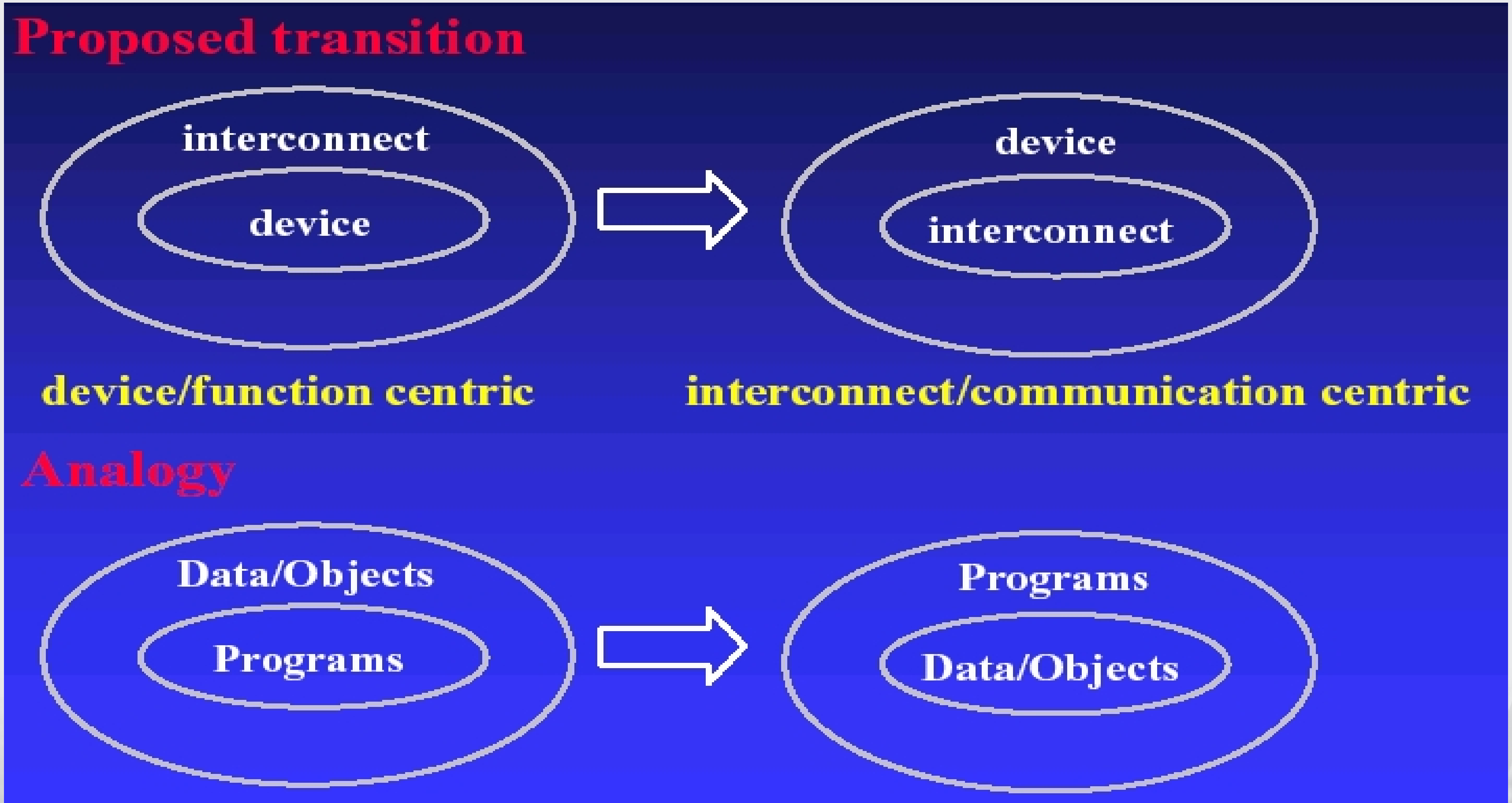
PE



I / O



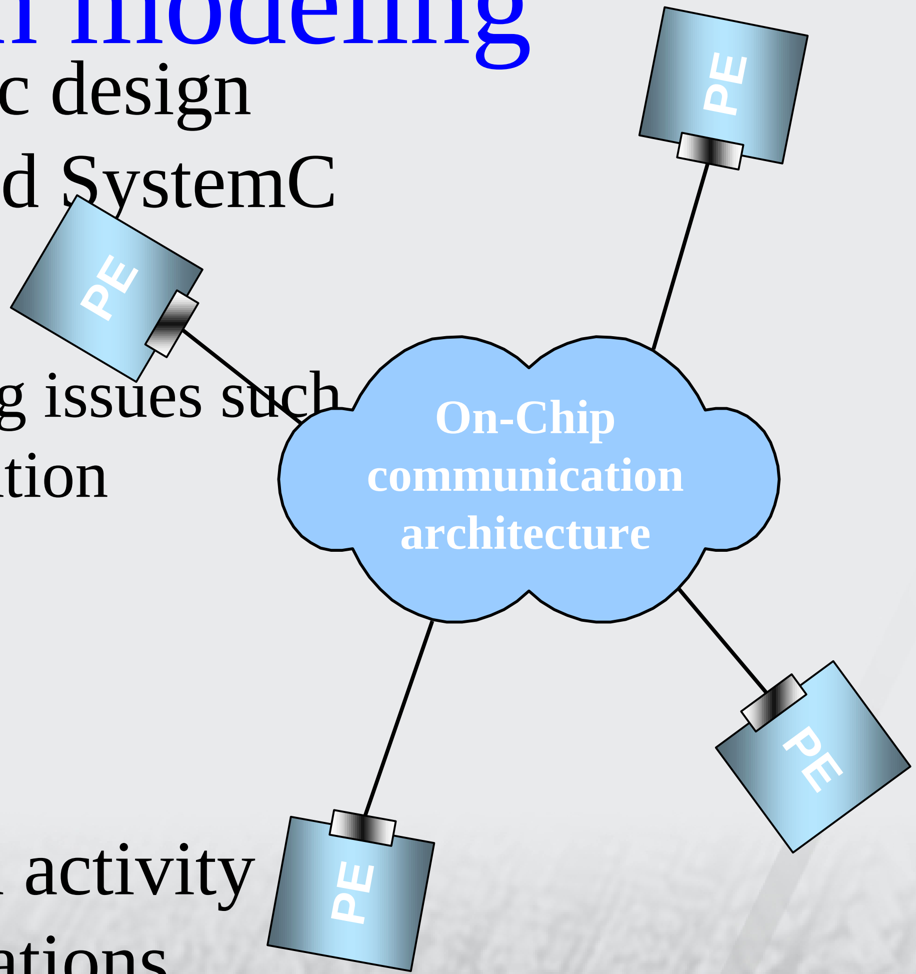
Communication Centric Methodology



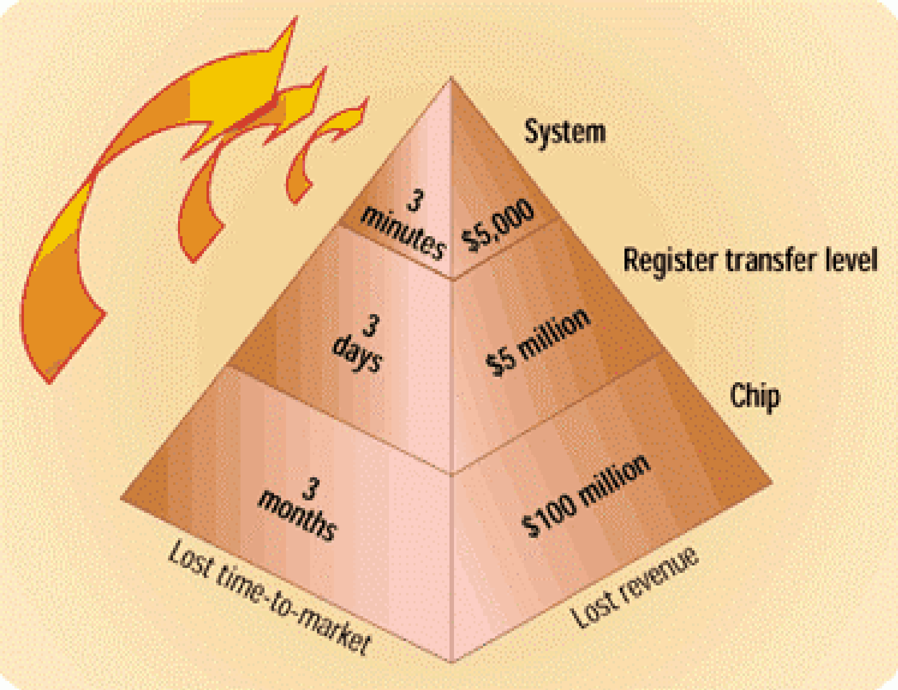
MEDEA+

OCCN : methodology for communication modeling

- Generic communication-centric design methodology based on C++ and SystemC
- OCCN addresses
  - high level performance modeling issues such speed, latency and power estimation
  - modeling productivity
  - model portability
  - simulation speed-up
- OCCN is an on-going research activity between several R&D organizations

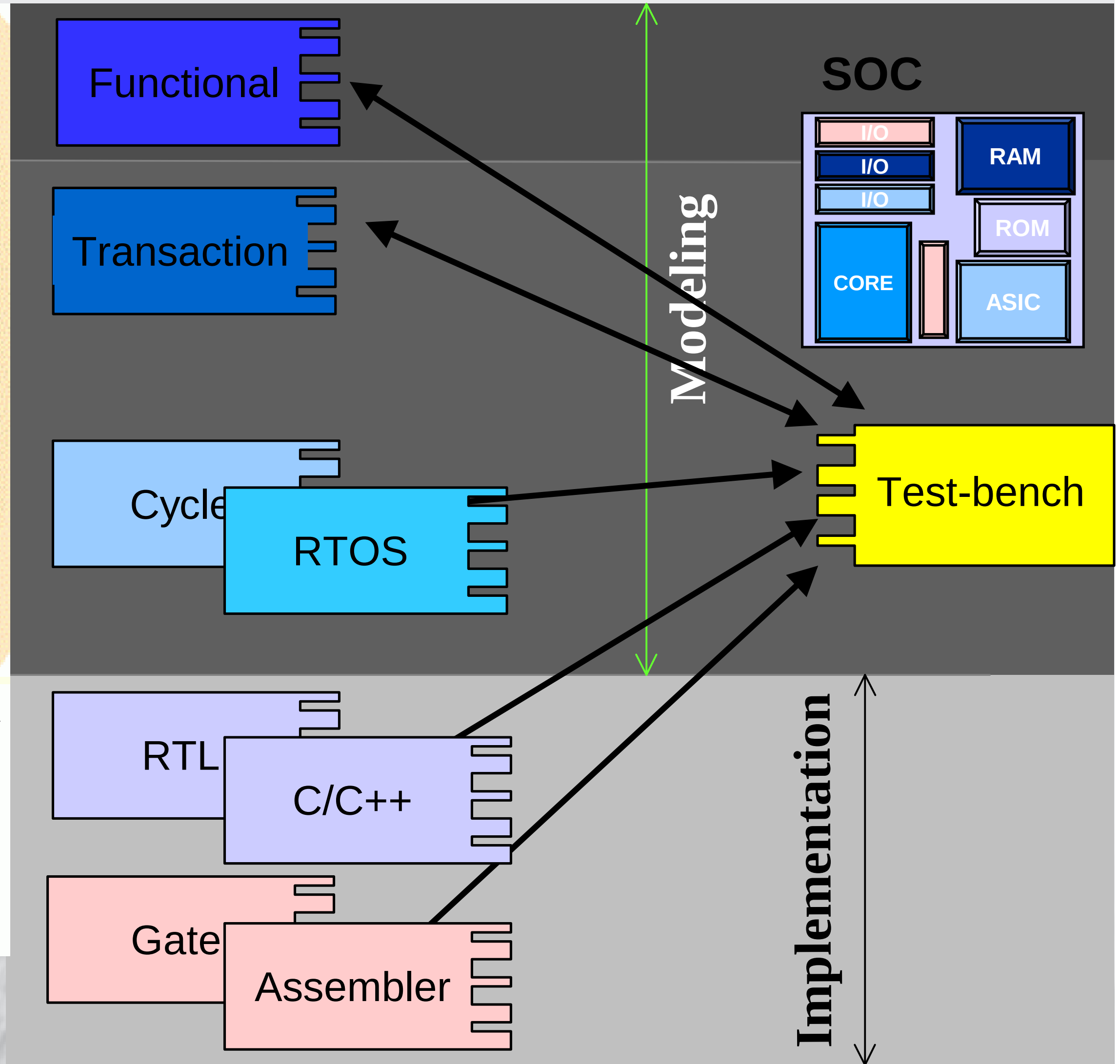


Compromise: Multi-levels Validation

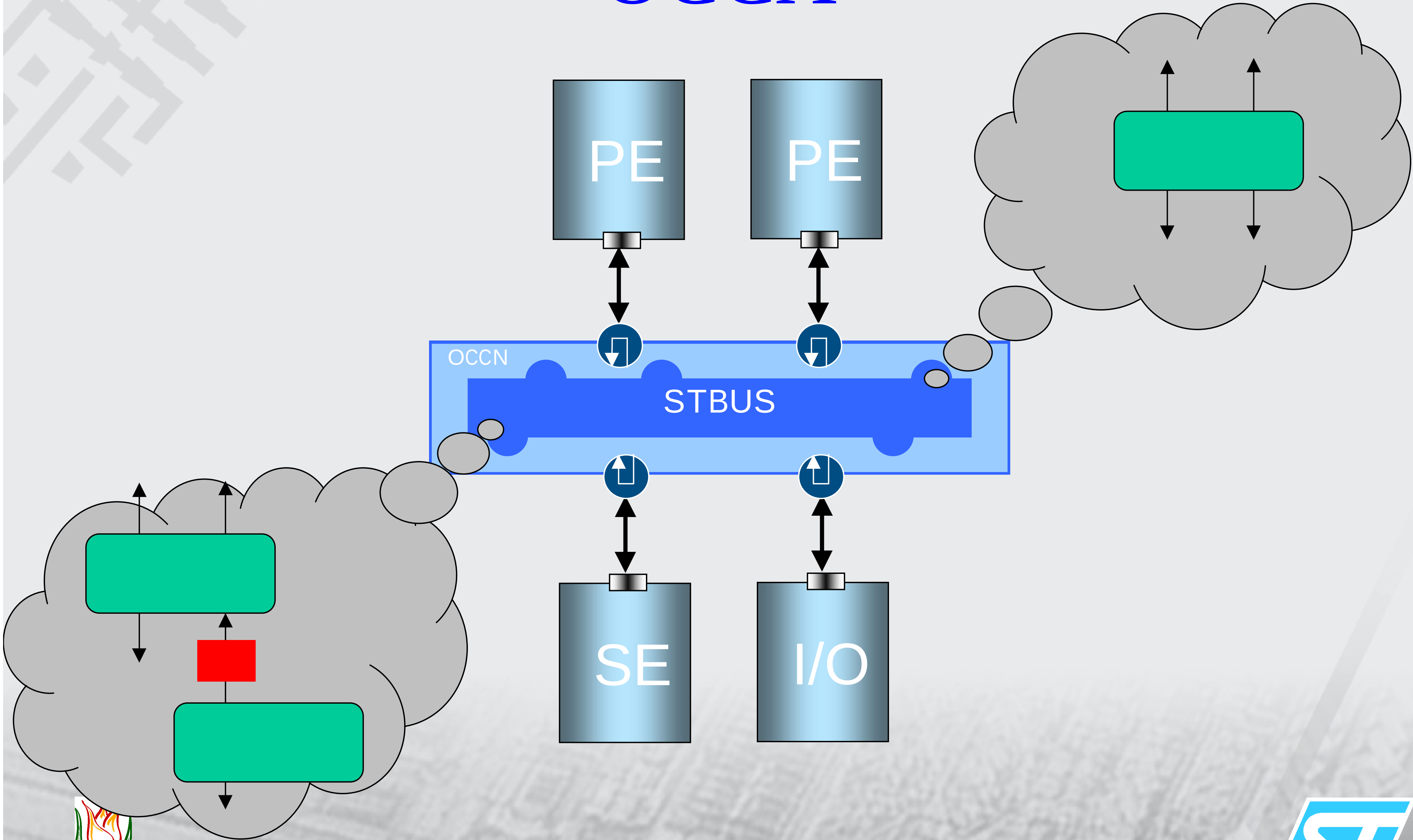


Source: Integrated Communications Design May, 2001

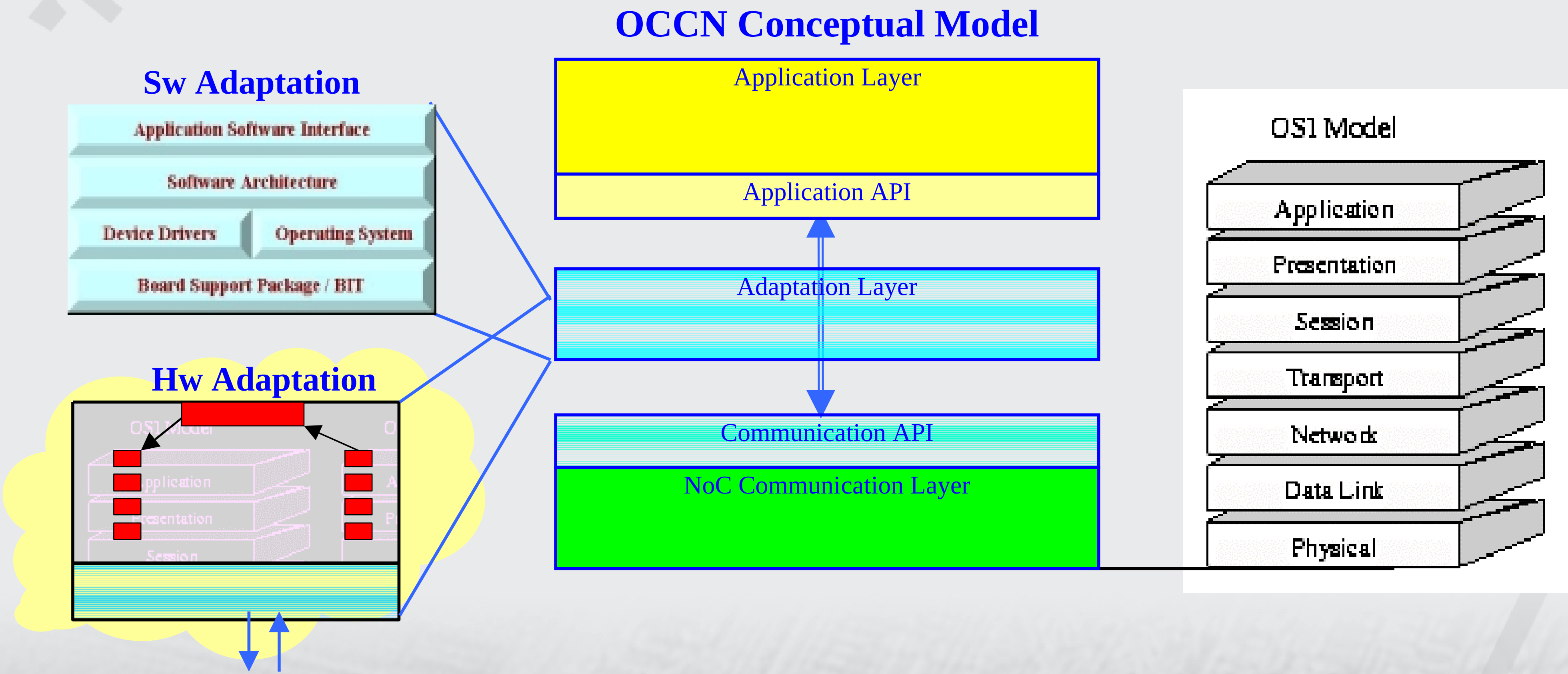
Higher Abstraction layer implies shorter Iteration Cycles and less Lost Revenue



OCCA

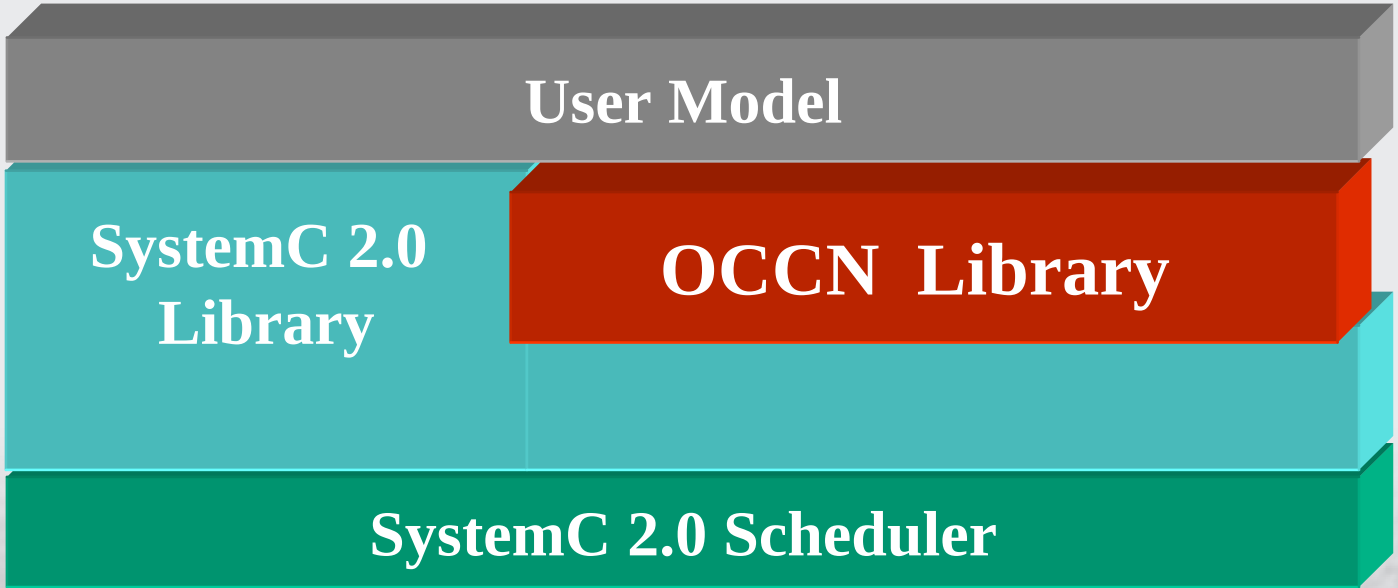


OCCN Conceptual Model



What is OCCN ?

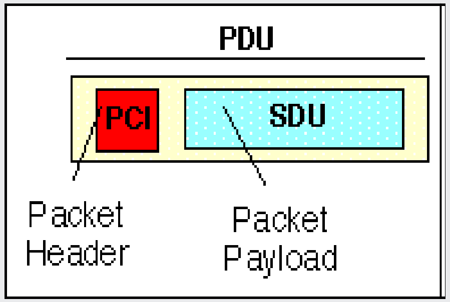
OCCN aims at IC modeling, providing a real **object-oriented methodology** based on a **C++ library** and a fully documented design flow **based on SystemC 2.0**

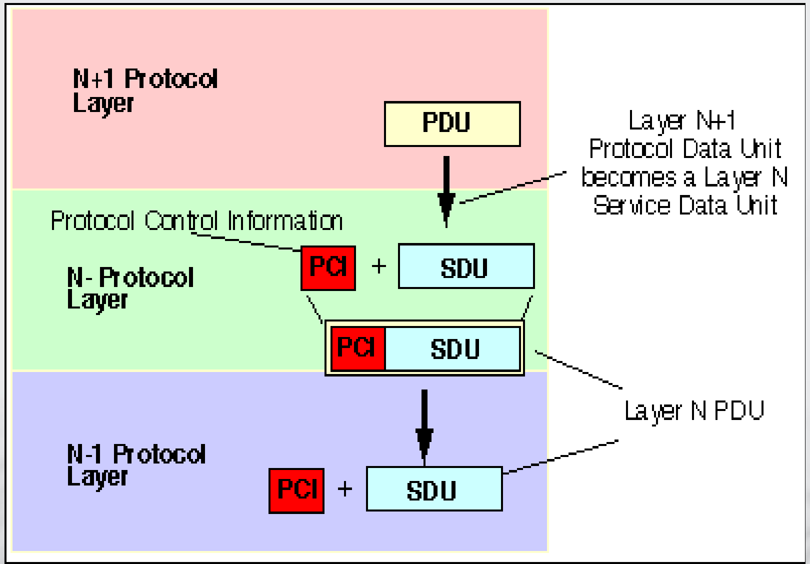


MEDEA+

OCCN core: the PDU

- Protocol = syntax + semantics
  - syntax = PDU
  - semantics = how the PDU are exchanged
- The PDUs exchanged have two parts:
  - a header also known as the Protocol Control Information (PCI)
  - a payload also known as a Service Data Unit (SDU)
- Several operators are defined for handling protocol operations





PDU Examples

8 bits

Pdu<char> p1;

P	T	Data

Struct DSLINK_token {unsigned int P:1; unsigned int T:1};

Pdu<DSLINK_token,char> p2;

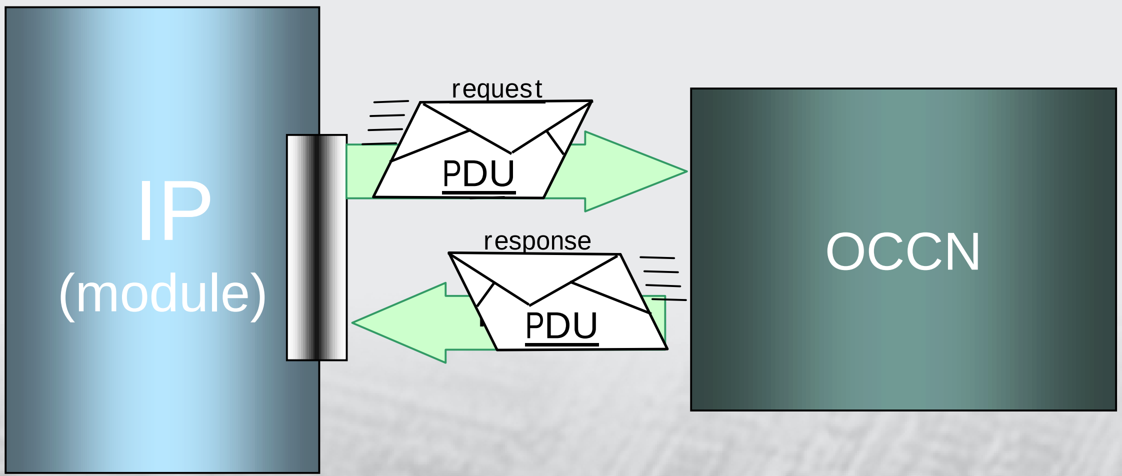
occn_hdr(pk1,P)=1;

pk1=‘a’;

Generic representation of a connection

- Any connection of a module to the communication node (network) is based on 2 sets of PDU
  - Pdu<uint32,PCIRequest>
  - Pdu<uint32,PCIResponse>

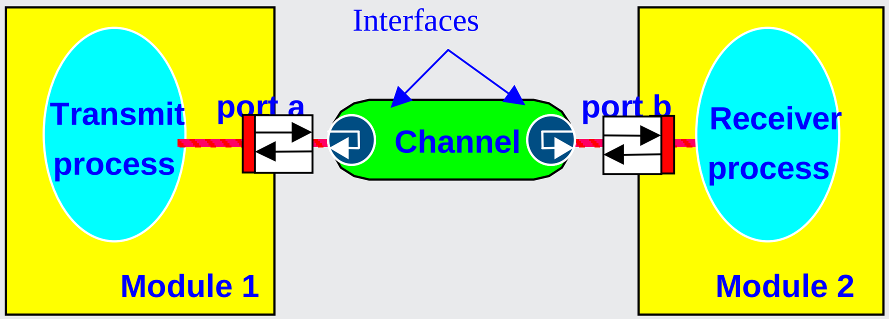
- The PCI sets are described thanks to C/C++ structures. They are defined according to the bus specification and thus are specific to a model. For instance it will be different for an AHB model and an STBUS model




```
Struct
{
  bool Request;
  unsigned char Opcode;
  bool Lock;
}
PCIControl
```

OCCN: communication

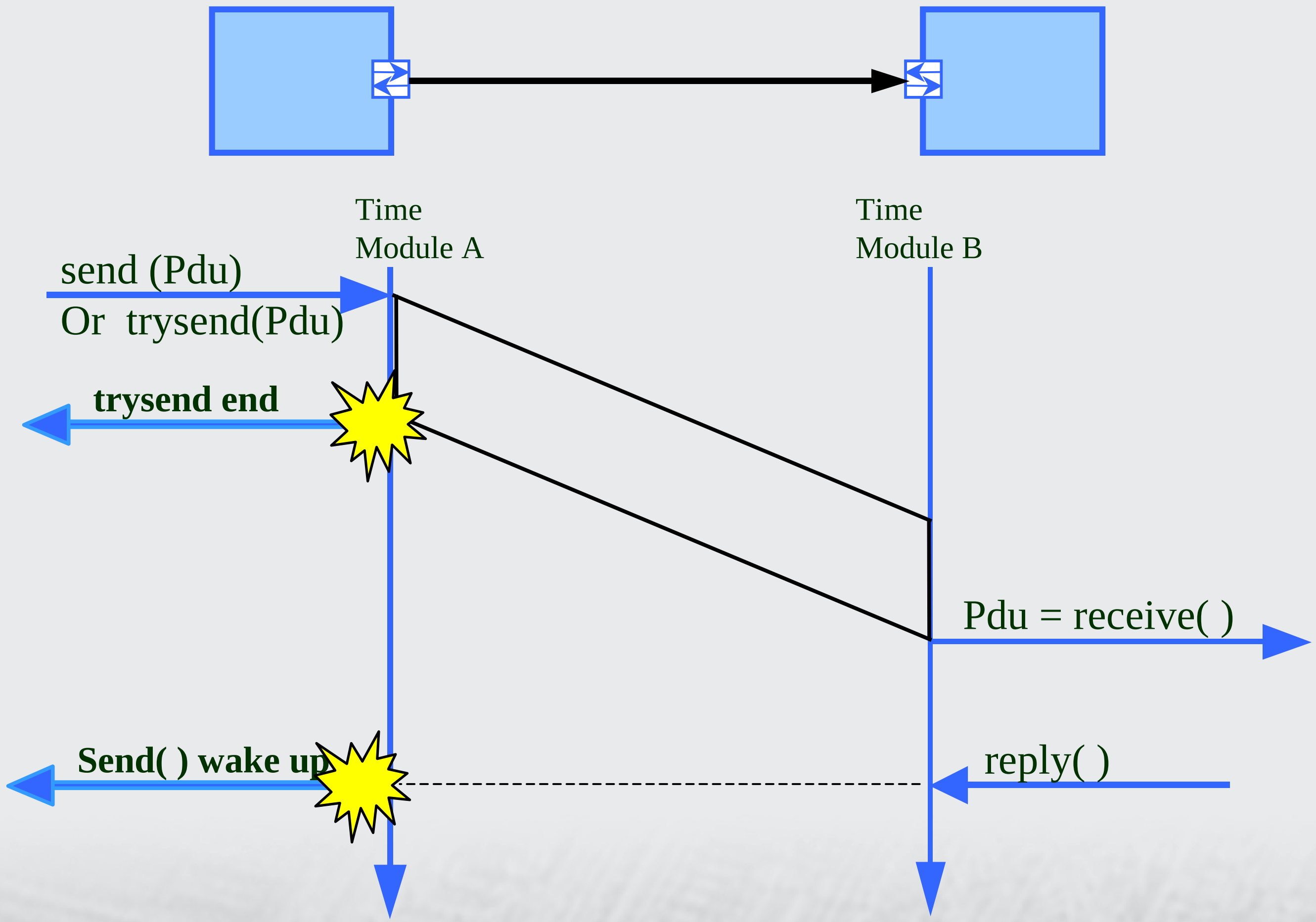
- SystemC based
- Simple Message Passing API



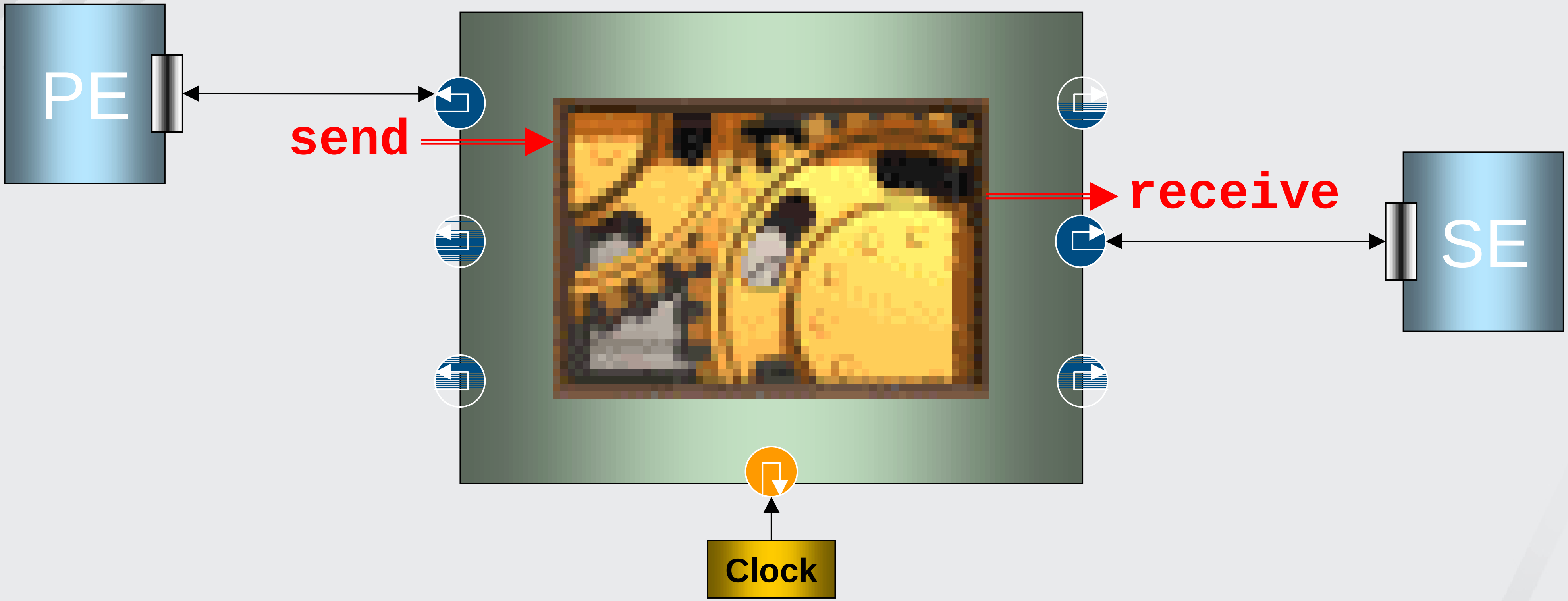

```
• Pdu<…>* send(Pdu<…>* p, sc_time& time_out=-1);
• int trysend(Pdu<…>* p);
• Pdu<…>* receive(int ack_time, sc_time& time_out=-1);
• Pdu<…>* receive(sc_time& ack_time, sc_time& time_out=-1);
• Pdu<…>* receive(sc_time& time_out=-1);
• void reply(Pdu<…>* p=0);
```

MEDEA+

OCCN core : API semantic



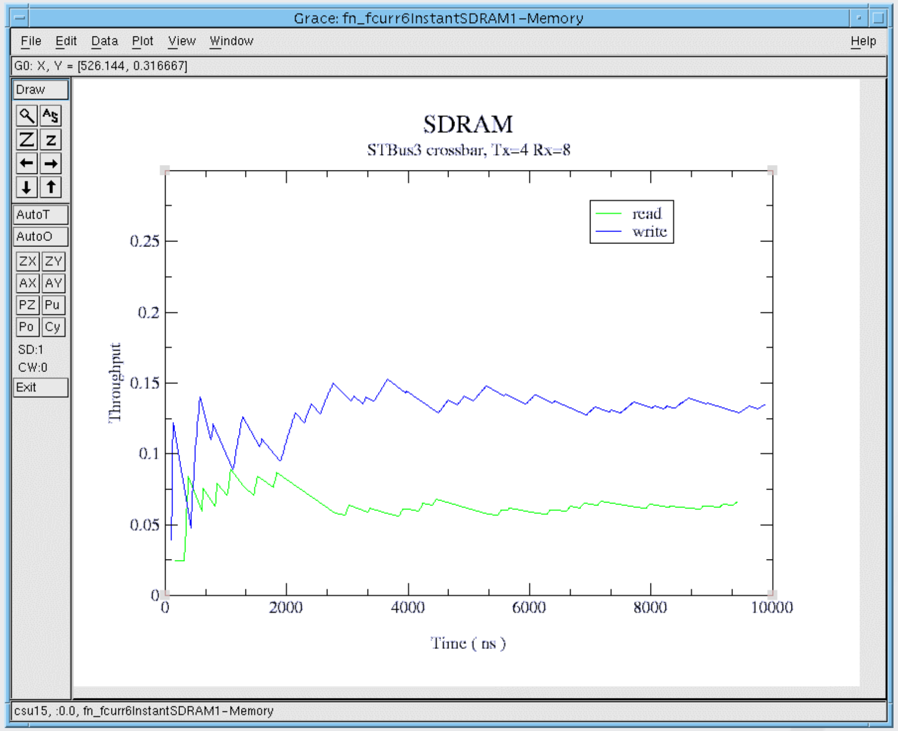
OCCN core : protocol state machine centralized



- allows synchronous and asynchronous communication modeling
- For synchronous com, PEs don't need to be connected to the clock signal

Performance measurement with Grace

- XY graph, XY charts, pie charts, polar, and fixed graphs.
- User-defined scaling, ticks, labels, symbols, line styles, fonts, colors.
- Merging, validation, cumulative average, curve fitting, regression, filtering, DFT/FFT, cross/auto-correlation, sorting, interpolation, integration, differentiation...
- Internal language, and dynamic module loading (C, Fortran, etc).
- Hardcopy support with PS, PDF, GIF and PNM formats.



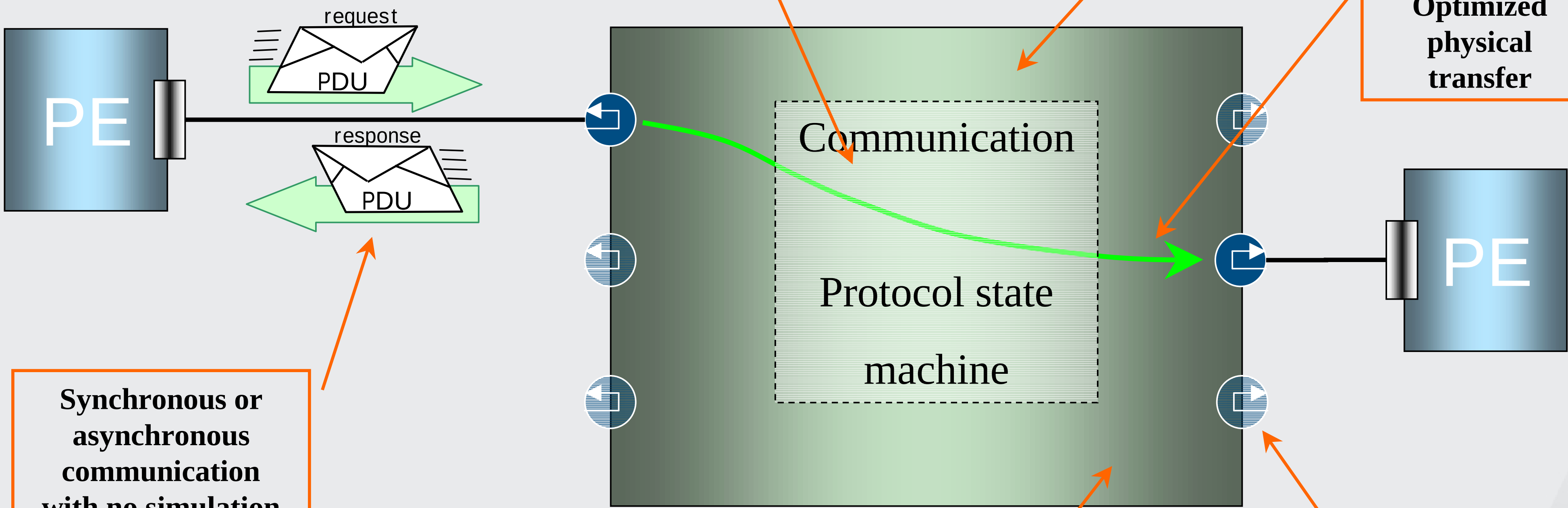
OCCN framework keypoints

Message passing oriented communication

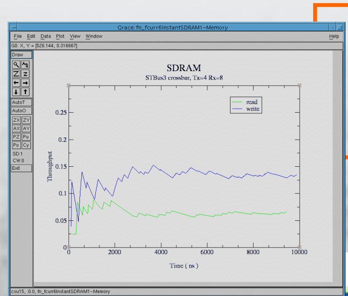
Protocol state machine centralized

Generic structure for communication node

Optimized physical transfer



Synchronous or asynchronous communication with no simulation overhead



System performance metrics

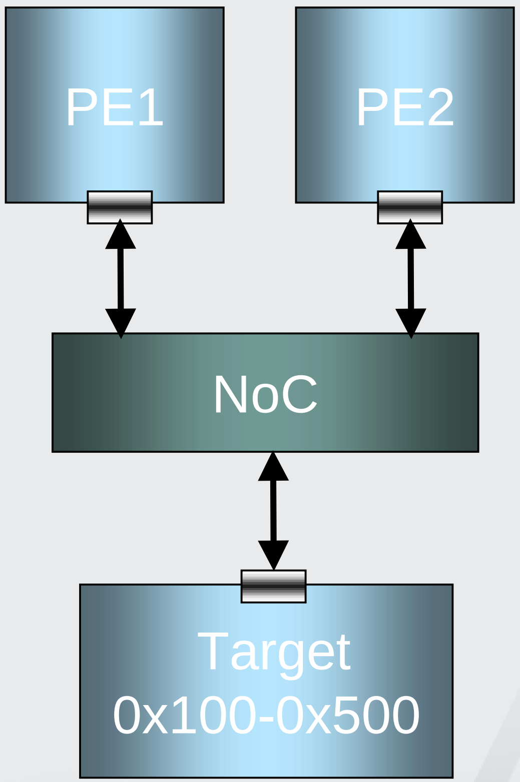
Dynamic number of bound PE

MP SoC architecure

```
main()
{
        sc_clock my_clock(10, SC_NS);
        PE pe1, pe2;
        SE se;
        NoC occa();


        occa.clk(my_clock);
        pe1.port(occa);
        pe2.port(occa);
        se.port(occa);

        occa.set_address_range(&se1.port,0x100,0x500);
        occa.set_priority(&pe1.port, 2);
        occa.set_priority(&pe2.port, 5);
        sc_start(-1);
}
```



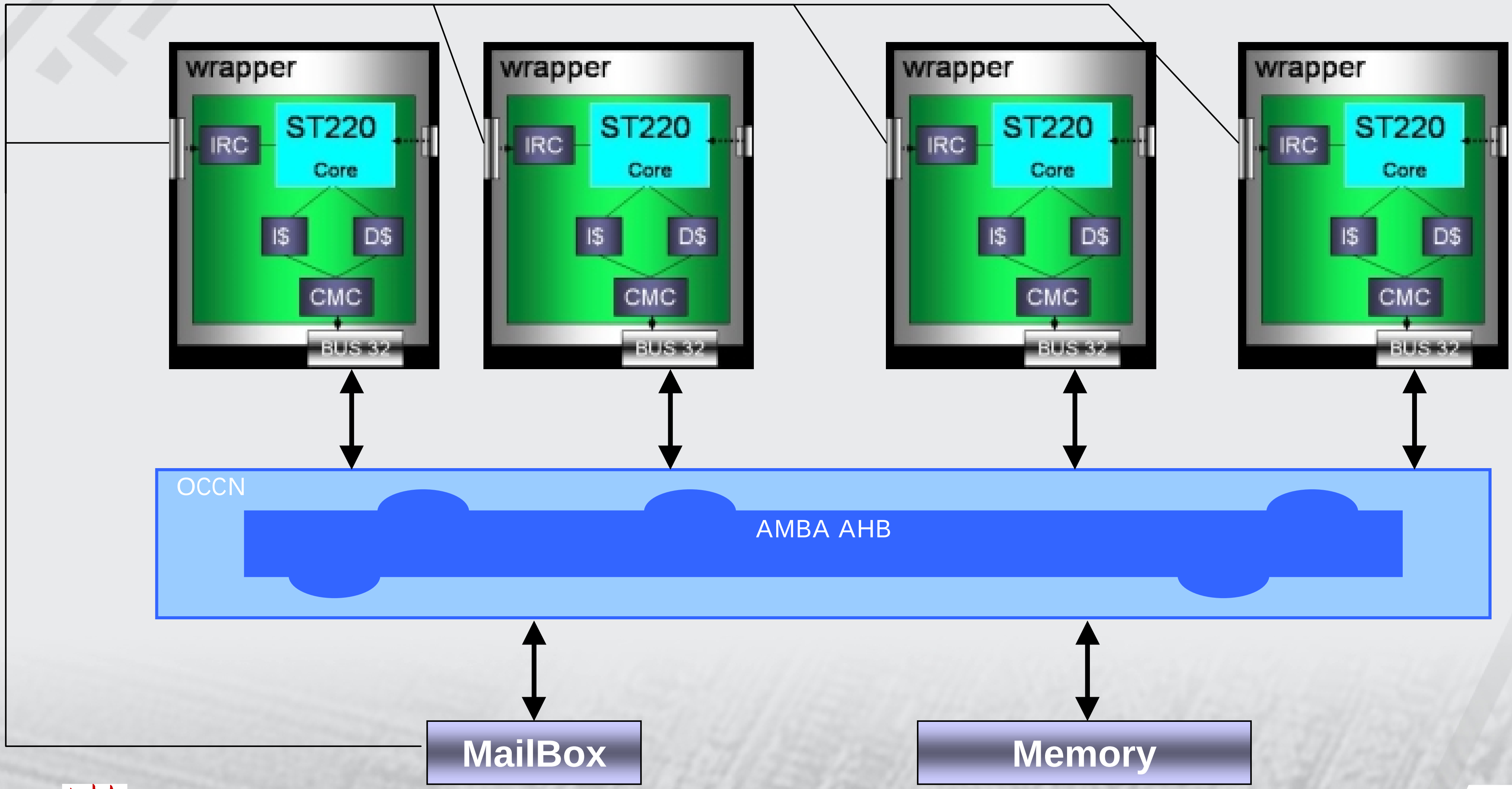
OCCN: PE code example

```
#include "producer.h"
producer::producer(sc_module_name name) : sc_module(name)
{SC_THREAD(read);}


void producer::read() {
  char c;
  Pdu<char>* msg;
  while (cin.get(c)) {
        msg = new Pdu<char>;
        // producer sends c
       *msg = c;
        out.send(msg);
  } // after the send the msg is not usable
}
```

Protocol inlining:
protocol is automatic generated

Case Study: NoC Platform



MEDEA+

Some preliminary numbers

- We are able to boot linux
  - on a 450Mhz machine
  - 7 millions of bundles
  - Without cache,bus and memory waiting times, we got 3 minutes
  - Without cache and using TLM CA bus, we got 10 minutes
- Expectation on a linux machine 3 minutes

Conclusion 1/2

- OCCN
  - based on SystemC methodology
  - open & flexible API
  - simulation speed-up
  - reusability
  - productivity
  - communication architecure exploration
- Similar work: Gigascale Silicon Research Center (GSRC) effort Princeton University: MESCAL Project

  Modern Embedded Systems Compilers Architectures and Languages

  Princeton and UC Berkeley

Conclusion 2/2

- Research Activity funded in Medea
- Public part -> http://occn.sourceforge.net

ACCENT
DESIGN TECHNOLOGY SERVICES

Univ. of Roma

Univ. of Bologna

ISD S.A
Integrated Systems Development

Univ. of Ancona

MEDEA+

Thank You and …