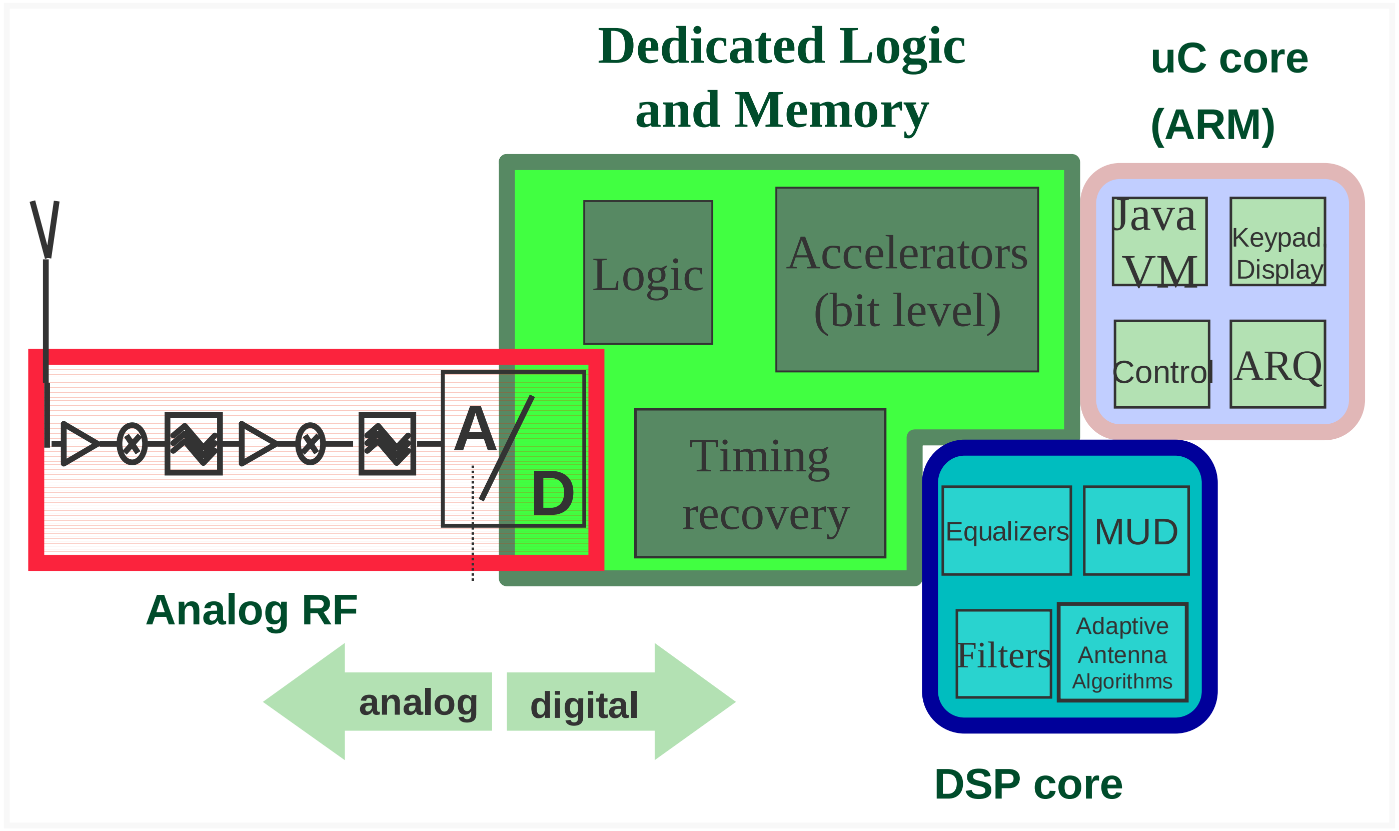
Interface-Based Design
Introduction

A. Richard Newton

Department of Electrical Engineering and Computer Sciences

University of California at Berkeley

Integrated CMOS Radio



Integrate within the same chip very diverse system functions like: wireless channel control, signal processing, codec algorithms, radio modems, RF transceivers… and implement them using a heterogeneous architecture (J. Rabaey)

Communication versus Computation

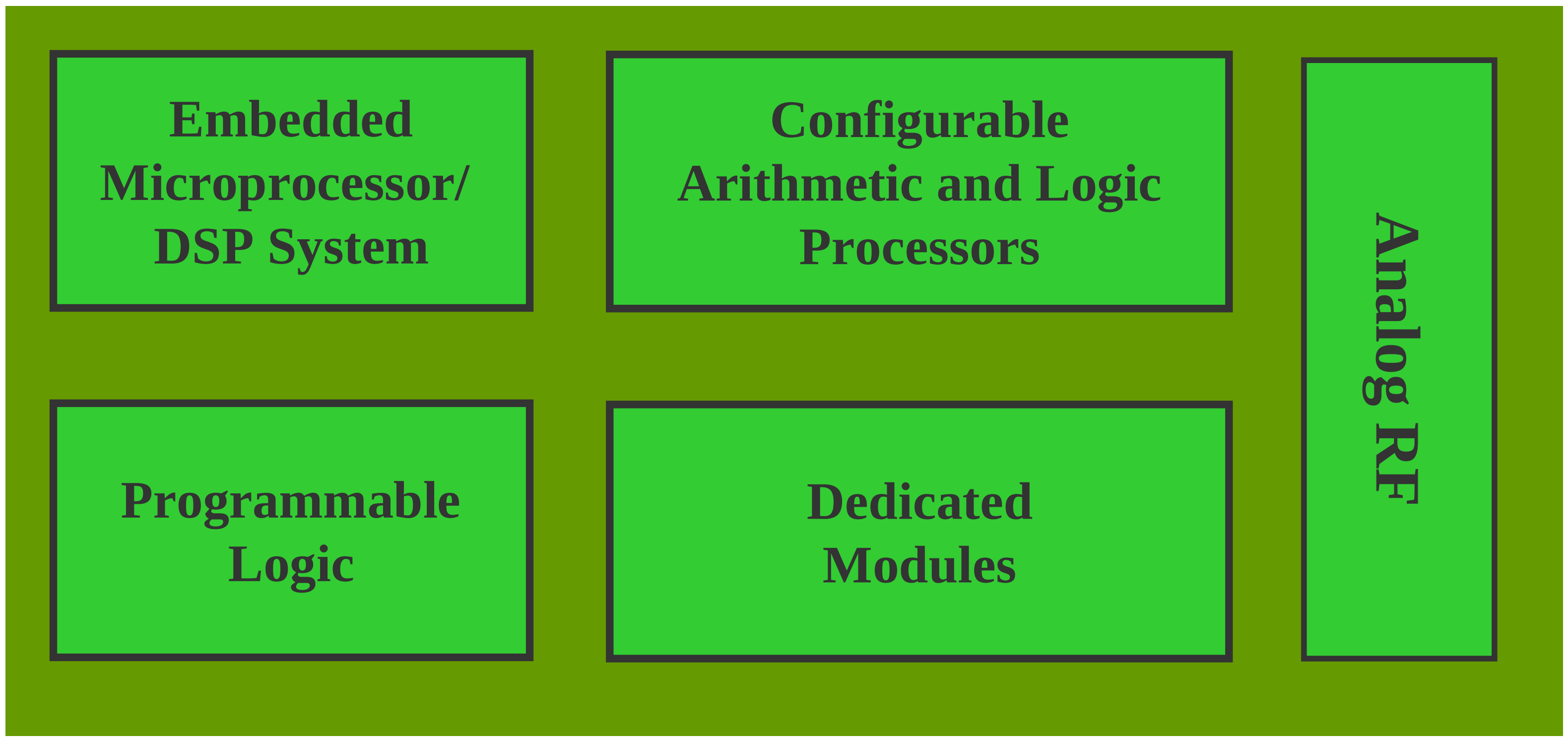
- Computation cost (2004): 60 pJ/operation (assuming continued scaling)
- Communication cost (minimum):
  - 100 m distance: 20 nJ/bit @ 1.5 GHz
  - 10 m distance: 2 pJ/bit @ 1.5 GHz
- Computation versus Communications
  - 100 m distance: 300 operations == 1bit
  - 10 m distance: 0.03 operation == 1bit

Computation/Communication requirements vary with distance, data type, and environment

(courtesy of J. Rabaey)

Energy-efficient Programmable Implementation Platform

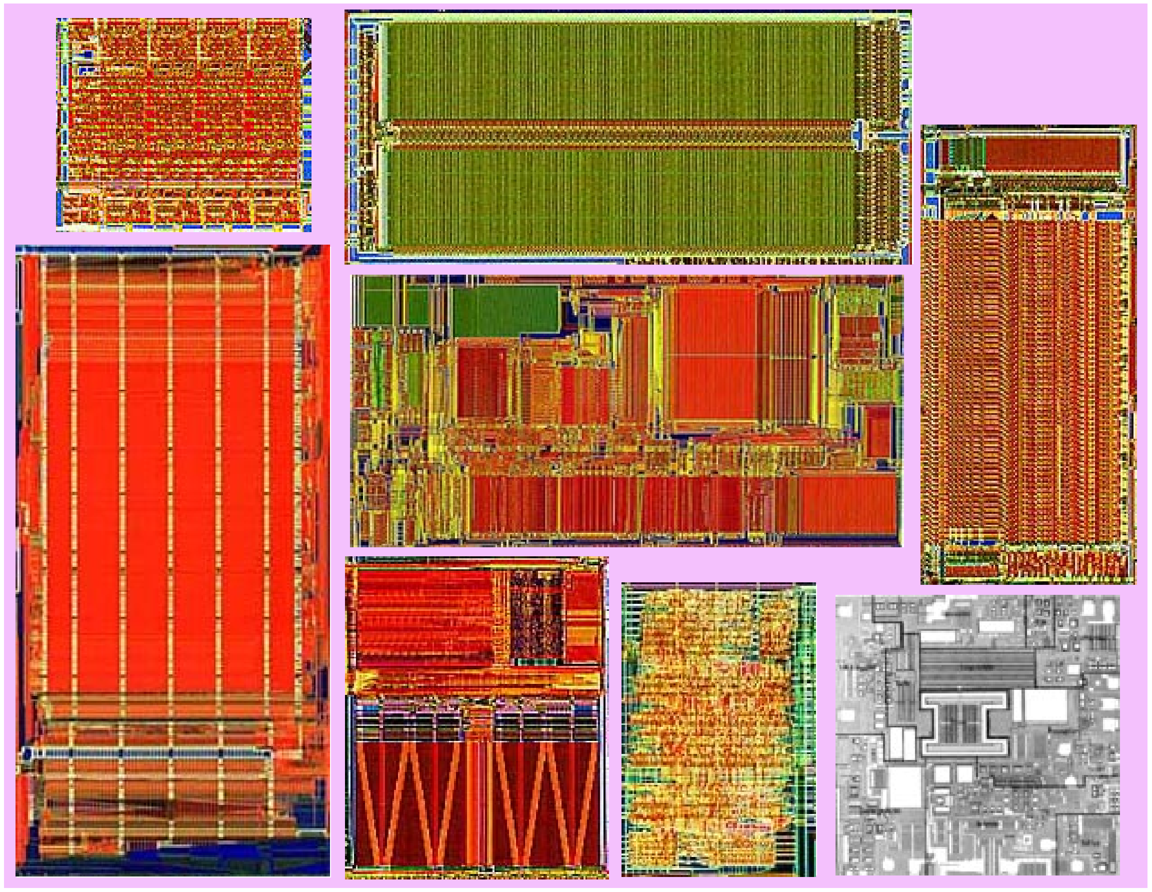
“Software-defined Radio”


Embedded Microprocessor/ DSP System
Configurable Arithmetic and Logic Processors
Analog RF
Programmable Logic
Dedicated Modules

Protocol Processing **Communication Channel**

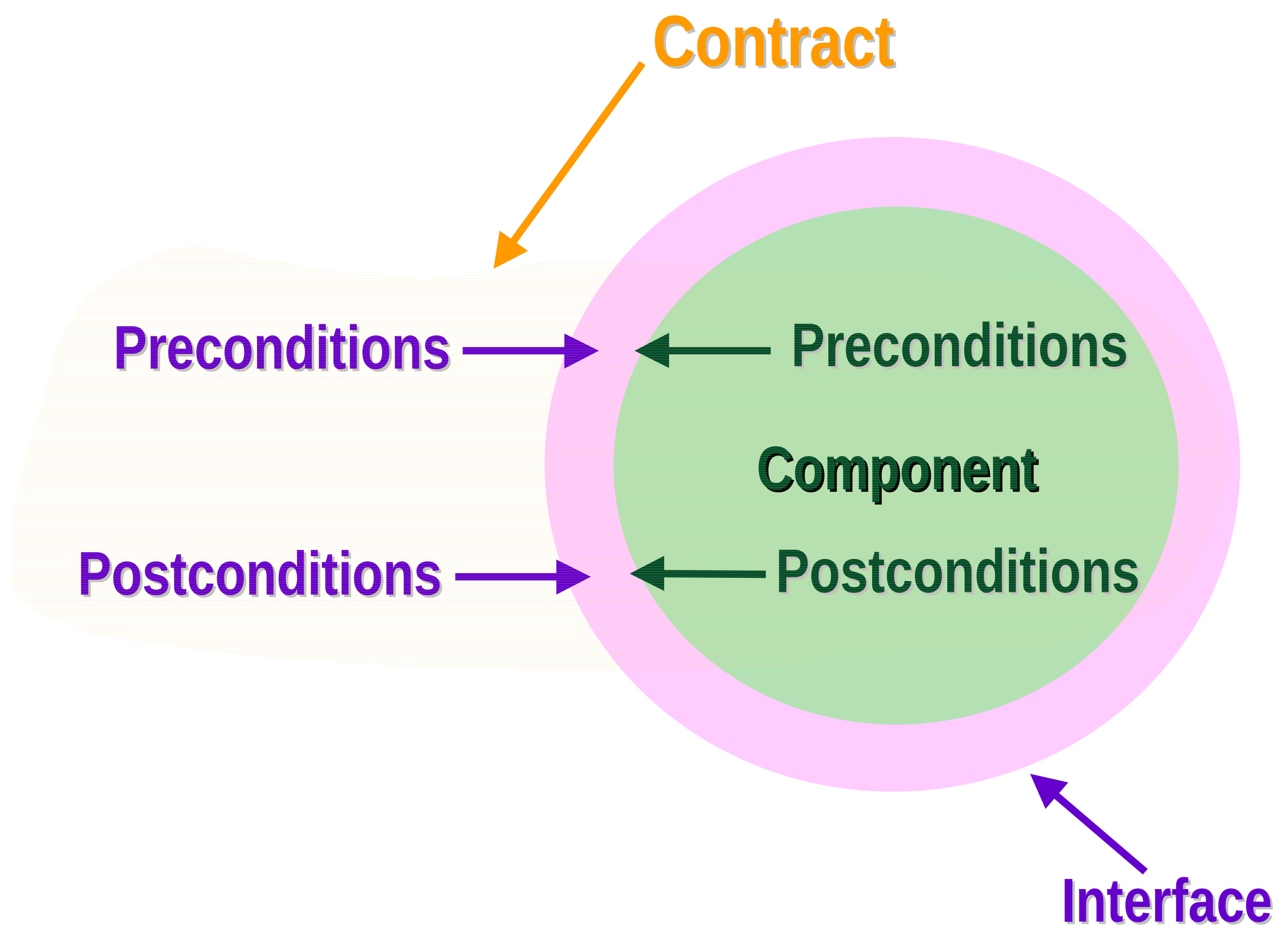
(courtesy of J. Rabaey)

The Design Object

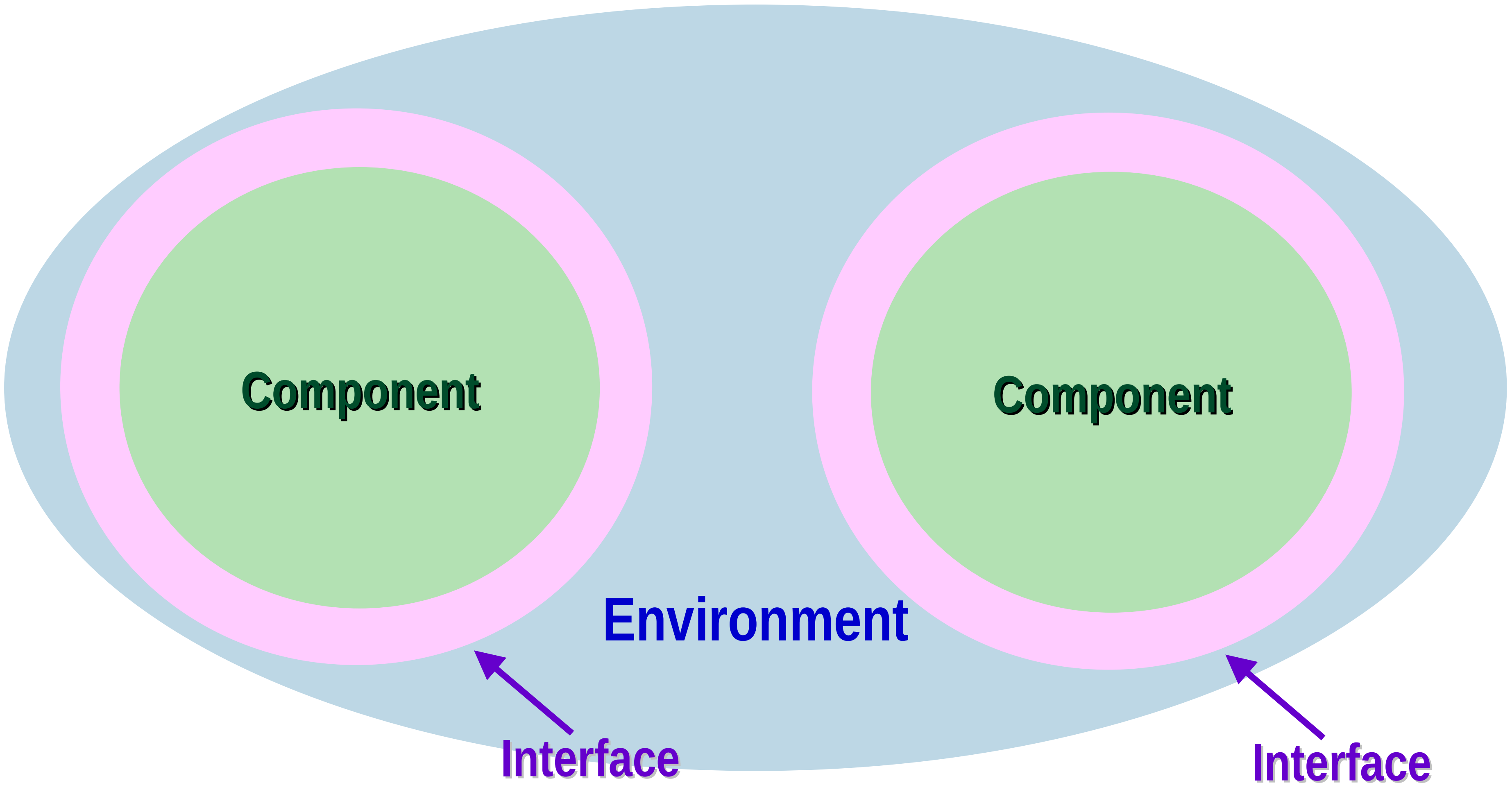


- Assemble Components from parameterized library
- Including:
  - Configurable processor core
  - Memories (RAM, ROM)
  - Special-purpose standard blocks (ASSPs)
  - Glue Logic
- Third-party special-purpose logic/MEMS/MEOS
- Integrate using standard approach to on-chip communication

Interfaces and Contracts



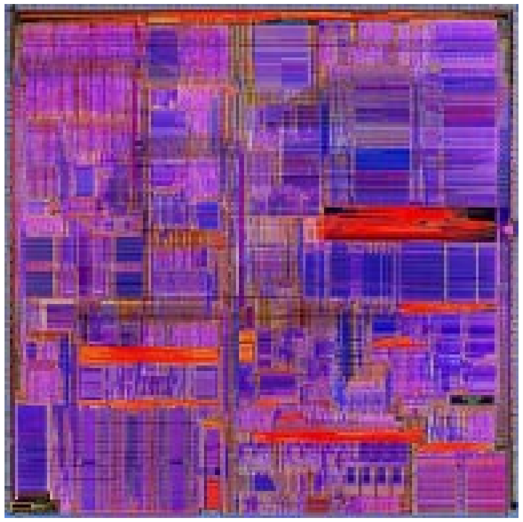
Interfaces and Contracts

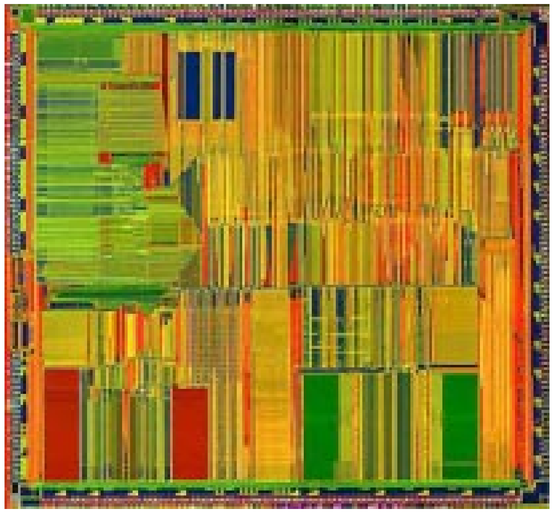


Interface: Levels of Abstraction Part 1: Mechanisms (Wiring)

- **Physical**: Geometrical arrangement of I/O locations, how to connect, etc.
- **Electrical:** Restrictions/requirements on currents, voltages, noise, risetimes, falltimes, etc.
- **Logic (Combinational):** Largely a transcoding (discretization) of electrical limits into logic domain
- **Sequential (Stateful):** form of the model: clocked synchronous, asynchronous (what?), etc.

High-End Systems


Intel Pentium Pro


Intel Pentium 2

IBM/Motorola PPC 620

High-End Systems

IBM/Motorola PPC 620

VSIA: Four Orthogonal Model Characteristics

- Temporal Detail
- Data Value Detail
- Functional Detail
- Structural Detail

VSIA: System-Level Data Abstractions

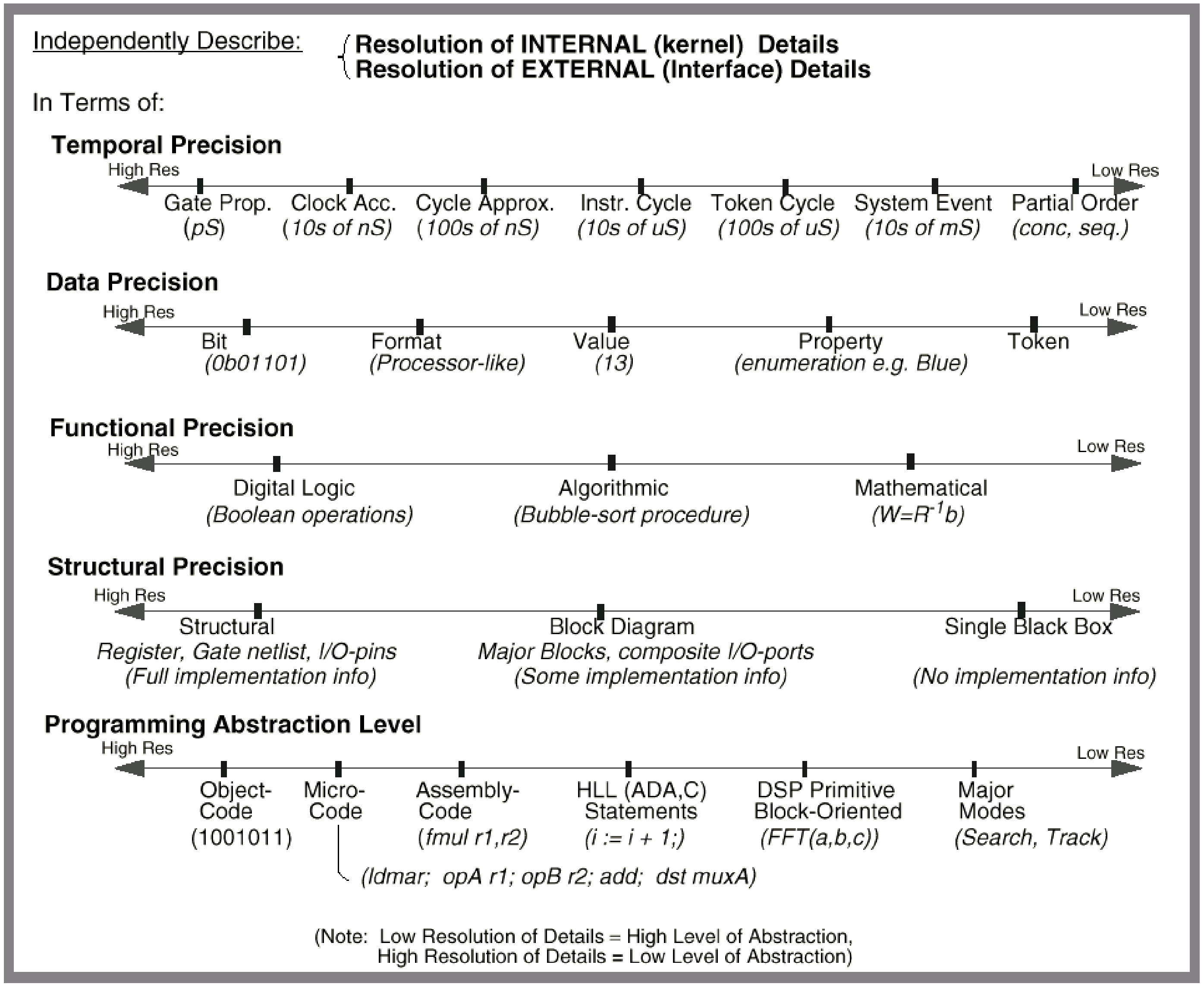

Independently Describe:
Resolution of INTERNAL (kernel) Details
Resolution of EXTERNAL (Interface) Details
In Terms of:
Temporal Precision
High Res
Low Res
Gate Prop. (pS)
Clock Acc. (10s of nS)
Cycle Approx. (100s of nS)
Instr. Cycle (10s of uS)
Token Cycle (100s of uS)
System Event (10s of mS)
Partial Order (conc, seq.)
Data Precision
High Res
Low Res
Bit (0b01101)
Format (Processor-like)
Value (13)
Property (enumeration e.g. Blue)
Token
Functional Precision
High Res
Low Res
Digital Logic (Boolean operations)
Algorithmic (Bubble-sort procedure)
Mathematical (W=R⁻¹b)
Structural Precision
High Res
Low Res
Structural
Register, Gate netlist, I/O-pins
(Full implementation info)
Block Diagram
Major Blocks, composite I/O-ports
(Some implementation info)
Single Black Box
(No implementation info)
Programming Abstraction Level
High Res
Low Res
Object-Code (1001011)
Micro-Code
(ldmar; opA r1; opB r2; add; dst muxA)
Assembly-Code (fmul r1,r2)
HLL (ADA,C) Statements (i := i + 1;)
DSP Primitive Block-Oriented (FFT(a,b,c))
Major Modes (Search, Track)
(Note: Low Resolution of Details = High Level of Abstraction,
High Resolution of Details = Low Level of Abstraction)

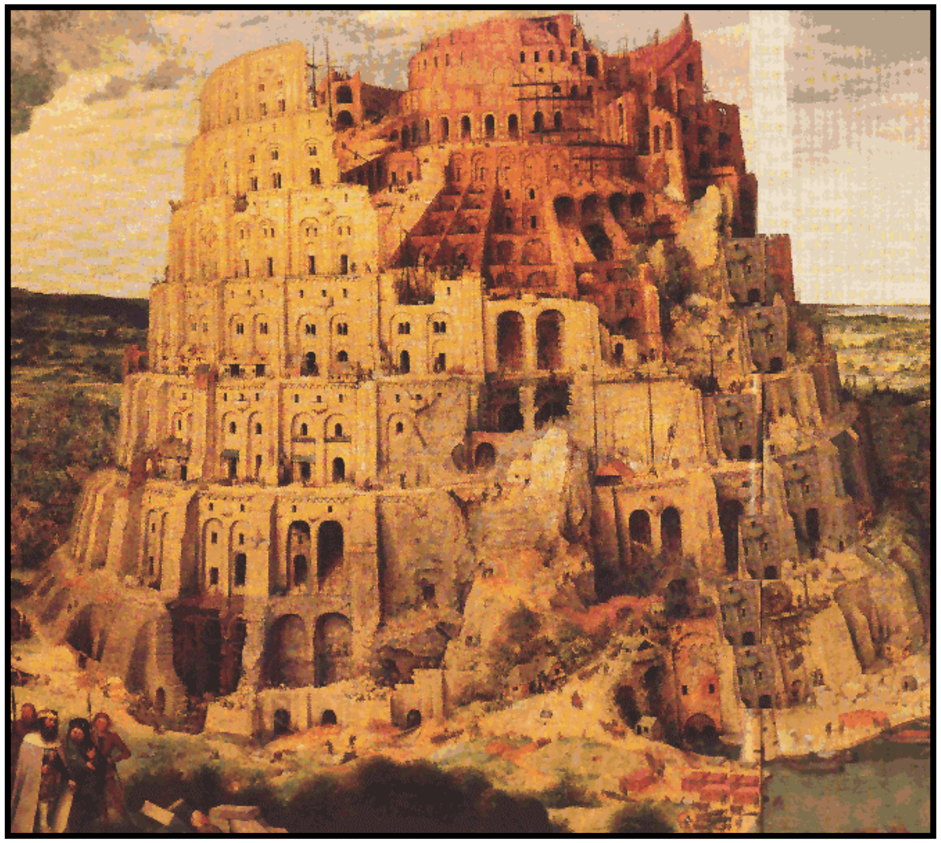
Figure 1 - New Taxonomy Axes

Interface: Levels of Abstraction
Part 2: Policies (Semantics?)

- Legal Data Types and Abstract Types
- "Protocols"
  - Local (handshakes)
  - Global (clocked synchronous)
- Transaction Models
- Implications of Concurrency

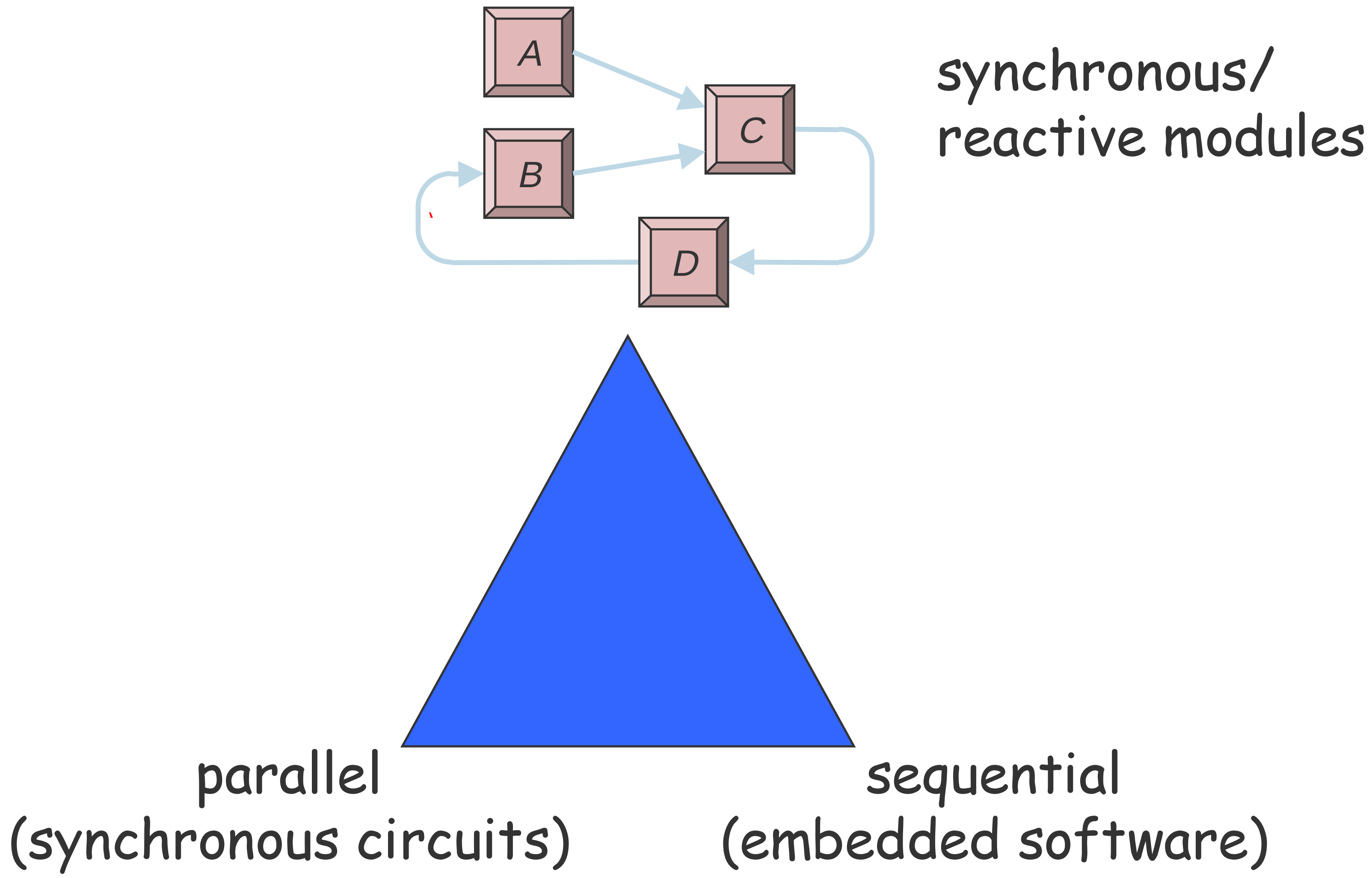
Interactions of Components

- Procedures
- Synchronous logic
- Asynchronous logic
- Bus protocols
- Shared memory
- Semaphores
- Rendezvous
- Timed events
- Streams
- Message passing
- Communication protocols/handshaking


Tower of Babel, in painting by Bruegel, 1563

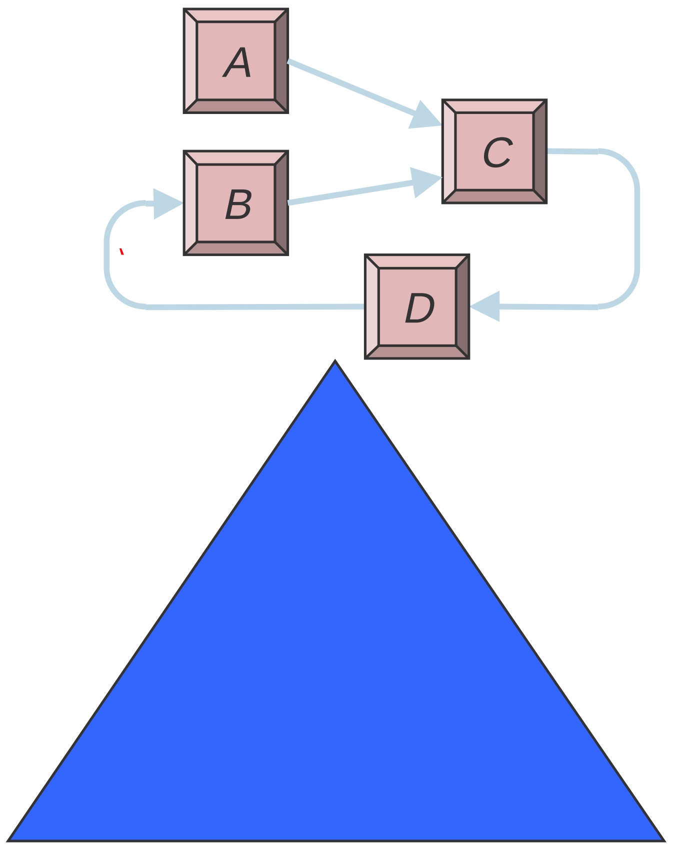
Source: Prof. Edward Lee

Abstracting Synchrony



Source: Prof. Edward Lee

Abstracting Rendezvous



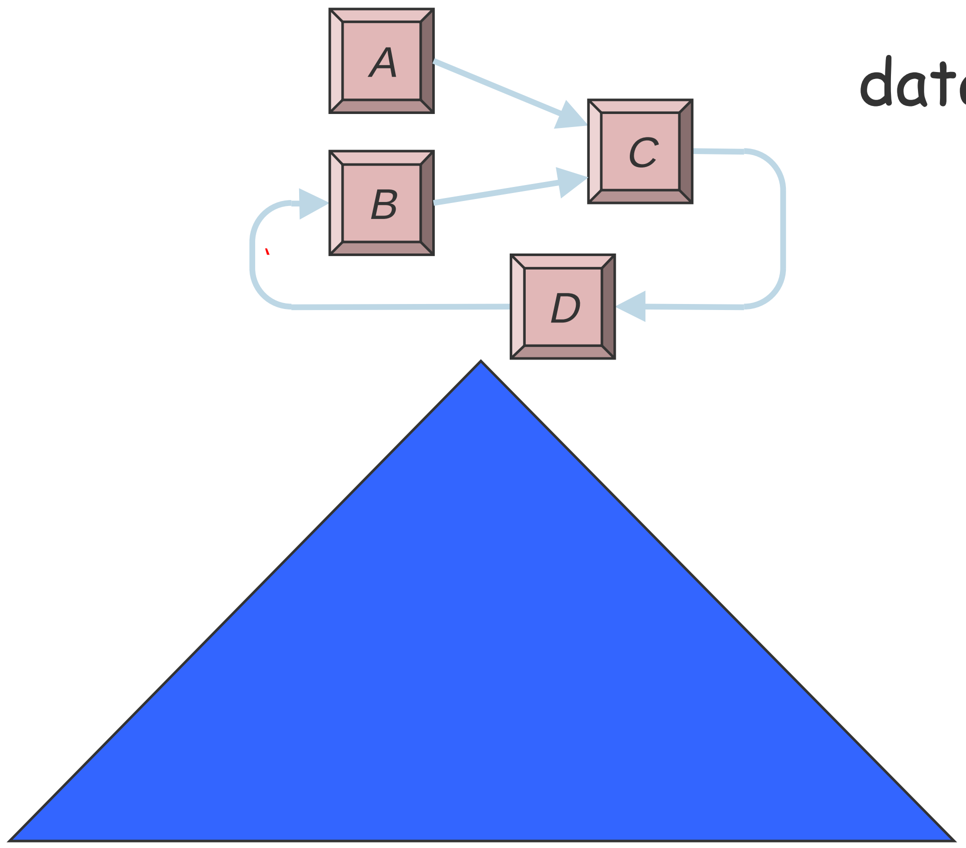
communicating sequential processes

parallel
(circuits with handshaking)

sequential
(software with threads)

Source: Prof. Edward Lee

Abstracting Message Passing



dataflow modules

parallel
(asynchronous circuits)

sequential
(software multitasking)

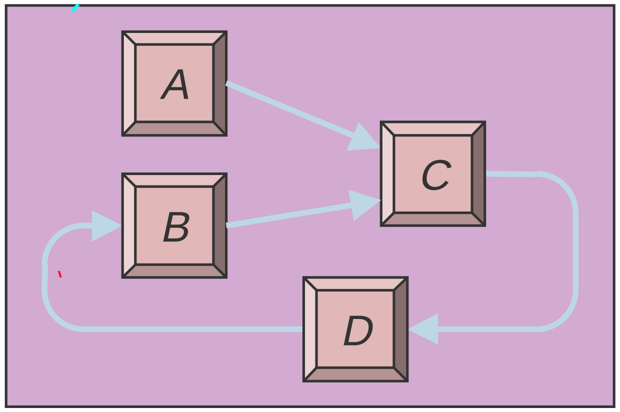
Source: Prof. Edward Lee

It's all about Communication!
Then *must* be all about Concurrency...

Useful Concurrent Semantics

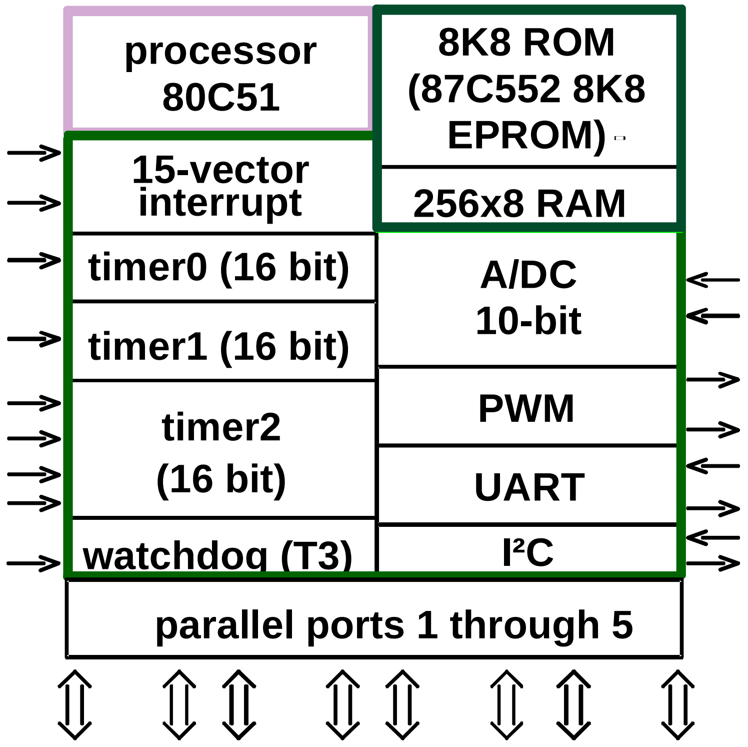
- Analog computers (ODEs)
- Spatial/temporal models (PDEs)
- Discrete time (difference equations)
- Discrete-event systems (DE)
- Synchronous-reactive systems (SR)
- Sequential processes with rendezvous (CSP)
- Process networks (Kahn)
- Dataflow (Dennis)

Block diagrams often provide a nice syntax for concurrent semantics



Source: Prof. Edward Lee

A Complete System-on-a-Chip



- complete system
- timers, PWM for control
- I²C-bus and par./ser. interfaces for communi-cation
- A/D converter
- watchdog (SW activity timeout): safety
- on-chip memory
- interrupt controller

Philips 83 C552: 8 bit-8051 based microcontroller

control dominated systems

Source: Prof. Rolf Ernst

Architectures for Higher Computation Requirements

Example: Motorola MC 683xx - family of controllers

Processor: CPU 32

- **68000 - processor enhanced by most of the 68030 features**
- **CISC processor: code density**
- **pipelining**
- **standard register sets (not in RAM)**
  $\Longrightarrow$ **context switch is more expensive**
- **virtual memory**
- **supervisor and user modes**

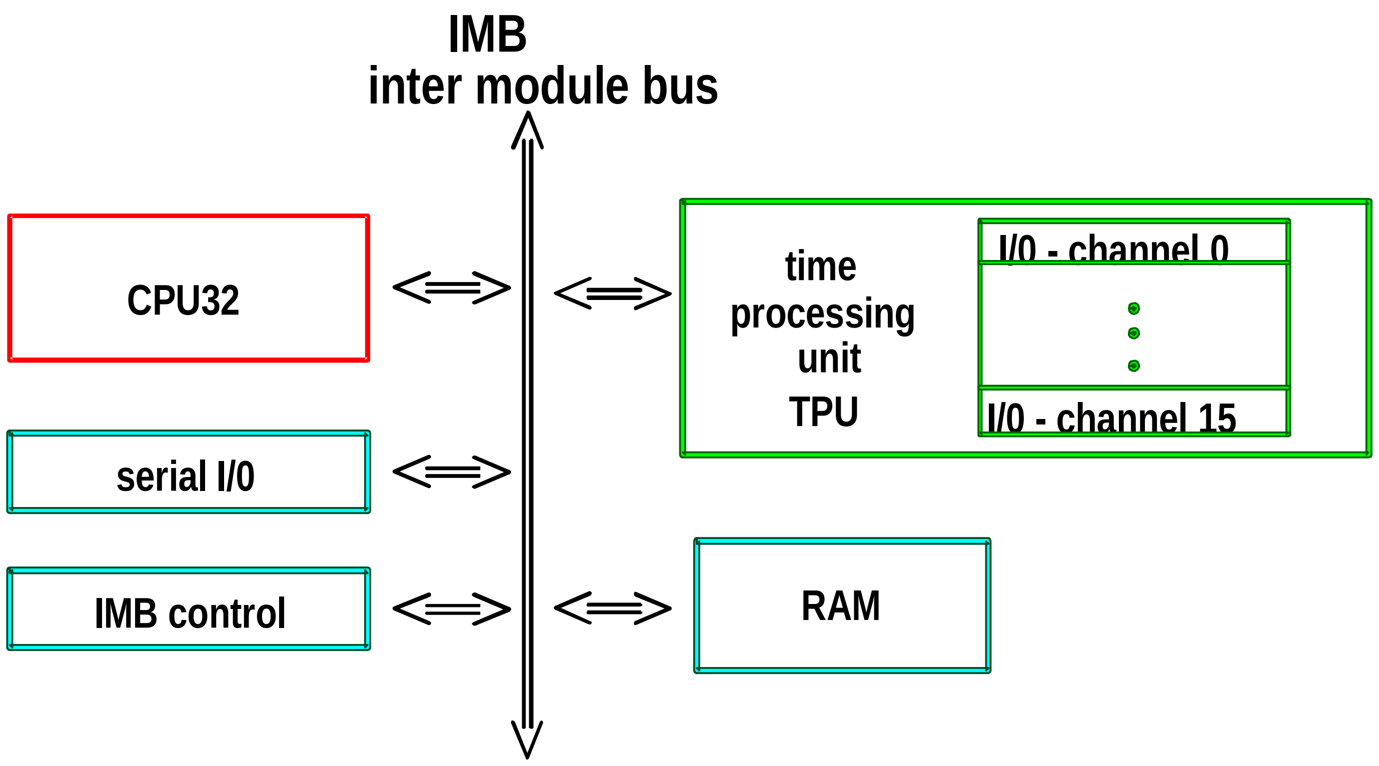
$\left.\begin{array}{l}\text{virtual memory}\\ \text{supervisor and user modes}\end{array}\right\}$ **aims at use of operating systems**

- table lookup instructions for compressed tables with built-in linear interpolation
  $\Longrightarrow$ **data density is concern**

control dominated systems

Source: Prof. Rolf Ernst

MC68332



Designed for automotive applications with mixture of computation intensive tasks and complex I/0 -functions

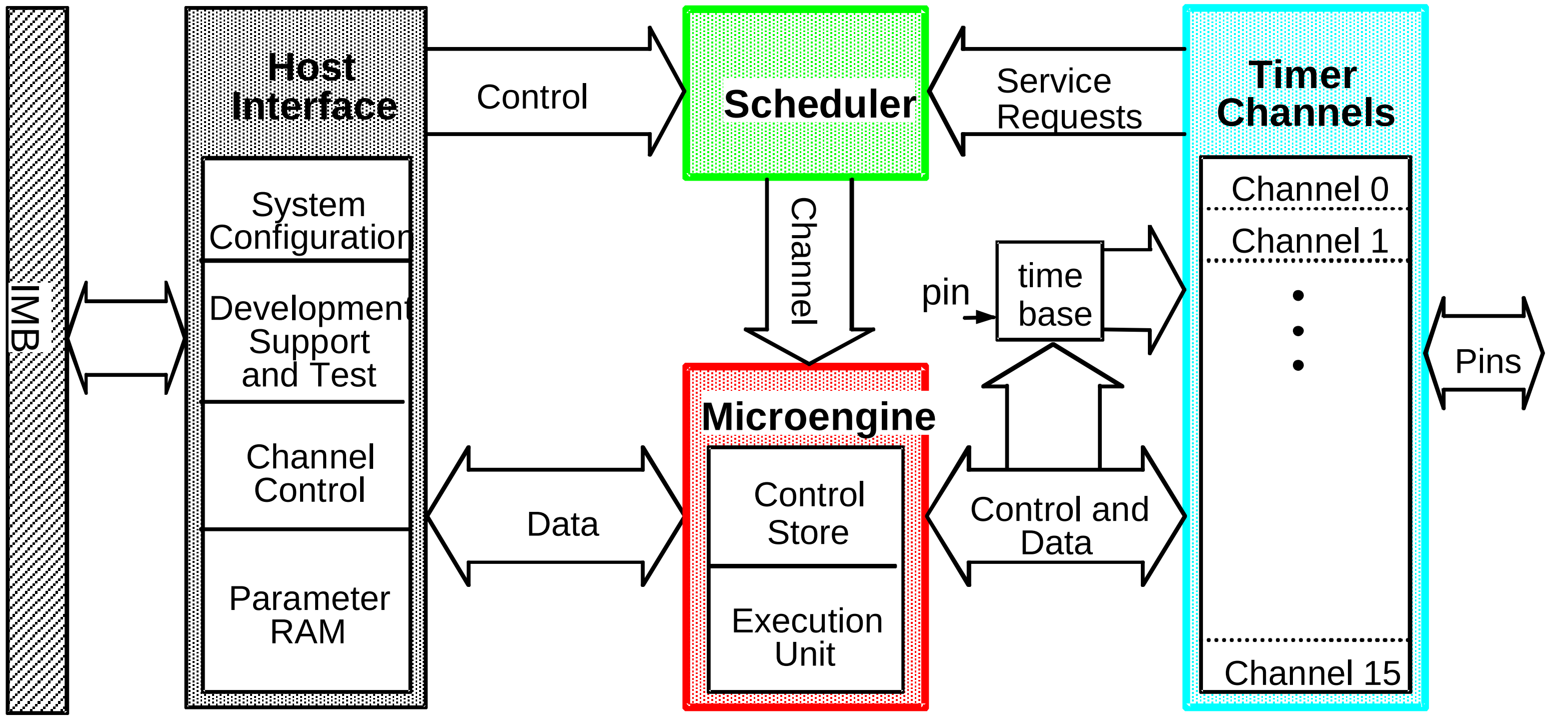
Idea: off-load CPU from frequent I/0 interactions to make use of computation performance: $\Rightarrow$ TPU

control dominated systems

Source: Prof. Rolf Ernst

MC68332

- independent programmable timer channels: single-shot "capture & compare"
- channel coupling and sequence control with control processor

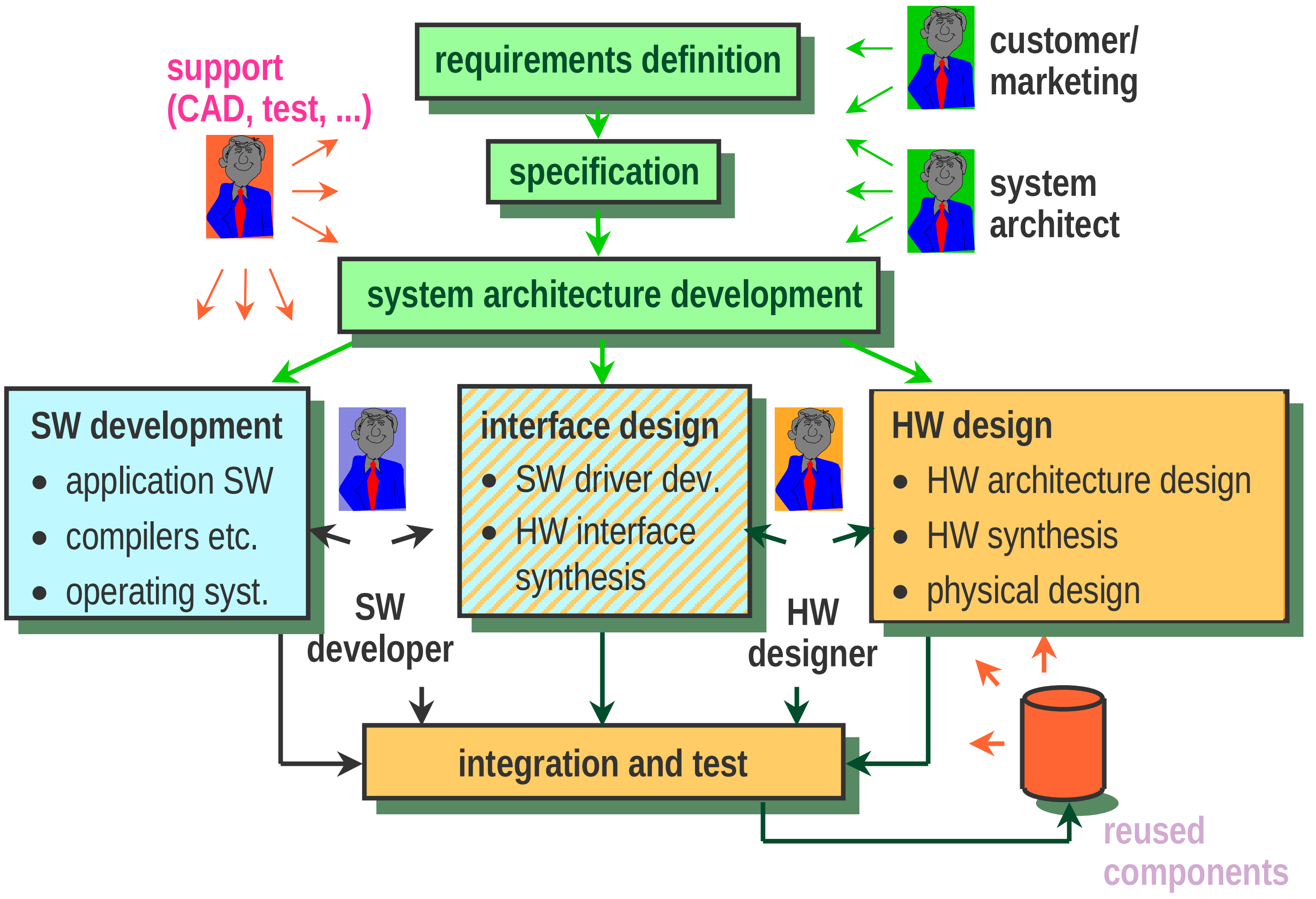


TPU: time processing unit: peripheral coprocessor

control dominated systems

Source: Prof. Rolf Ernst

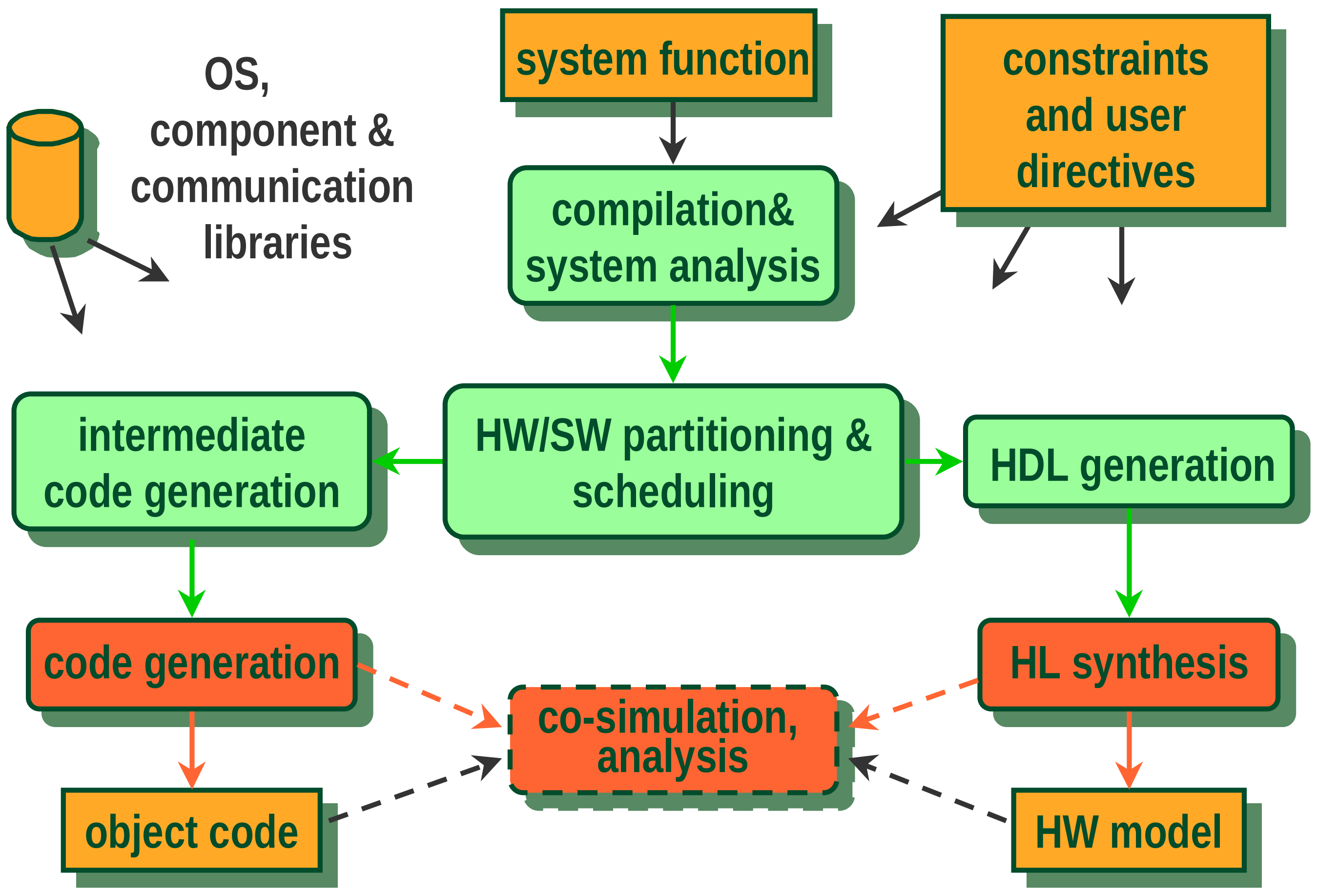
Embedded System Design Process



Embedded system design process

Source: Prof. Rolf Ernst

Co-synthesis Design Flow - Principle



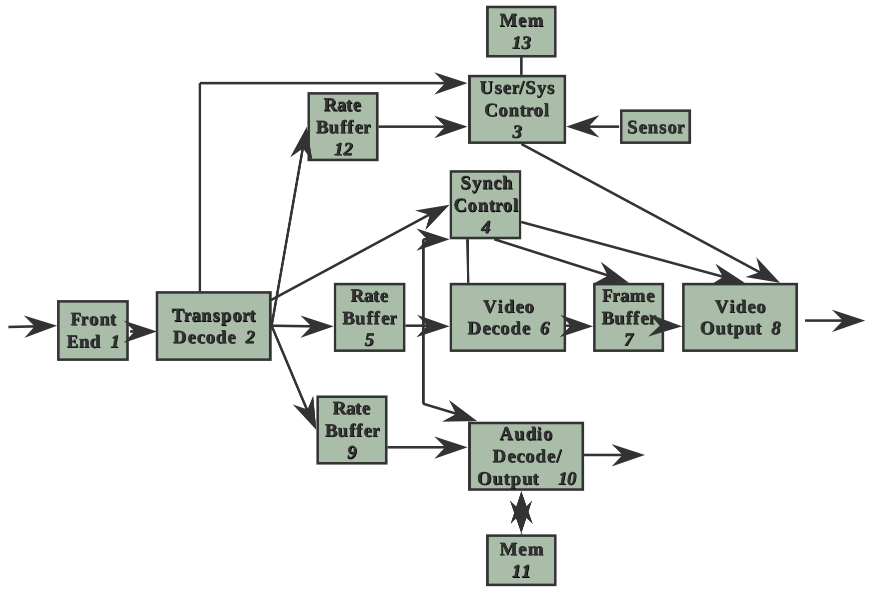
OS, component & communication libraries
system function
constraints and user directives
compilation& system analysis
HW/SW partitioning & scheduling
intermediate code generation
HDL generation
code generation
HL synthesis
co-simulation, analysis
object code
HW model

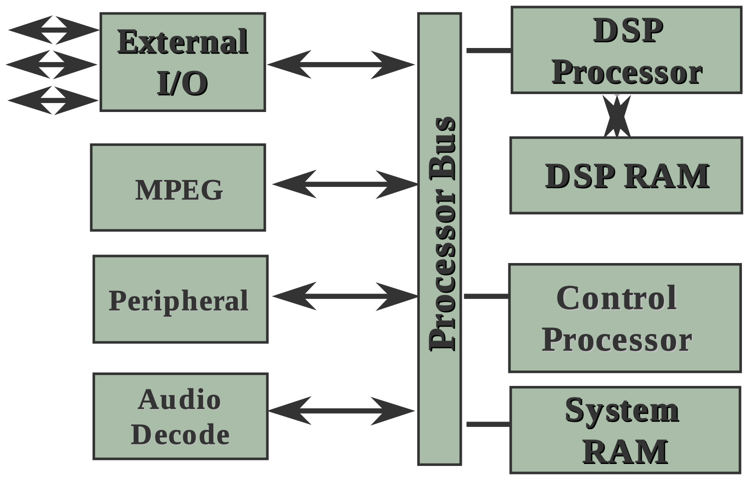
State of the art - Optimization and co-synthesis

Source: Prof. Rolf Ernst

Separate Behavior from Microarchitecture

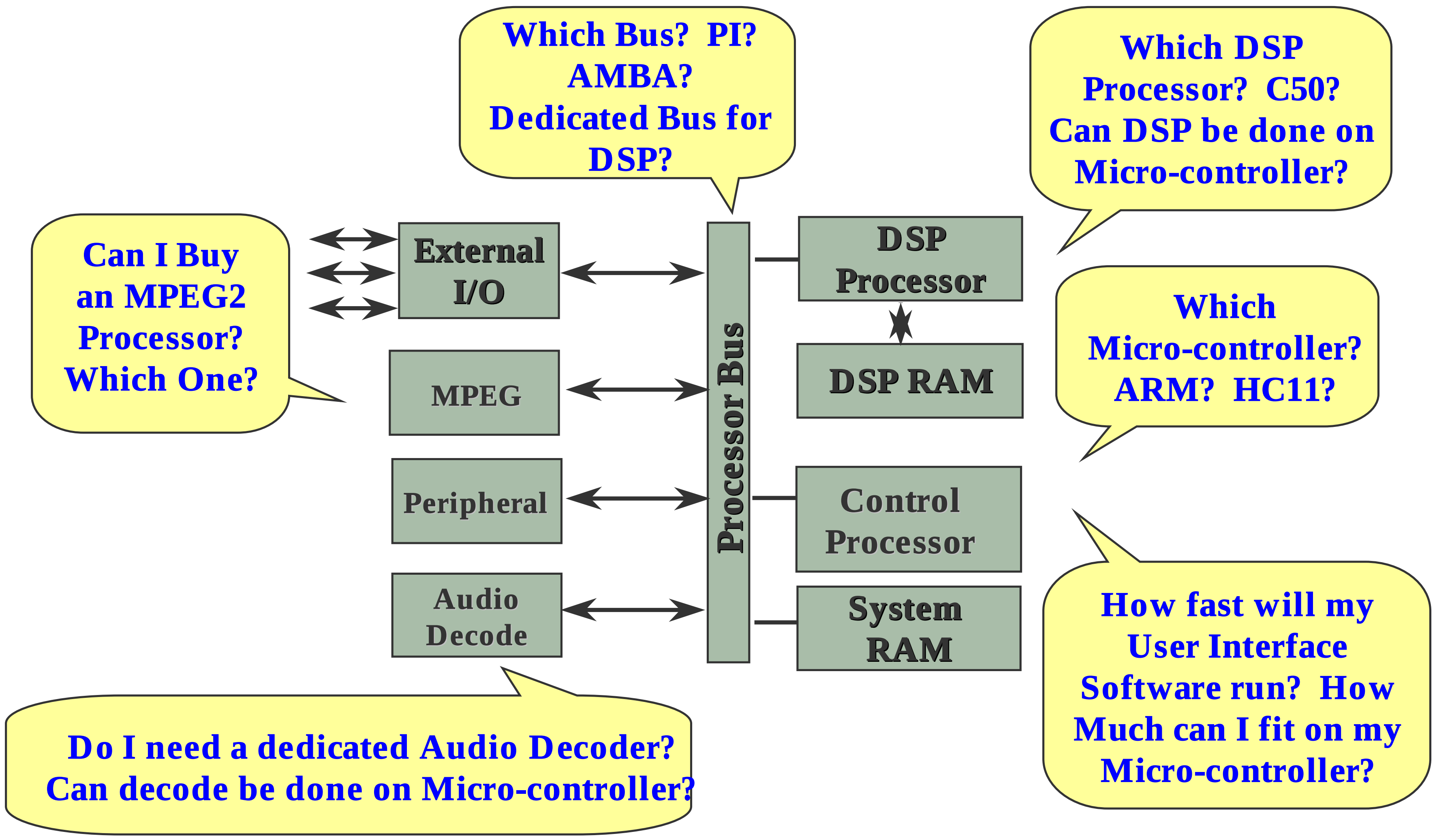
- System Behavior
  - Functional Specification of System.
  - No notion of hardware or software!
- Implementation Architecture
  - Hardware and Software
  - Optimized Computer





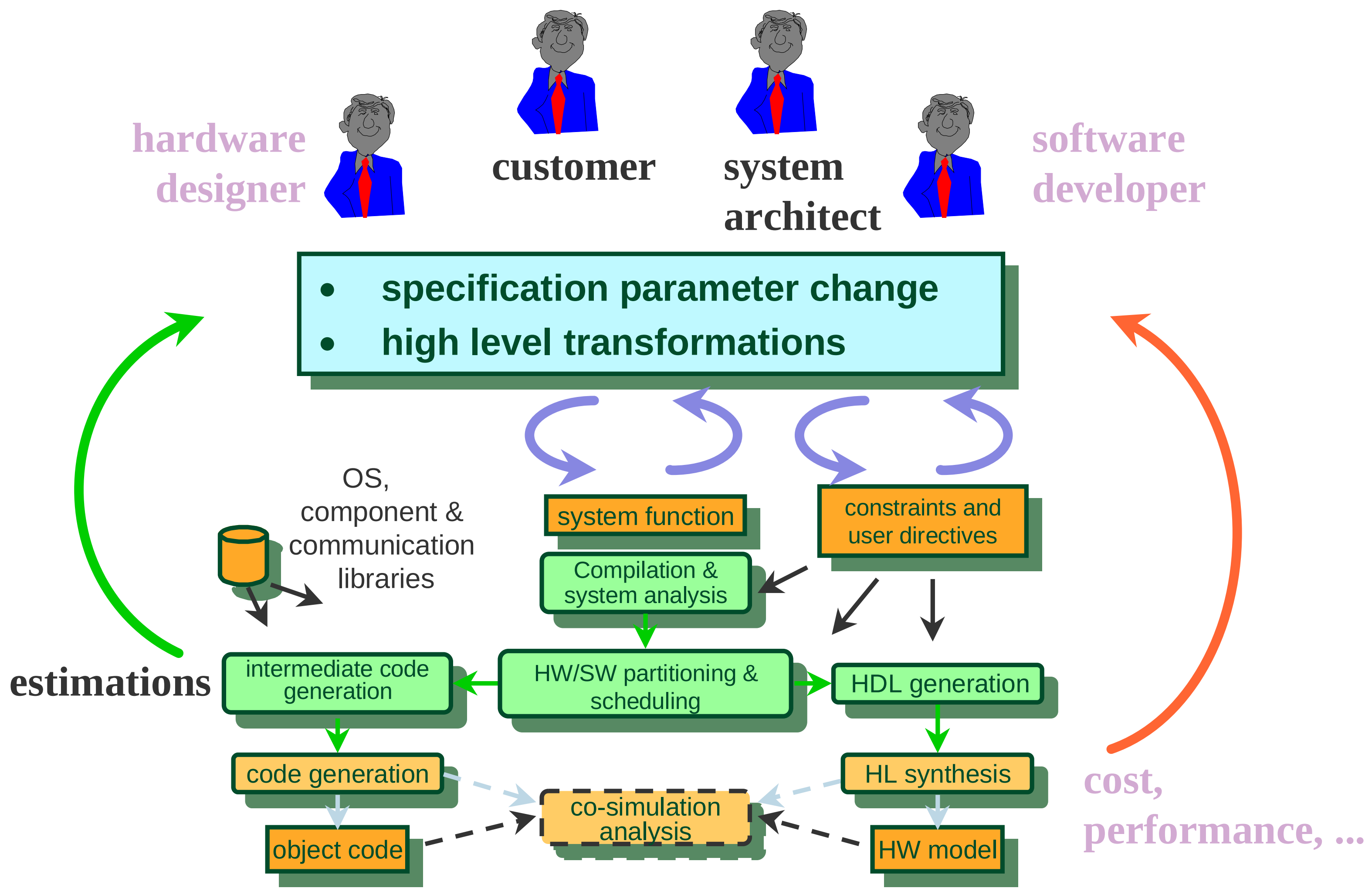
Source: Prof. Alberto Sangiovanni

IP-Based Design of Implementation


Which Bus? PI? AMBA? Dedicated Bus for DSP?
Which DSP Processor? C50? Can DSP be done on Micro-controller?
Can I Buy an MPEG2 Processor? Which One?
Which Micro-controller? ARM? HC11?
How fast will my User Interface Software run? How Much can I fit on my Micro-controller?
Do I need a dedicated Audio Decoder? Can decode be done on Micro-controller?
External I/O
MPEG
Peripheral
Audio Decode
Processor Bus
DSP Processor
DSP RAM
Control Processor
System RAM

Source: Prof. Alberto Sangiovanni

Co-design using co-synthesis and design space exploration



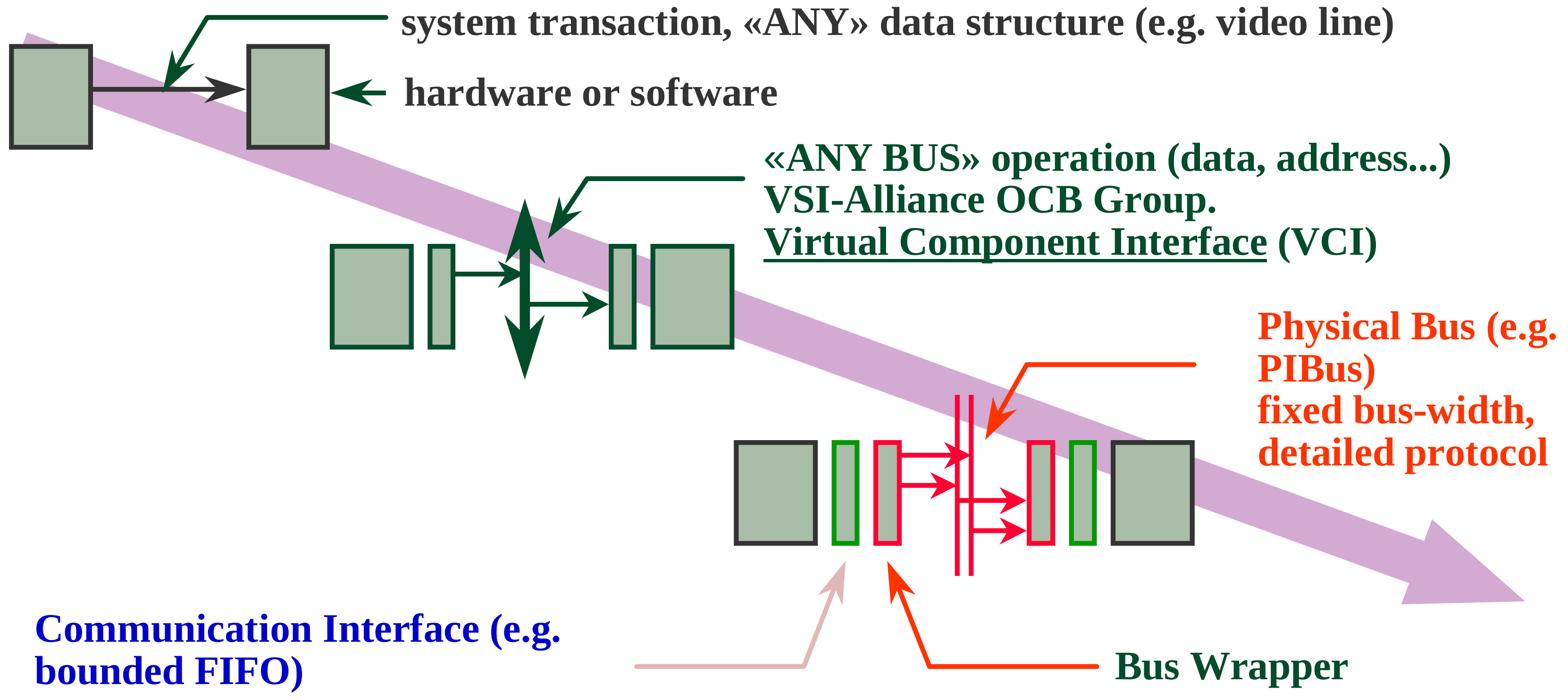
hardware designer
customer
system architect
software developer
specification parameter change
high level transformations
OS, component & communication libraries
system function
constraints and user directives
Compilation & system analysis
estimations
intermediate code generation
HW/SW partitioning & scheduling
HDL generation
code generation
HL synthesis
co-simulation analysis
object code
HW model
cost, performance, ...

State of the art - Optimization and co-synthesis

Source: Prof. Rolf Ernst

Communication Refinement

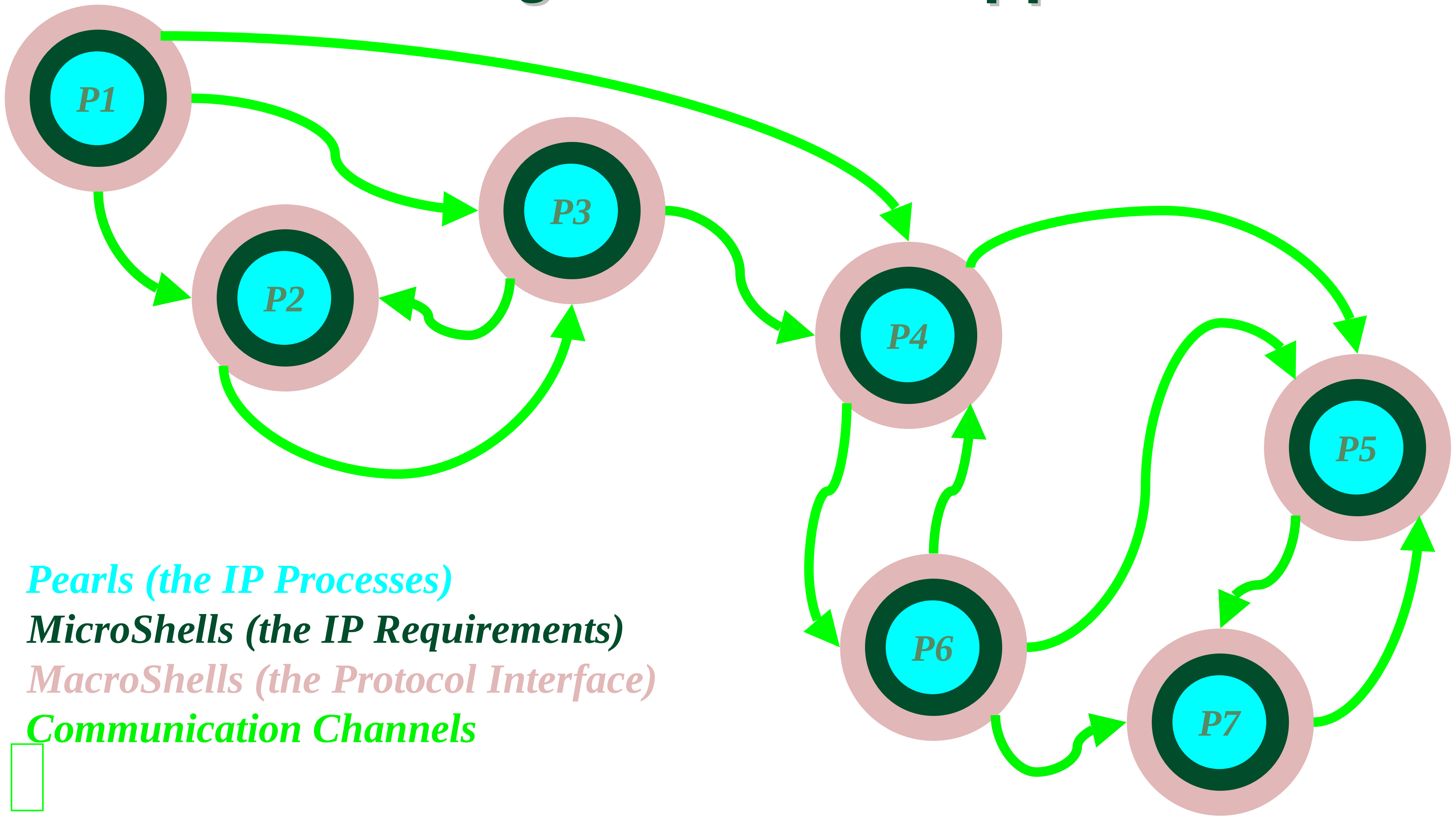
Standard interfaces constitute the backbone of an IP market: abstract form the concerns of hardware implementation (multi-target VC), abstract from the concerns of a particular bus (bus-independent VC)



system transaction, «ANY» data structure (e.g. video line)
hardware or software
«ANY BUS» operation (data, address...)
VSI-Alliance OCB Group.
Virtual Component Interface (VCI)
Physical Bus (e.g. PIBus)
fixed bus-width,
detailed protocol
Communication Interface (e.g. bounded FIFO)
Bus Wrapper

Source: Prof. Alberto Sangiovanni

The *Orthogonalization* Approach



Pearls (the IP Processes)
MicroShells (the IP Requirements)
MacroShells (the Protocol Interface)
Communication Channels

Source: Prof. Alberto Sangiovanni

Communication Design

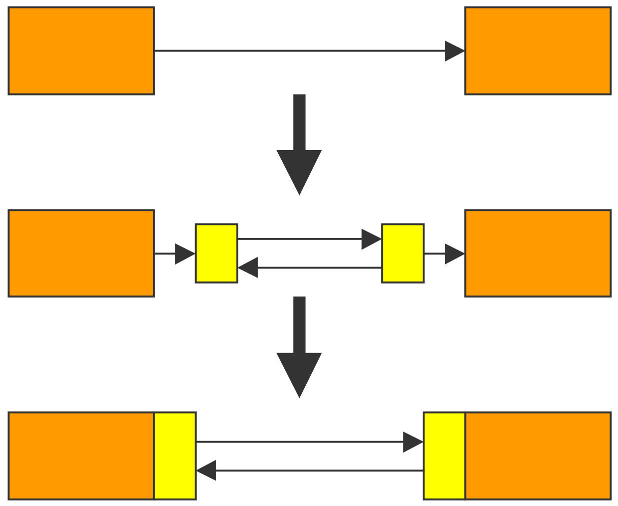
- Determine a protocol so that no matter what the communication topology and the nature of the IP's the functionality of the overall system is guaranteed (TCP/IP like)
- Given the IP set and the interconnections, automatically synthesize protocols and macro-shells
- Given the IP set and a set of time-varying interconnections, automatically synthesize adaptive protocol and macro-shells that optimize "performance" according to the current topology

In collaboration with Jim Rowson

Source: Prof. Alberto Sangiovanni

Model of Computation

- Network of CFSMs
  - Globally asynchronous, locally synchronous (GALS)
  - Extend the model to loss-less communication (abstract CFSM)
  - Communication refined to implementation



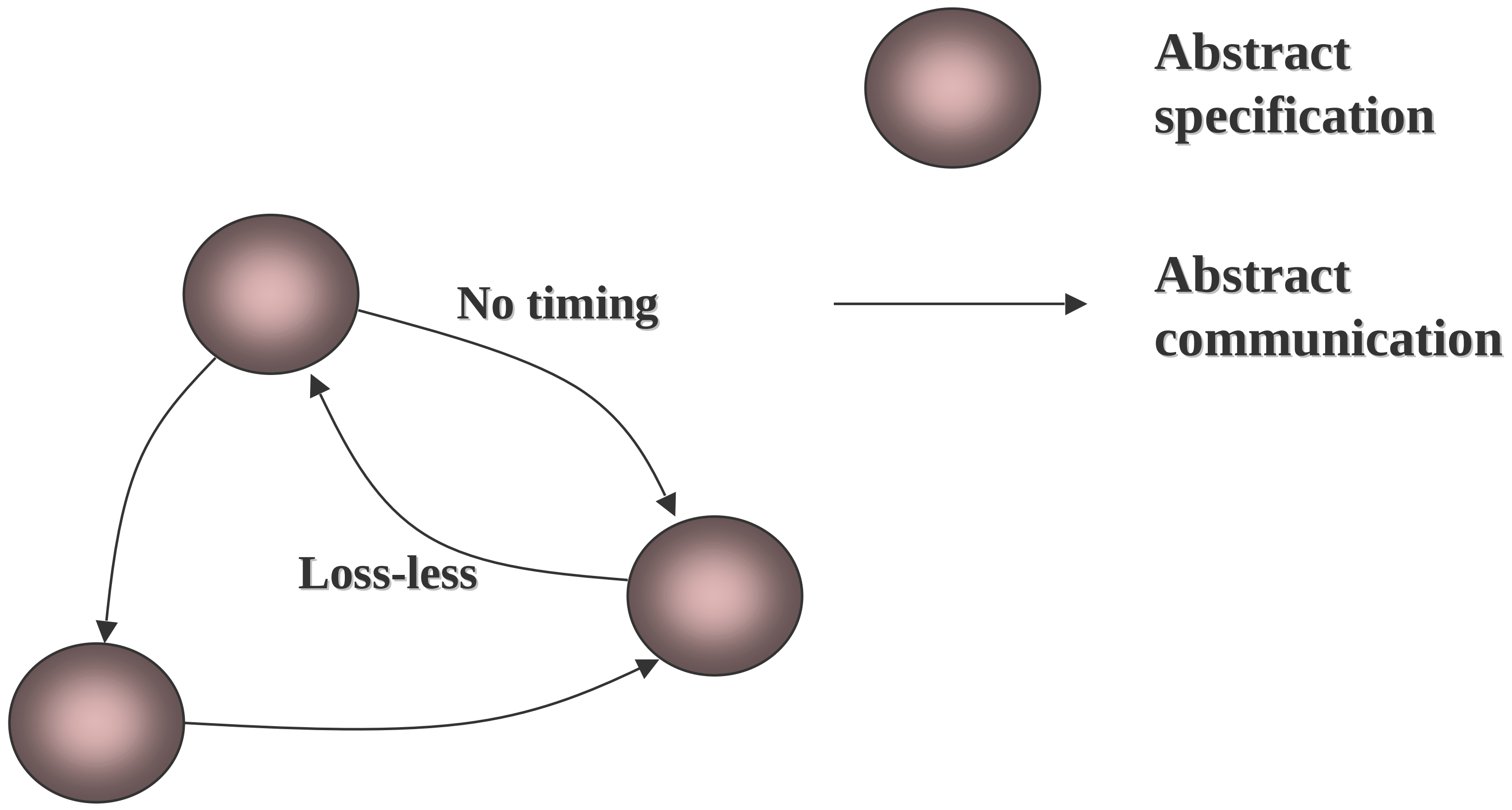
- **Refinement steps:**

  - preserve desired properties at each transformation

  - propagate constraints to lower levels of abstraction (top-down).

Source: Prof. Alberto Sangiovanni

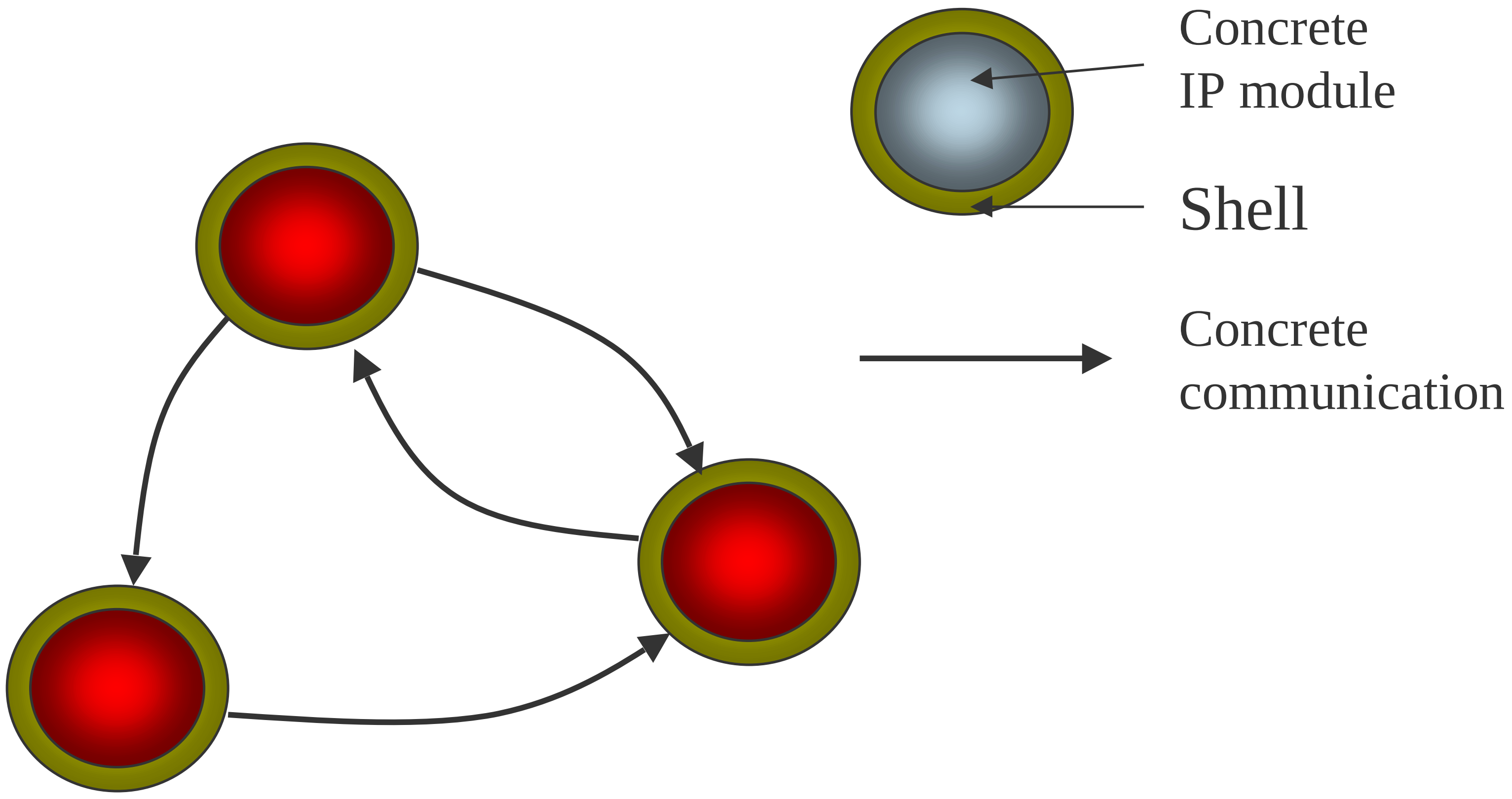
Abstract CFSM



- Maximally non-deterministic view of design
- Design progresses by successive determinization

Source: Prof. Alberto Sangiovanni

CFSM Refinement



Source: Prof. Alberto Sangiovanni

Directions

- Energy-efficient architectures for protocol processing
  - most effort and results in “data-flow” components
  - complex protocol processing is becoming bottleneck
  - instruction processors energy-inefficient
  - CFSM-based architectures attractive from software perspective
- Heterogeneous Platforms and their Software Operation Environment

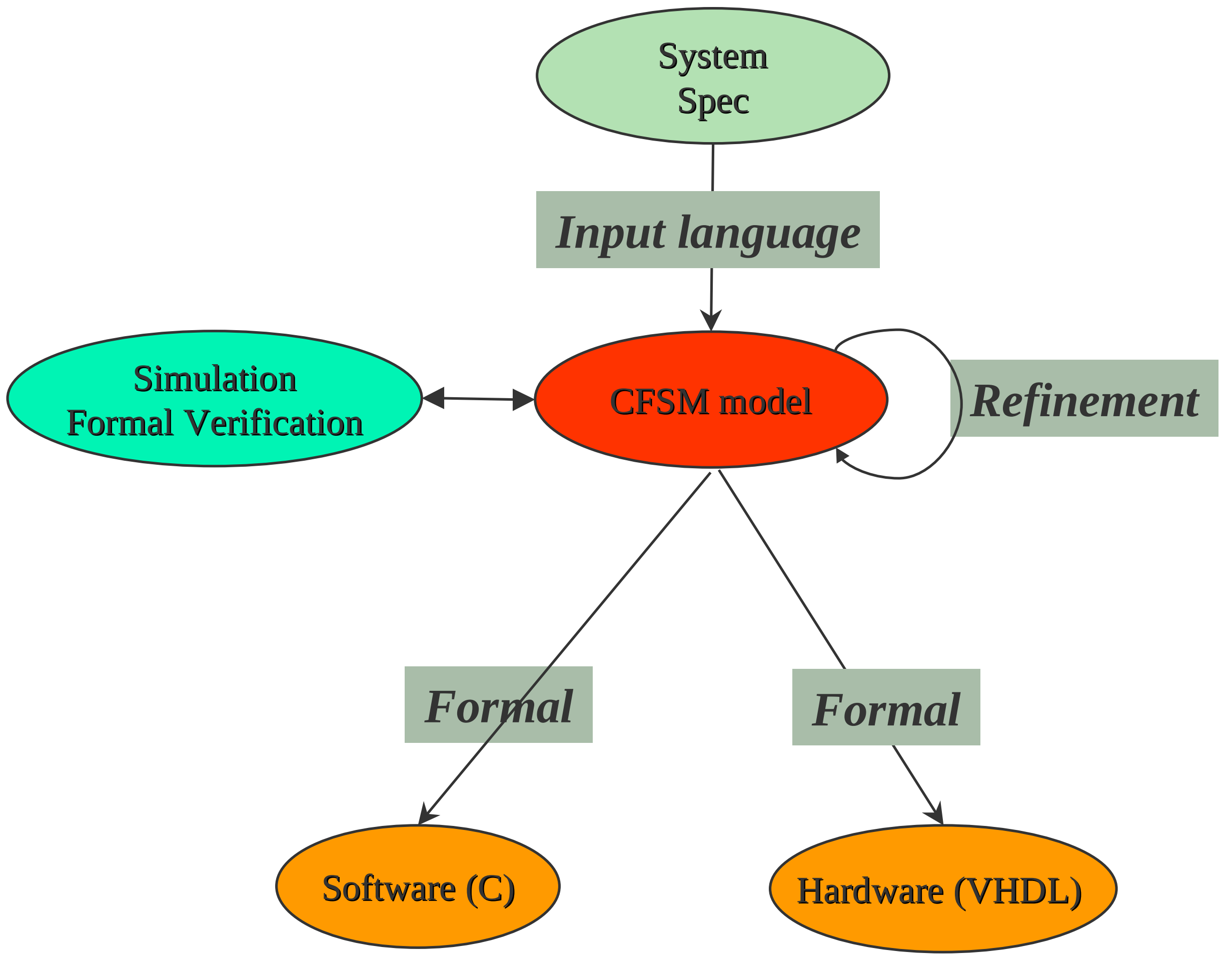
Source: Prof. Alberto Sangiovanni

Protocol Design

- Specification
  - formally describing what the protocol is supposed to do
- Abstraction
  - consistent layering promotes re-use and verification
- Verification
  - is the protocol logically consistent?
- Performance Estimation
  - is the protocol efficient?
- Implementation
  - building a system that implements the specification

Source: Prof. Alberto Sangiovanni

Refinement-based Protocol Design Methodology



Source: Prof. Alberto Sangiovanni

System-on-Chip and IP-based Design

- Two parts to research:
  - Glue Logic design methodology:
    - Merge Place and Route with Logic Synthesis (Constraint Driven Synthesis)
    - Investigate regular circuit fabrics (solve the problem by construction paradigm)
  - Interconnect Design Methodology (Interface-based paradigm)
    - Block Encapsulation

Source: Prof. Alberto Sangiovanni

The Methodology

- Orthogonalize *computation* and *communication*
- *Plug-and-Play* system design
- Chip assembled using *IP cores* exchanging data by means of a *communication protocol*
- Interface Logic Blocks (*the shells*) *encapsulate* and protect the IP cores (*the pearls*)
- *Assume-Guarantee Reasoning* is adopted to *formally verify* IP cores and communication protocols in separate steps

Work in collaboration with K. McMillan, L. Lavagno and A. Saldanha

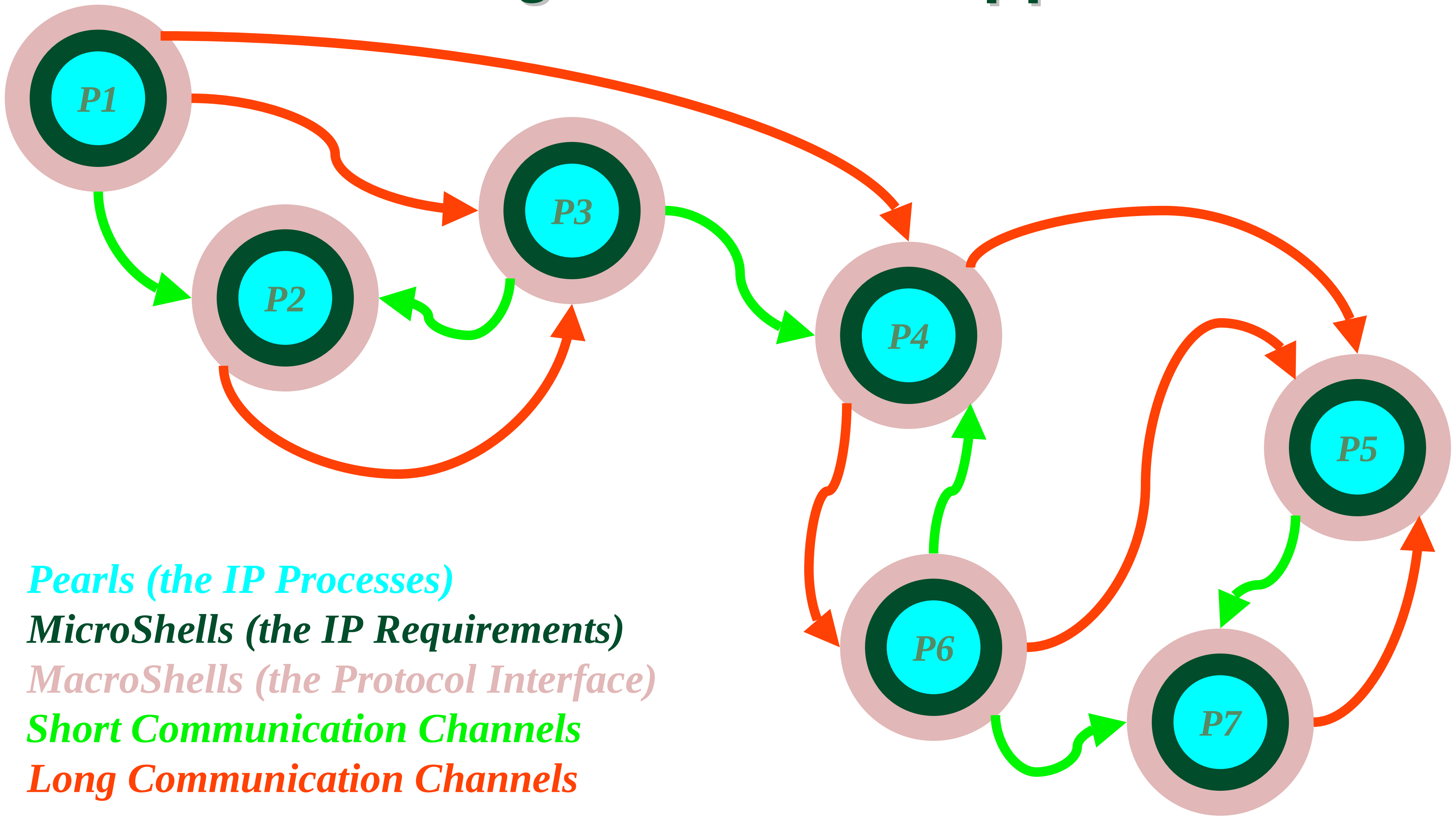
Source: Prof. Alberto Sangiovanni

Latency-Insensitive Communication Protocol

- Long channels are ***segmented*** by inserting simple memory stages (***Relay Stations***)
- Channel latencies are considered arbitrary
- Requirement on IP cores :
  - they must be *stallable*
- Micro Shells :
  - controls stalling mechanism
- Macro Shells :
  - synchronize data and interface with channels

Source: Prof. Alberto Sangiovanni

The *Orthogonalization* Approach



Pearls (the IP Processes)

MicroShells (the IP Requirements)

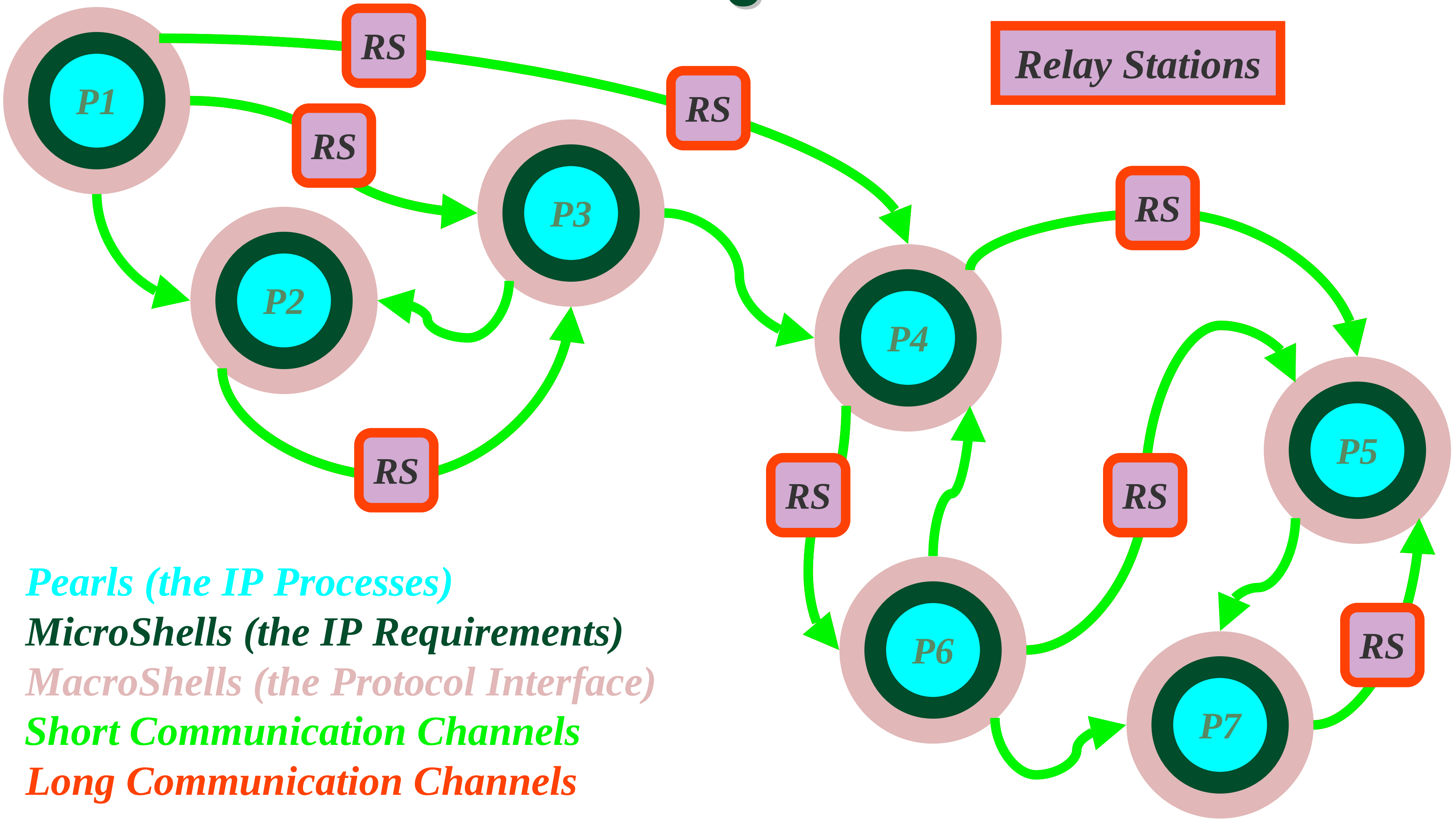
MacroShells (the Protocol Interface)

Short Communication Channels

Long Communication Channels

Source: Prof. Alberto Sangiovanni

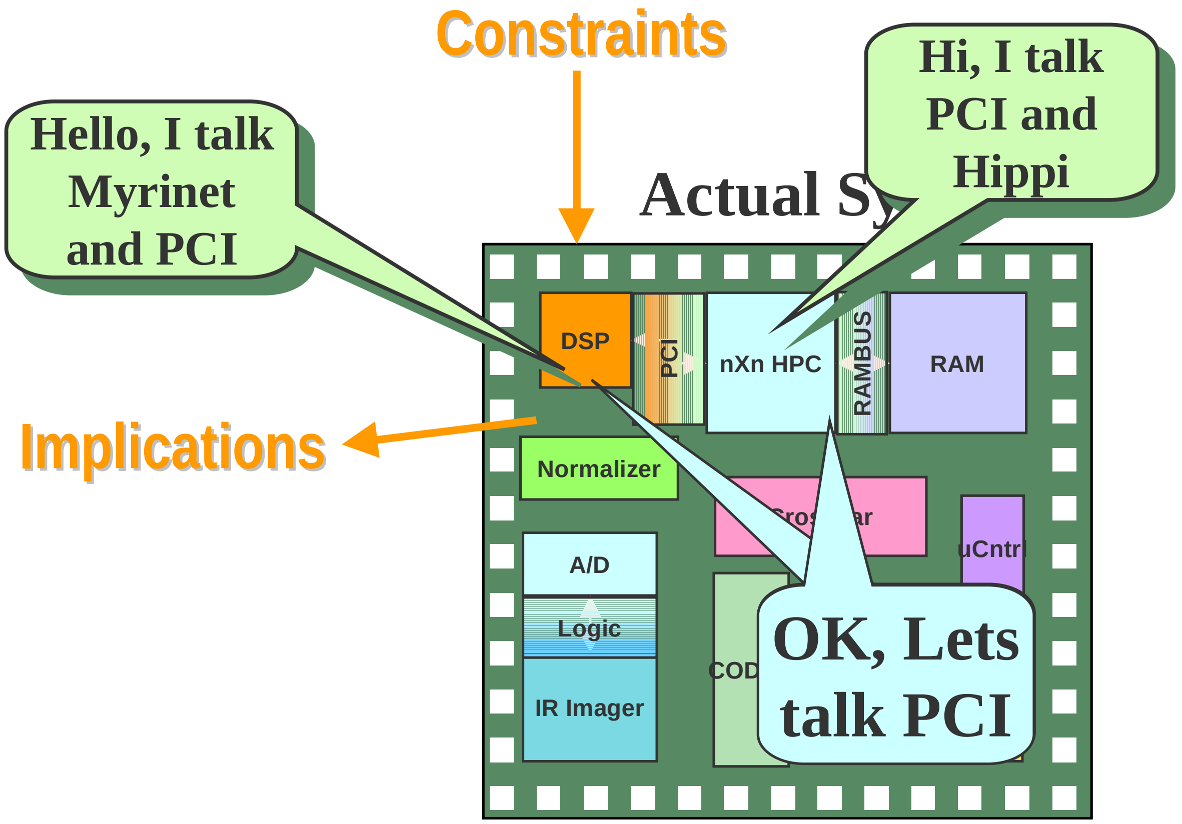
Channel Segmentation



Pearls (the IP Processes)
MicroShells (the IP Requirements)
MacroShells (the Protocol Interface)
Short Communication Channels
Long Communication Channels

Source: Prof. Alberto Sangiovanni

Automated Interface Synthesis



Source: DARPA ISAT *Silicon 2010* Study, 1997
(Randy Harr, Synopsys)