Interface-Based Design

Introduction

A. Richard Newton
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
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 == 1 bit
  - 10 m distance: 0.03 operation == 1 bit

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

(courtesy of J. Rabaey)
Energy-efficient Programmable Implementation Platform

“Software-defined Radio”

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

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

- Preconditions
- Postconditions

Component

Preconditions

Postconditions

Contract

Interface
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:

\[
\begin{align*}
\text{Resolution of INTERNAL (kernel) Details} \\
\text{Resolution of EXTERNAL (Interface) Details}
\end{align*}
\]

In Terms of:

**Temporal Precision**

- High Res: Gate Prop. (pS), Clock Acc. (10ns of nS), Cycle Approx. (100ns of nS), Instr. Cycle (100ns of µS), Token Cycle (100ns of µS), System Event (100ns of mS), Partial Order (conc. seq.)
- Low Res: Partial Order

**Data Precision**

- High Res: Bit (0b01101), Format (Processor-like), Value (13), Property (enumeration e.g. Blue), Token
- Low Res: Token

**Functional Precision**

- High Res: Digital Logic (Boolean operations), Algorithmic (Bubble-sort procedure), Mathematical (W=R^1b)
- Low Res: Mathematical

**Structural Precision**

- High 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)
- Low Res: Single Black Box

**Programming Abstraction Level**

- High Res: Object-Code (1001011), Micro-Code (imul r1, r2), Assembly-Code (i := i + 1), HLL (ADA,C) Statements (FFT(a,b,c)), DSP Primitive (Block-Oriented), Major Modes (Search, Track)
- Low Res: Major Modes

(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

Source: Prof. Edward Lee
Abstracting Synchrony

A

B

C

D

synchronous/re reactive modules

parallel
(synchronous circuits)

sequential
(embedded software)

Source: Prof. Edward Lee
Abstracting Rendezvous

A
B
C
D

communicating sequential processes

parallel (circuits with handshaking)  sequential (software with threads)

Source: Prof. Edward Lee
Abstracting Message Passing

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

#### Philips 83 C552: 8 bit-8051 based microcontroller

- **Complete system**
- Timers, PWM for control
- I²C-bus and par./ser. interfaces for communication
- A/D converter
- Watchdog (SW activity timeout): safety
- On-chip memory
- Interrupt controller

<table>
<thead>
<tr>
<th>Component</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor 80C51</td>
<td>8K8 ROM (87C552 8K8 EPROM)</td>
</tr>
<tr>
<td>15-vector interrupt</td>
<td>256x8 RAM</td>
</tr>
<tr>
<td>Timer0 (16 bit)</td>
<td>A/D Converter</td>
</tr>
<tr>
<td>Timer1 (16 bit)</td>
<td>10-bit</td>
</tr>
<tr>
<td>Timer2 (16 bit)</td>
<td>PWM</td>
</tr>
<tr>
<td>Watchdog (T3)</td>
<td>UART</td>
</tr>
<tr>
<td>Parallel ports 1 through 5</td>
<td>I²C</td>
</tr>
</tbody>
</table>

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) → context switch is more expensive
- virtual memory 
  - supervisor and user modes
- table lookup instructions for compressed tables with built-in linear interpolation → data density is concern

control dominated systems

Source: Prof. Rolf Ernst
Designed for automotive applications with mixture of computation intensive tasks and complex I/O functions

Idea: off-load CPU from frequent I/O interactions to make use of computation performance: \( \rightarrow \) TPU

control dominated systems

Source: Prof. Rolf Ernst
- independent programmable timer channels: single-shot "capture & compare"
- channel coupling and sequence control with control processor

TPU: time processing unit: peripheral coprocessor

Source: Prof. Rolf Ernst
Embedded System Design Process

Source: Prof. Rolf Ernst
Co-synthesis Design Flow - Principle

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 DSP Processor? C50? Can DSP be done on Micro-controller?

Which Bus? PI? AMBA? Dedicated Bus for DSP?

Can I Buy an MPEG2 Processor? Which One?

Which Micro-controller? ARM? HC11?

Can I need a dedicated Audio Decoder? Can decode be done on Micro-controller?

How fast will my User Interface Software run? How Much can I fit on my Micro-controller?

Source: Prof. Alberto Sangiovanni
Co-design using co-synthesis and design space exploration

- specification parameter change
- high level transformations

State of the art - Optimization and co-synthesis

Source: Prof. Rolf Ernst
Standard interfaces constitute the backbone of an IP market: abstract from 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

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

Concrete IP module

Shell

Concrete communication

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

System Spec

Input language

CFSM model

Simulation
Formal Verification

Refinement

Formal

Software (C)

Formal

Hardware (VHDL)

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

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

Source: Prof. Alberto Sangiovanni
Automated Interface Synthesis

Hello, I talk Myrinet and PCI

Constraints

Hi, I talk PCI and Hippi

Implications

OK, Let's talk PCI

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