Automatic OS and HW/SW interface generation for communication and macro-architecture refinement of multiprocessor SoC architectures

Wander Cesário

TIMA Laboratory

46, avenue Félix Viallet

38031 GRENOBLE Cedex France

http://tima.imag.fr/

Outline

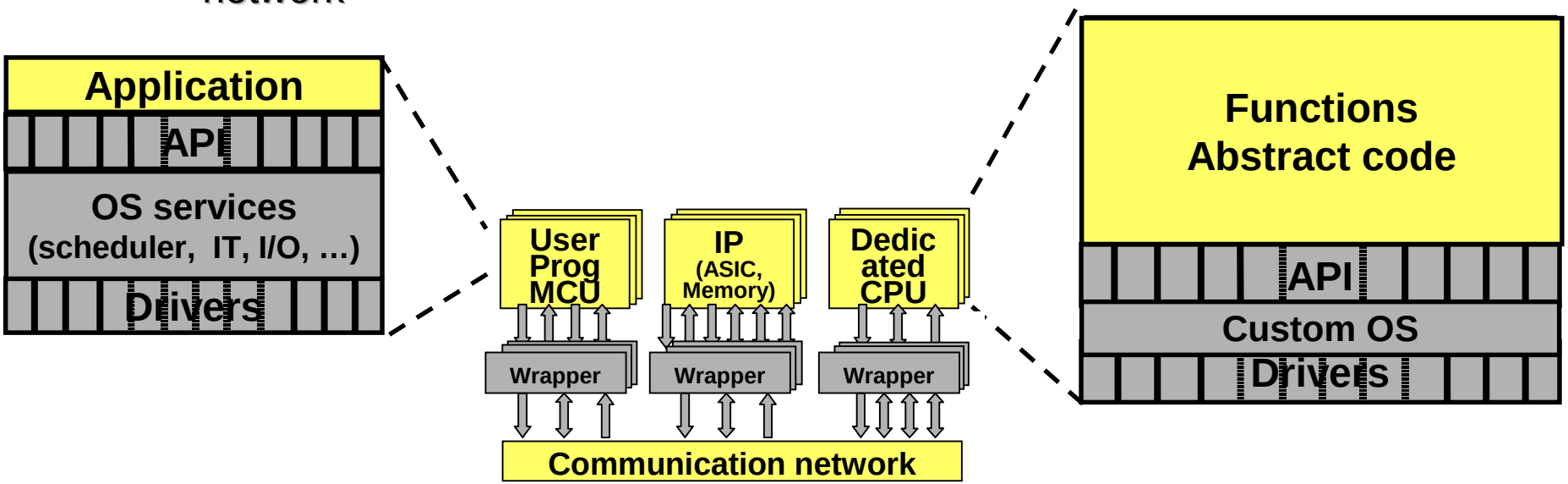
- Multiprocessor SoC
- HW/SW interface design
- VDSL case study
- Results

Motivations and Objectives

- Current SoC design
  - RTL design of HW interfaces
  - Low-level hand coded SW for dedicated processors (e.g. DSP)
- Challenges
  - 200 Million gates SoC <=> 6,000,000 lines of RTL code, 600 man-year effort
  - Dedicated SW complexity larger than SoC in 2005
- Contributions
  - Design at a higher level than RTL of HW interfaces
  - Higher level code for dedicated SW

Multiprocessor SoC Architecture

- Heterogeneous components
  - Specific hardware (ASIC, IP, Memory, non digital, …)
  - User programmable (MCU running application/OS)
  - Designer programmable (Dedicated CPU, ASIP, DSP)
- Complex HW/SW communication network
  - Communication network (bus, switched)
  - Glue, unstructured logic that links components to the communication network

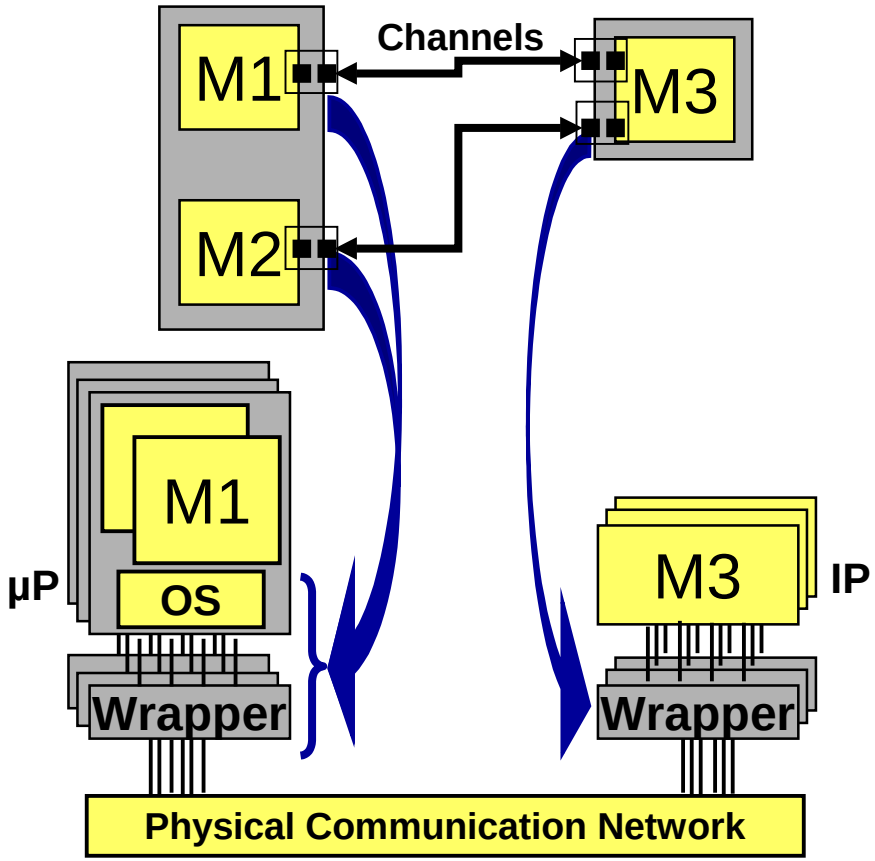


Outline

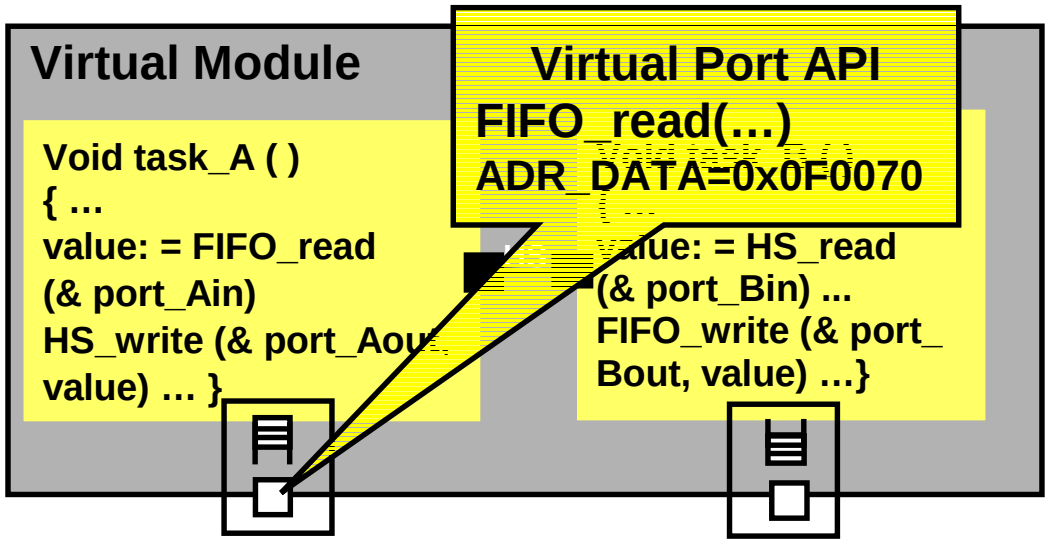
- Multiprocessor SoC
- HW/SW interface design
- VDSL case study
- Results

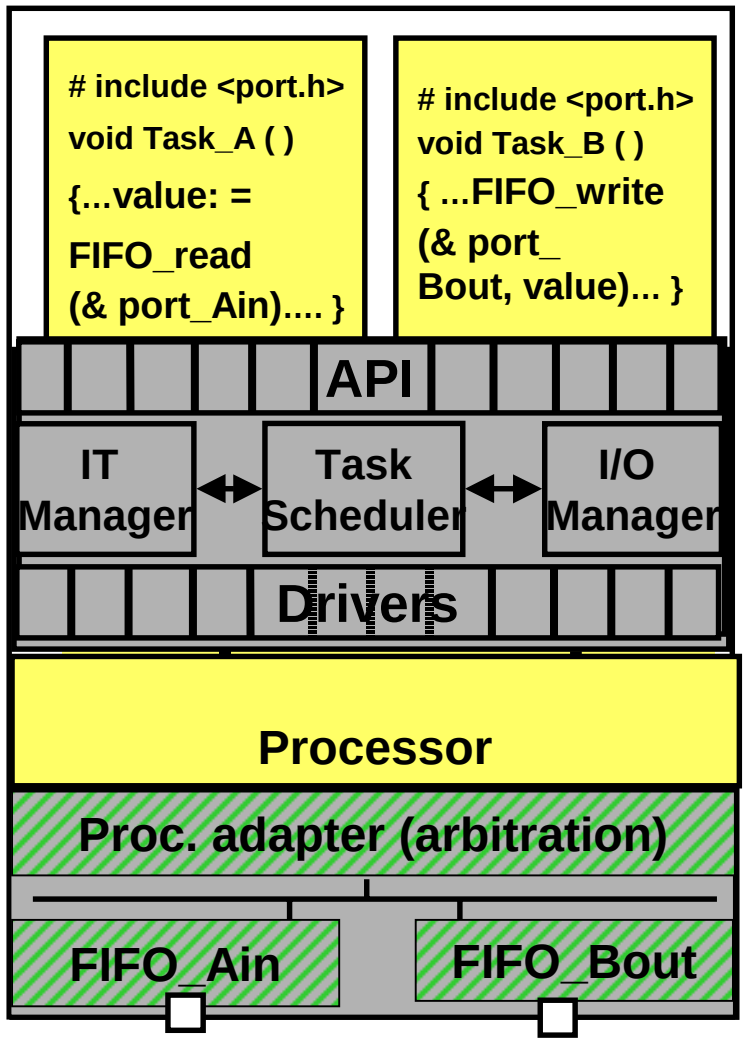
HW/SW Interfaces Design Automation

- Virtual architecture specification: virtual modules use wrappers to separate computation from communication (SW code: C++/SystemC + OS API)
- Architecture model allowing easy integration of heterogeneous components and sophisticated on-chip communication network through HW and SW wrapper concepts (Same SW code runs on implementation)
- Seamless system to RTL flow through automatic generation of wrappers using a systematic assembly approach to build application-specific on-chip HW/SW communication



HW/SW communication abstraction





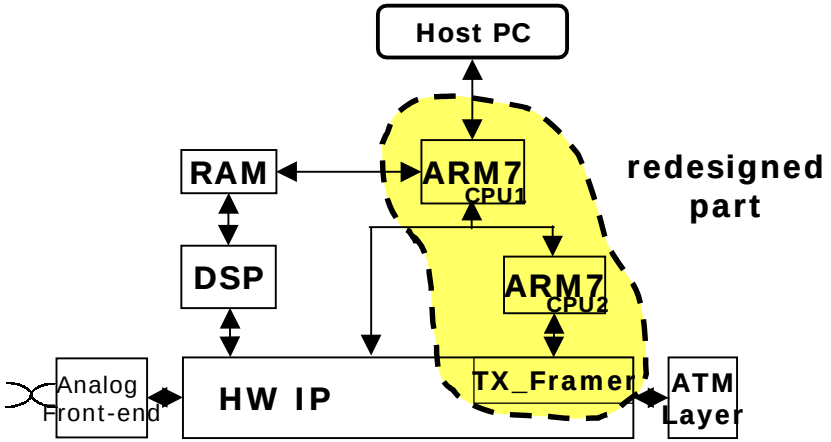
- Virtual Architecture
  - HW/SW wrapper: virtual ports (API, parameters)
  - SW, standard C++/SystemC built on top of API
- RTL architecture
  - Same SW code, runs on implementation
  - SW wrapper (implements API, task control, interrupts, I/O)
  - HW wrapper (bridge to comm. network)

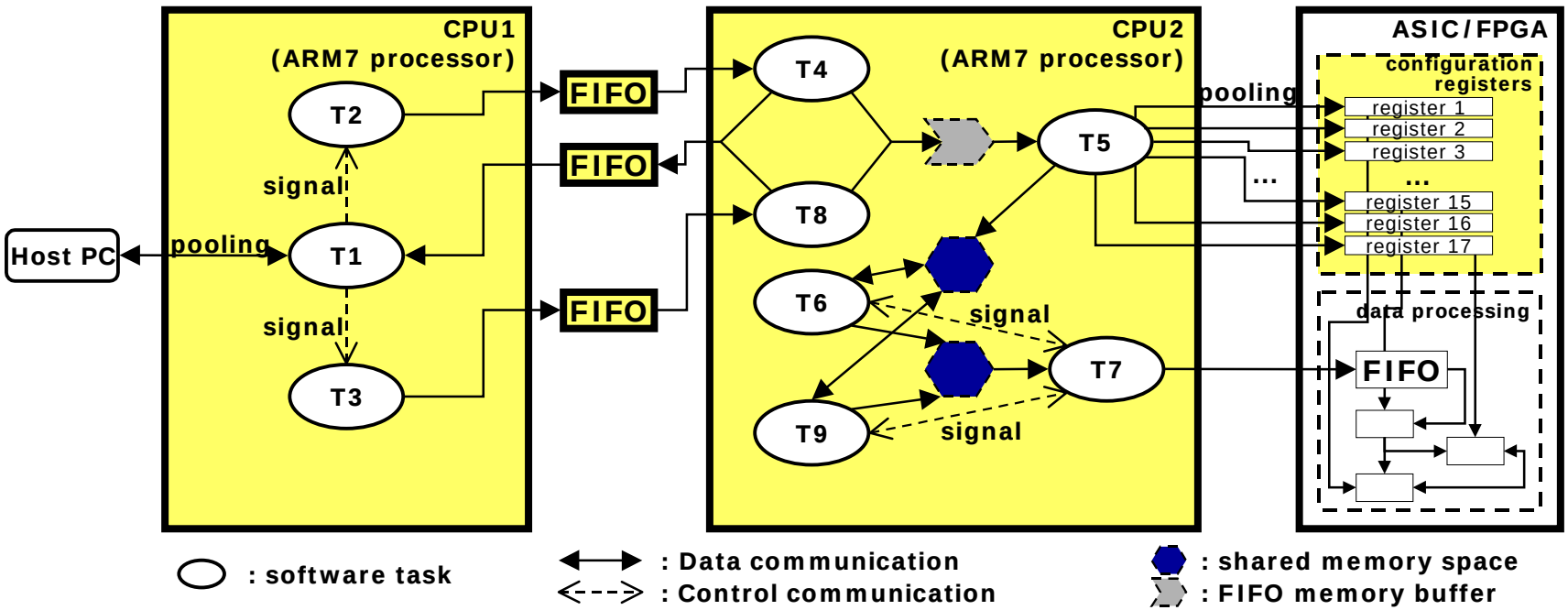
Outline

- Multiprocessor SoC
- HW/SW interface design
- VDSL case study
- Results

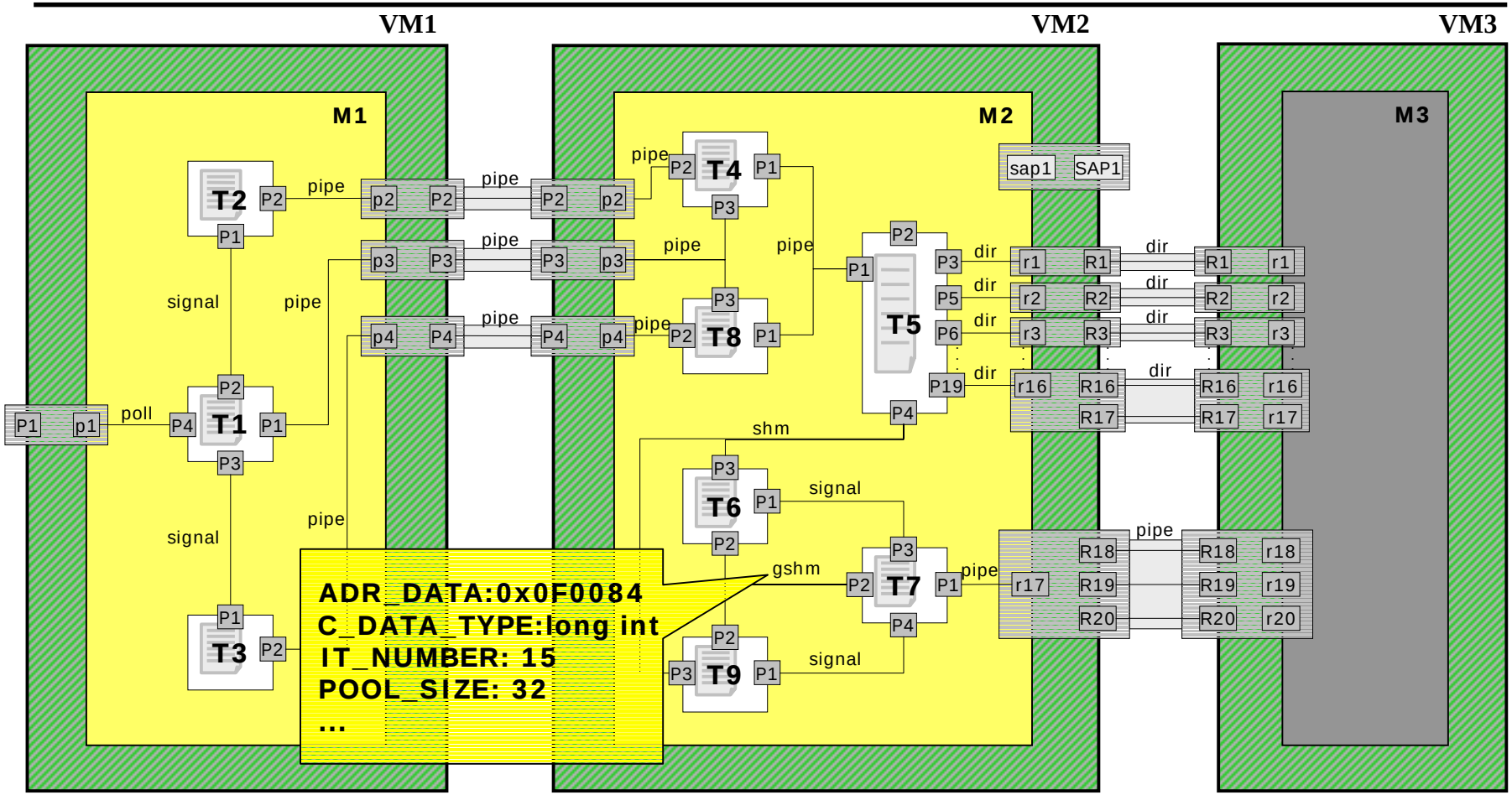
VDSL application: the subset

- Inter-processor communication
- Communication between processor and ASIC
- Parallel communicating tasks
- Different communication/control protocols



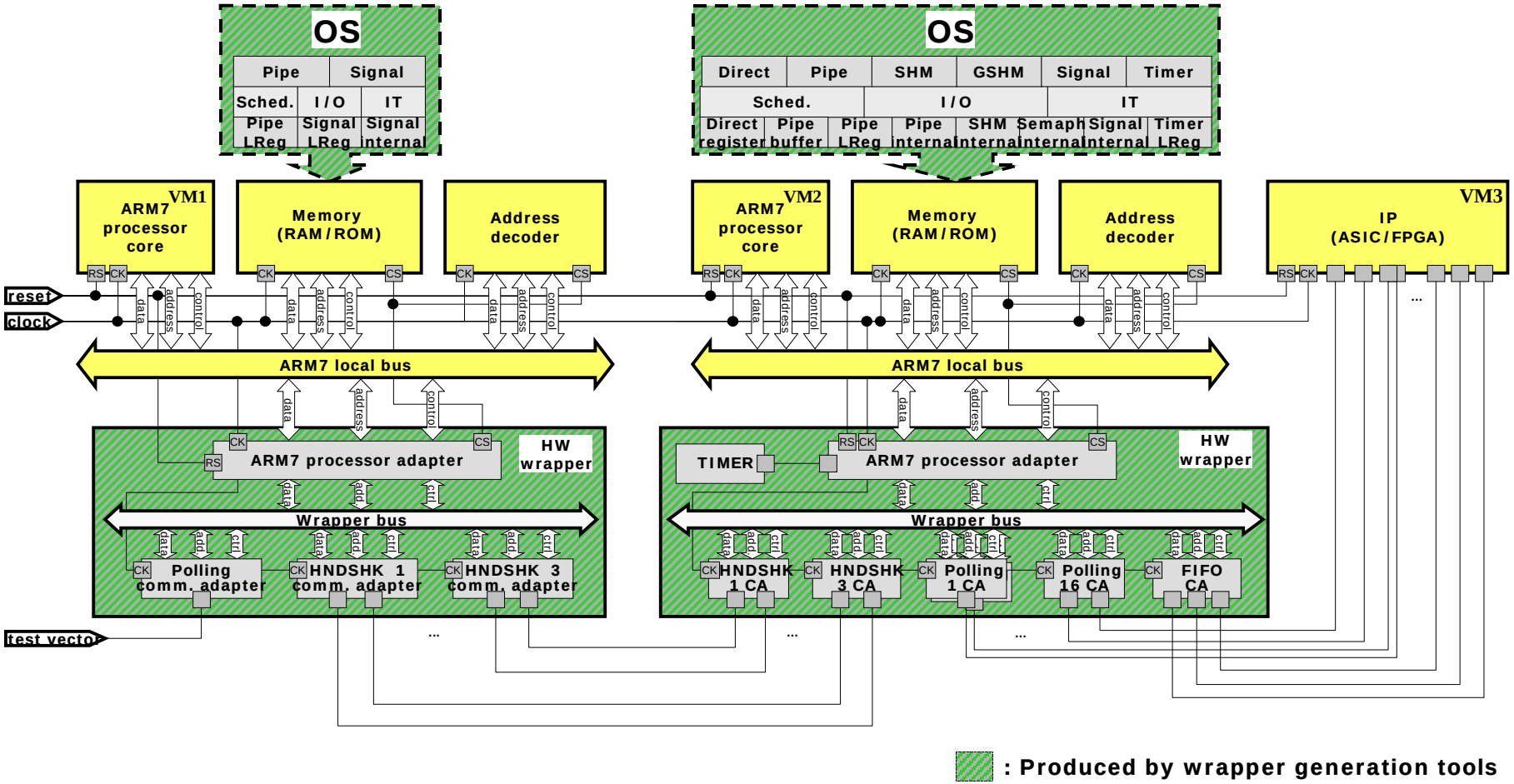


VDSL: the specification



- Powerful modular scheme
  - M1 and M2 include parallel SW tasks, M3 a HW IP RTL model
  - Clear separation between communication and computation
  - Reference model for all designers (HW, SW, SoC)

VDSL: SystemC RTL model



Outline

- Multiprocessor SoC
- HW/SW interface design
- VDSL case study
- Results

VDSL design results (summary)

- Design effort gain: 15x (5 m.y. vs 4 months)
- Virtual architecture model: 150 lines / 21 nets
- RTL architecture (0.35µm technology):

OS results	# of lines in C	# of lines in Assembly	Code size (bytes)	Data size (bytes)
VM1	968	281	1484	500
VM2	1872	281	2624	1020
Context switch (cycles)				36
Latency for interrupt treatment (cycles)				59 (OS) + 28 (ARM7)
System call latency (cycles)				50
Resume of task execution (cycles)				26

HW wrappers	# of gates	Critical path delay (ns)	Max. freq. (MHz)
VM1	968	5.95	168
VM2	1872	6.16	162
Latency for read operation (clock cycles)			6
Latency for write operation (clock cycles)			2
Number of code lines (RTL VHDL)			2168

Conclusions and future work

- Conclusions:
  - Automatic wrapper generation can be as efficient as manually writing/configuring HW interfaces/operating systems
  - The virtual architecture model allows designers to deal with HW/SW interfaces at a high abstraction level: 5570/150 = 37x
  - Communication/computation independence: virtual architecture SW runs on the RTL implementation
- Future work:
  - Global verification of configuration parameters coherence
  - Support of different on-chip communication networks
  - Implementation of generated VDSL design in an ARM prototype board