Intellectual Property Macrocell for SpaceWire Interface Compliant with AMBA-APB Bus

L. Fanucci, A. Renieri, P. Terreni

Tel. +39 050 2217 668, Fax. +39 050 2217522

Email: luca.fanucci@iet.unipi.it

Outline

- **Motivations of the work**
- **Reference SoC Platform**
- **SpaceWire Interface overview**
  - SpaceWire Encoder-Decoder
  - AMBA™ APB interface
- **Design and Verification Methodology**
- **Results and Conclusions**

Motivations of the Work

Forthcoming space applications demand:

more…

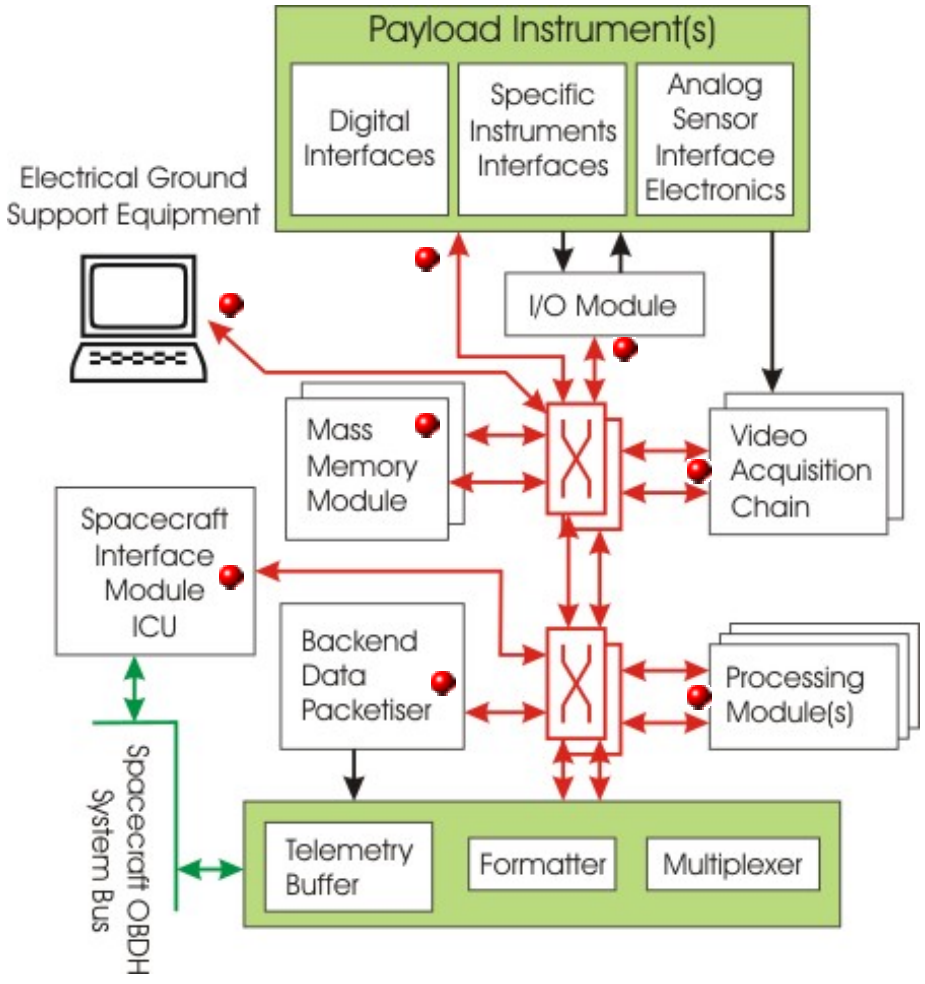
- computational power
- standardization
- reliability

but less…

- mass, weight and power consumption
- cost
- development time

Candidate solutions are based on embedded systems, supporting standard and efficient communication protocols, developed by means of proper design and verification methodologies.

SpaceWire Standard



- Huge amount of data to transfer on board to the satellite
- Critical for High-capability remote-sensing missions
- Cost, flexibility and reuse

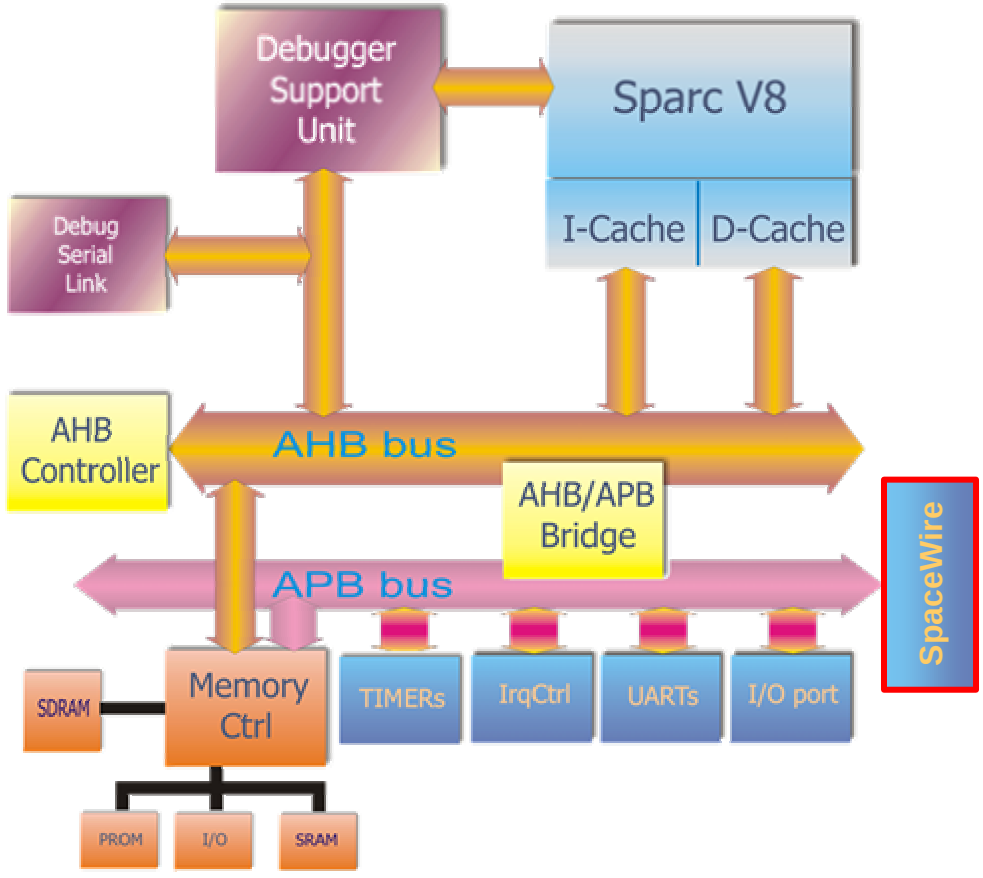
⇩

- New serial data link standard, SpaceWire (ECSS-E-50-12A)
- Development of complex System On Chip (SoC)

⇩

- SpaceWire Interface Intellectual Property (IP) macrocell

Reference SoC Architecture



ESA LEON platform as reference SoC architecture

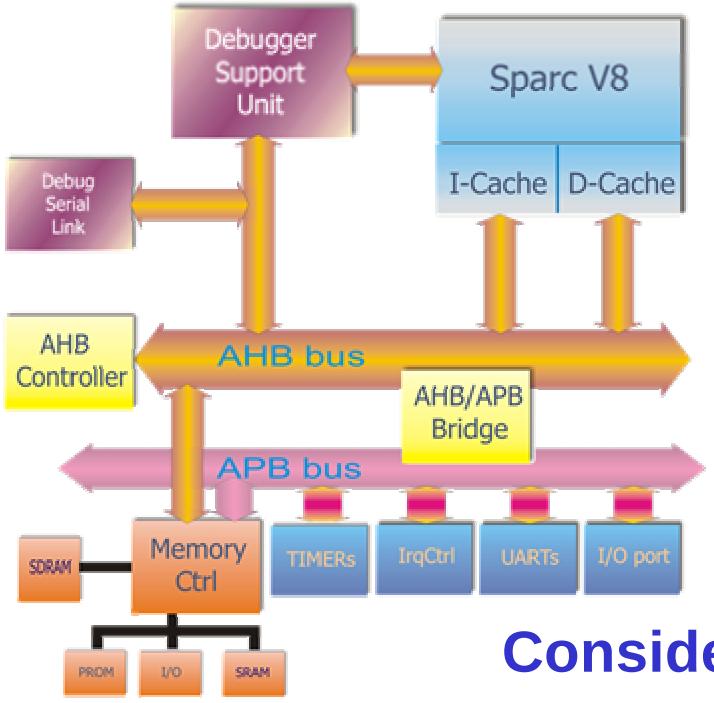
SpaceWire I/F AMBA™ APB bus compliant

SpaceWire interface IP suitable to any AMBA™ based SoC architecture

LEON platform site:
http://www.gaisler.com

ARM AMBA™ bus site:
http://www.arm.com

ESA LEON Embedded Core



The LEON platform provides:

- SPARC-V8 compliant processor
- AMBA 2.0 bus core
- In-circuit debug interface
- General purpose peripherals

Considered configuration

Processor:

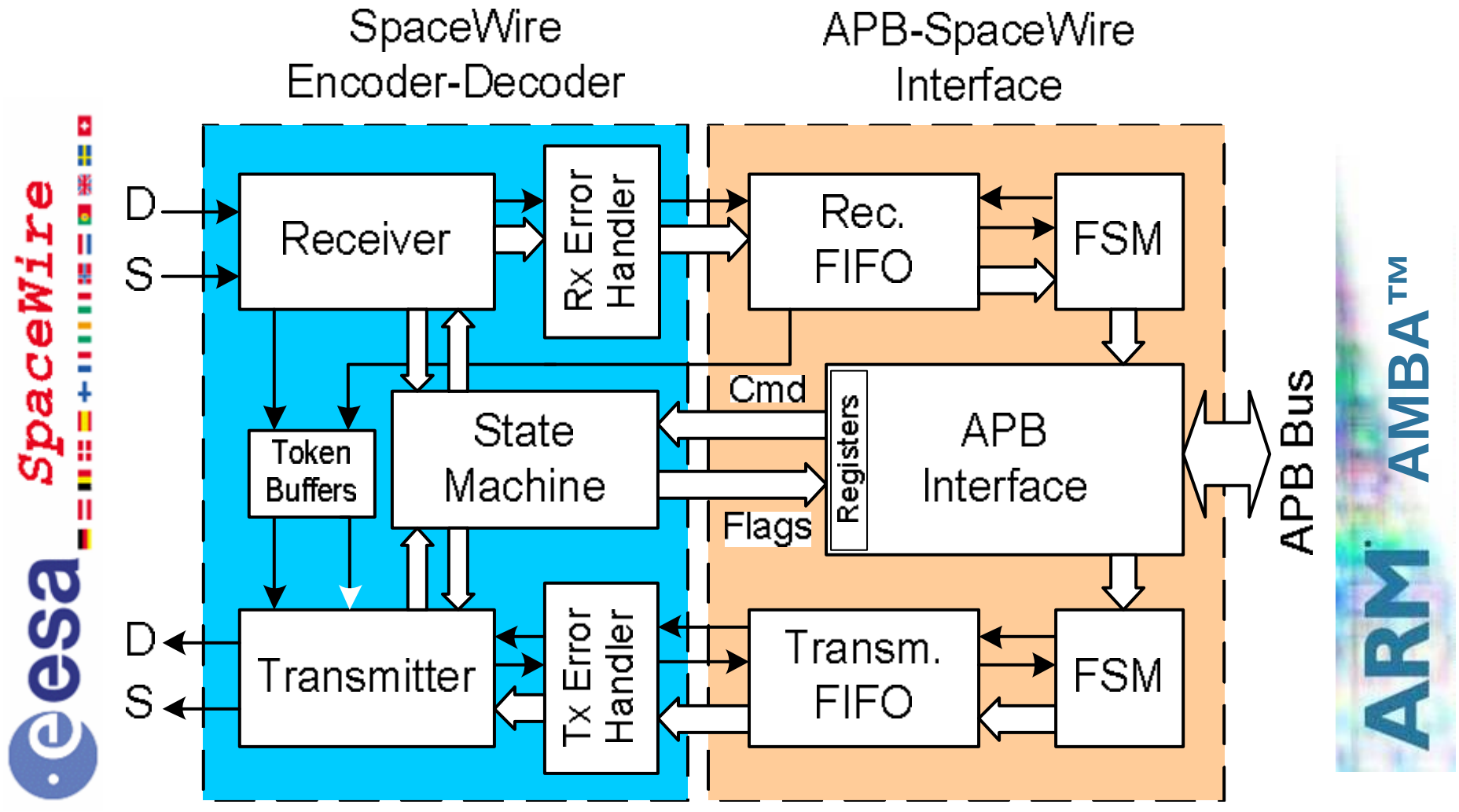
- 32 Register windows
- 2K I-Cache and D-Cache
- Multiplier 16X16 (latency=4)
- MAC present
- No FPU neither coprocessor

Peripherals:

- Debug Support Unit present
- 32KB AHB RAM
- 2 UARTs + Parallel I/O port
- 1 Interrupt Controller
- 2 Timers + Watchdog

SpaceWire I/F for APB Bus

CNR - Università di Pisa - ITALY



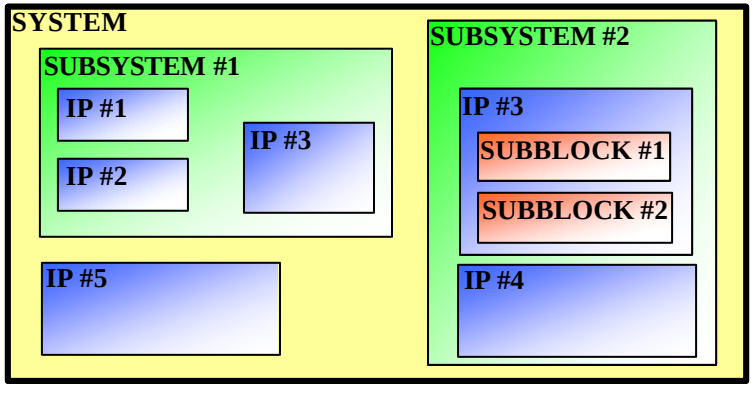
Verification methodology 1/2

The more the system complexity grows, the bigger the effort required for the verification phase in order to guarantee the system reliability

A proper verification methodology needs to be adopted

Identify the correct hierarchy-level among:

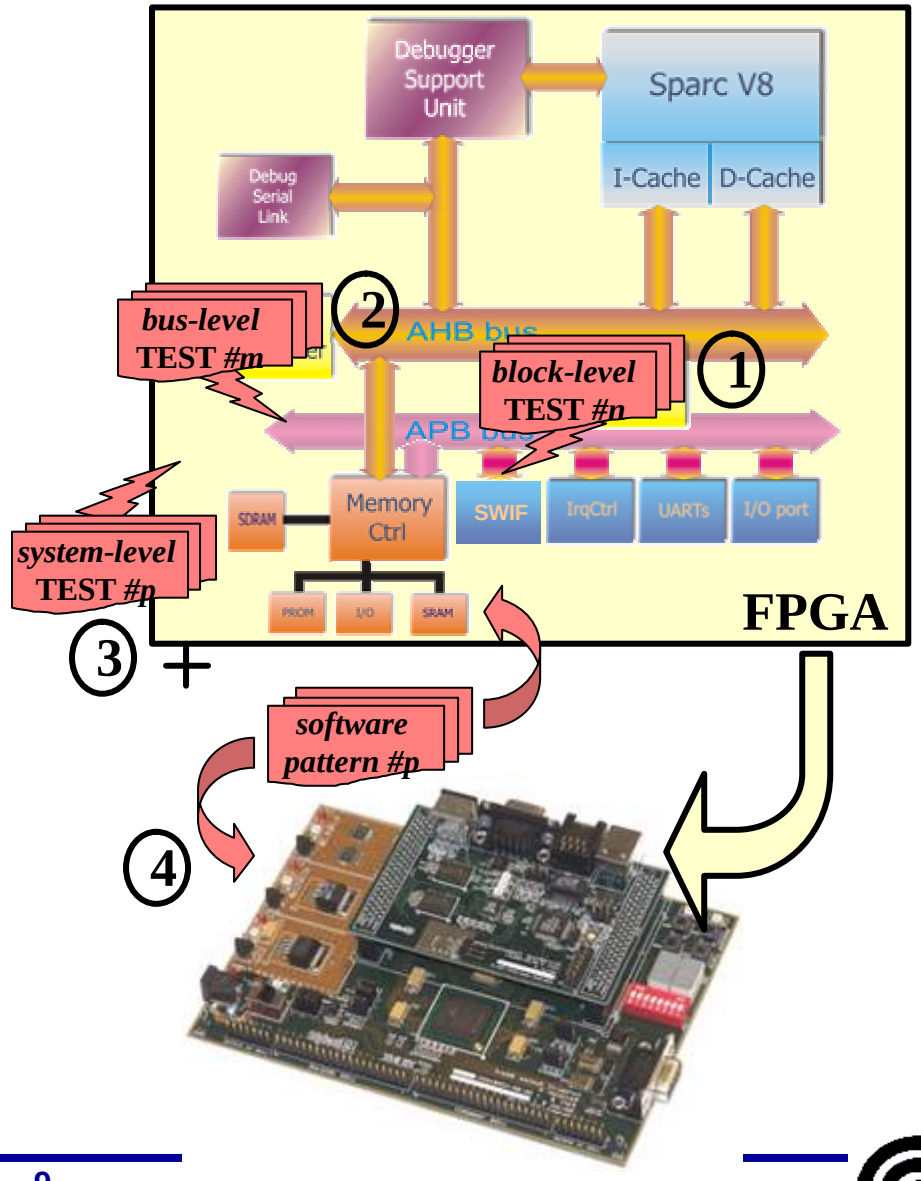
- *sub-block*
- *IP*
- *sub-system*
- *system*



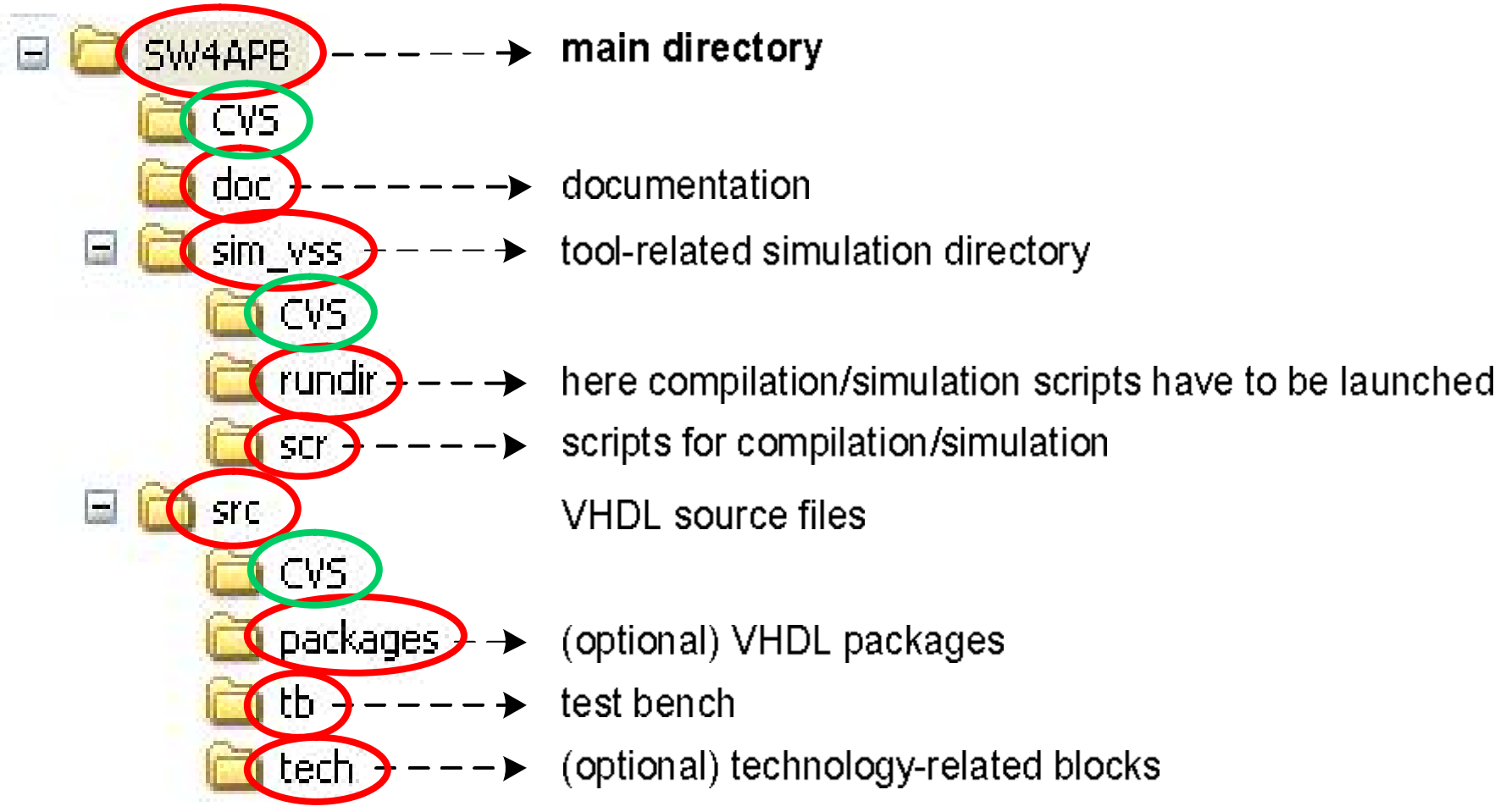
Identify the correct abstraction-level among *architectural*, *RTL*, *gate-level*, *layout*

Verification methodology 2/2

1) **block-level simulation**: the IP is verified as stand alone
2) **bus-level simulation**: the IP is verified with IPs having the same functionality or sharing the same bus
3) **system-level simulation**: the IP is verified once it is plugged on the whole system. The verification is mainly performed via software patterns
4) **hardware emulation**: the system is mapped onto a fast-prototyping board and the software patterns developed in step 3) are used



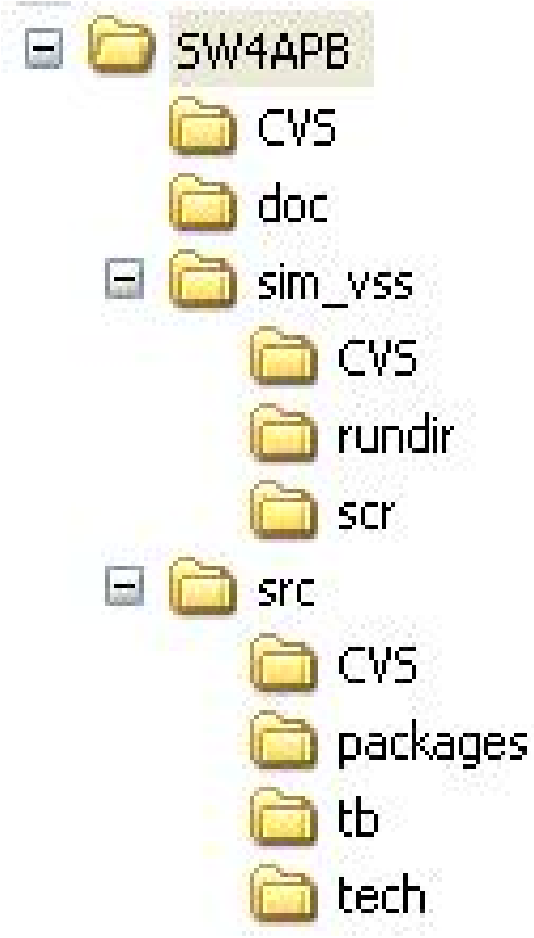
Database Structure 1/2



Sources under CVS control (Concurrent Versions System)

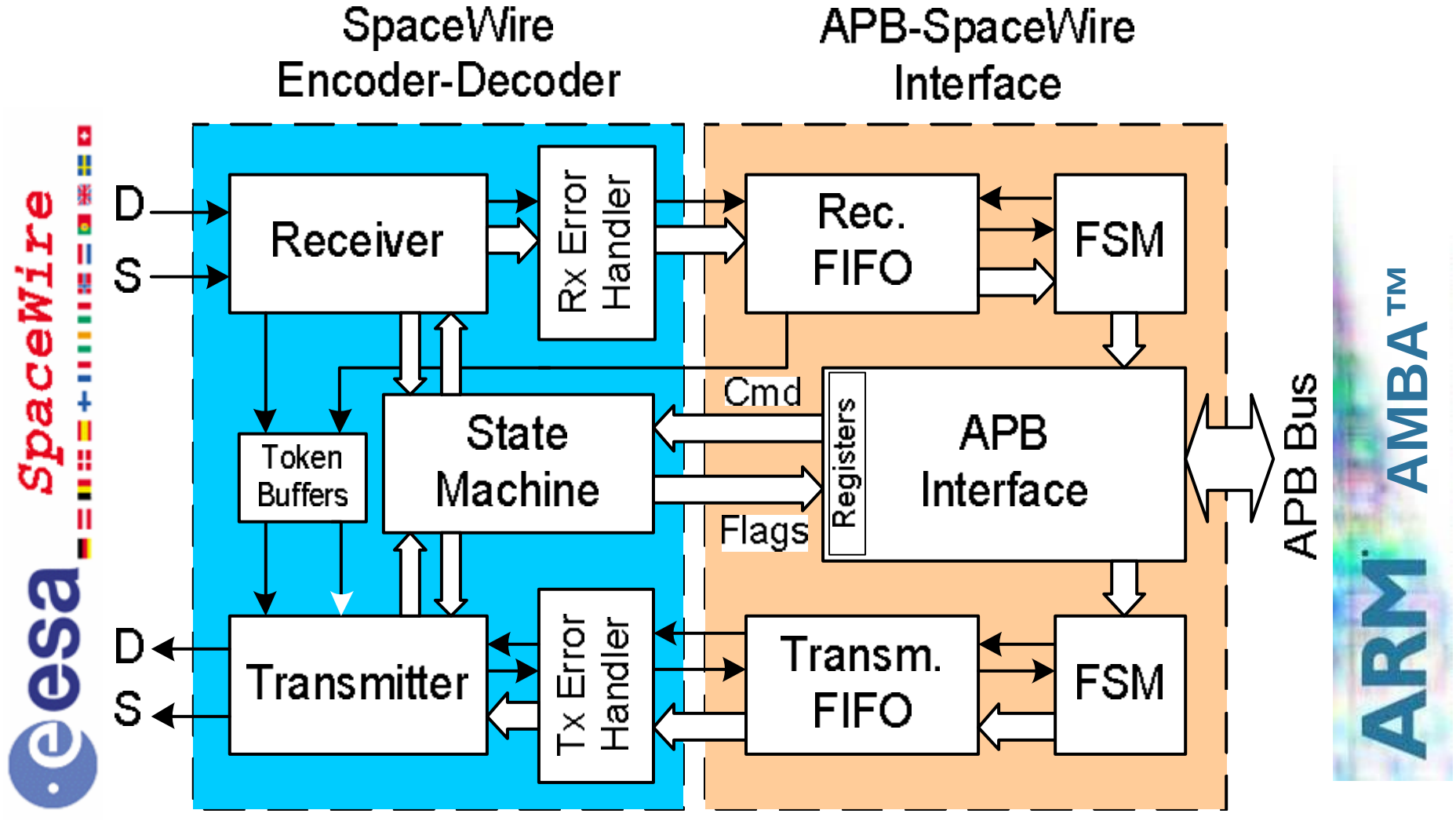
Database Structure 2/2

Well defined database structure leads to:

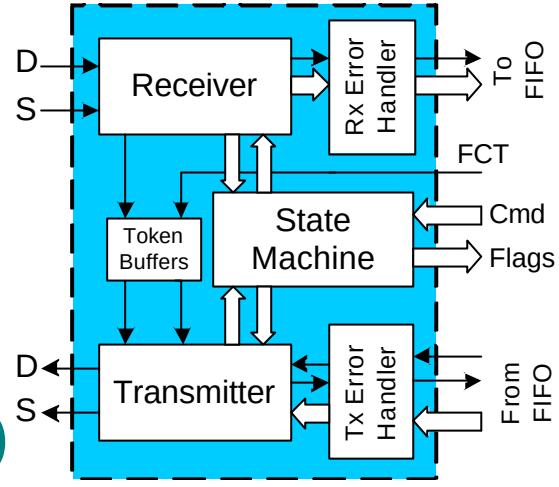


- ✓ Effective design and easy integration flow
- ✓ "Plug and Play" integration of IPs
  - Reuse of scripts for top level integration
- ✓ Control over synthesis and simulation errors
  - Top level script includes already verified lower level hierarchy scripts (IPs)

SpaceWire Interface



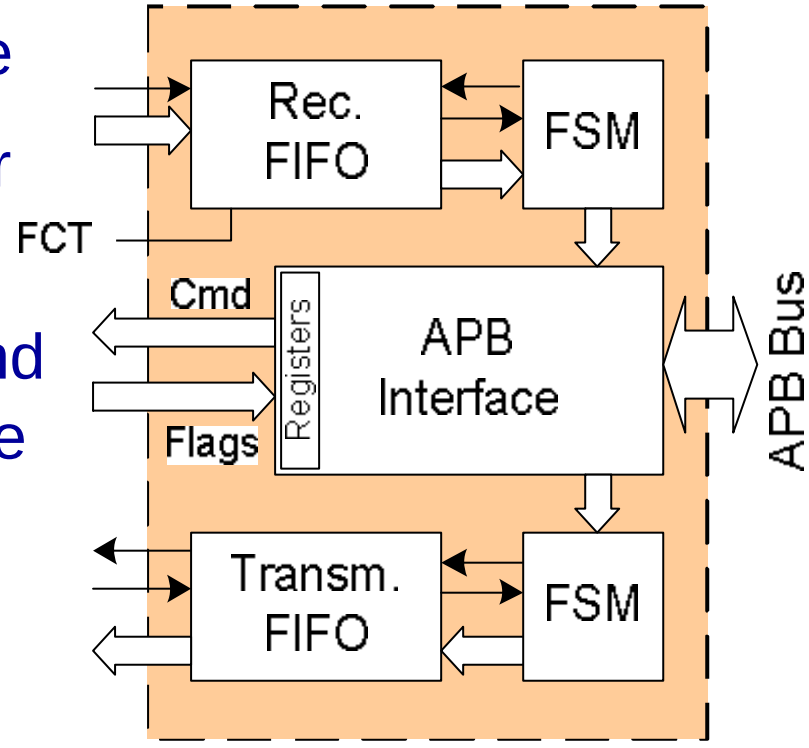
SpaceWire Encoder-Decoder



- ✓ Generic External Interface
- ✓ 3 clock domains
  - Receiver (recovered from DS)
  - Transmitter (derived from system clock)
  - State Machine and timers (system clock)
- ✓ Synchronization between clock domains
  - Token Buffers
- ✓ Error Handlers
  - Tx: to spill partially transmitted packets
  - Rx: to add EEP to partially received packets
- ✓ Extra signals to check the Status of the SpaceWire link

APB Interface

- ✓ AMBA™ 2.0 Compliant Interface
- ✓ 2 asynchronous FIFO buffers for Receiver and Transmitter
- ✓ Status registers for command and status signals to/from SpaceWire interface
- ✓ Two Normal Characters moved concurrently on APB bus



31	30:25	24	23:16	15	14:9	8	7:0
DAV 2	Res.	Character 2 Data/Control flag	Character 2	DAV 1	Res.	Character 1 Data/Control flag	Character 1

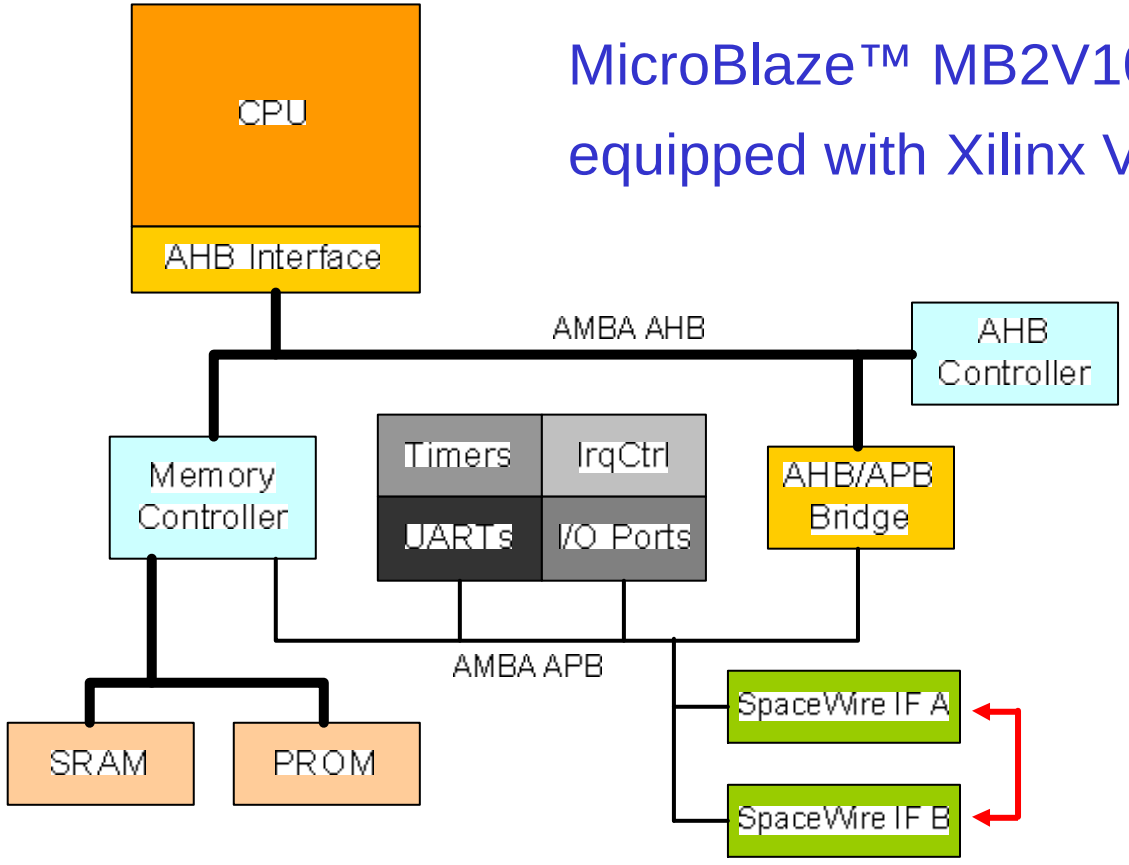
DAV used in read operations only

Synthesis Results

Component	Complexity	
	Xilinx LUTs	***Equiv. Gates***
Encoder-Decoder	275 LUTs	2 KGates
APB Interface	202 LUTs	1.5 KGates
Overall	477 LUTs	3.5 KGates

Maximum Clock Speed on *XILINX VIRTEX-II* FPGA
120 MHz

Functionality Tests



MicroBlaze™ MB2V1000 fast prototyping board equipped with Xilinx Virtex2 XC2V1000 FPGA

- APB functionality verified on LEON platform
  - APB write
  - APB read
- SpaceWire link
  - Data transfer between two I/Fs

Lack of a SpaceWire conformance test

Conclusions

- The use of a standardized SpaceWire link facilitates on-board connectivity with different equipment developed by different manufacturers
- Future SoCs for space applications call for an easily integrable SpaceWire IP Macrocell
- A SpaceWire interface compliant with AMBA™ APB bus has been presented
- A proper design and verification methodology in conjunction with a well defined IP database structure has been set-up and used aiming to further improve overall system reliability
- The SpaceWire interface has been tested using a LEON-based SoC platform
- Further functional tests are on-going. (SpaceWire Conformance Tester ?)