HW/SW co-engineering in ECSS-E-ST-40C Rev1 and ECSS-E-ST-20-40C standards

Agustin Fernandez Leon & Christophe Honvault

21/10/2020

Software and Hardware ECSS standards

- ECSS-E-ST-40C “Software” covers all aspects of space software engineering including requirements definition, design, production, verification and validation, transfer, operations and maintenance.
- ECSS-Q-ST-80C “Software Product Assurance” defines a set of software product assurance requirements to be used for the development and maintenance of software for space systems.

- ECSS-Q-ST-60-02C “ASIC and FPGA development” defines a comprehensive set of requirements for the user development of digital, analog and mixed analog-digital custom designed integrated circuits, such as application specific integrated circuits (ASICs) and field programmable gate arrays (FPGAs). It covers programme management, engineering and quality assurance.

HW/SW Co-Engineering and ECSS

- System-on-Chips start to be largely used in space systems. The links between the microprocessor executing the software and the FPGA implementing hardware logic need to be tightened
- The need for an optimized process to develop on SoC has been identified.
- There is a rare opportunity to implement this into ECSS:
  - Revision of the ECSS-E-ST-40C “Software” is on-going
  - New ECSS-E-ST-20-40C “ASIC and FPGA engineering” is in preparation, ECSS-Q-ST-60-02C will be updated accordingly.
- Working Groups of ECSS-E-ST-40C Rev1 and ECSS-E-ST-20-40C are communicating. First accomplishment: Definition of Software converging to:

software

set of instructions and data executed on a processing platform

ECSS-E-ST-40C Rev1

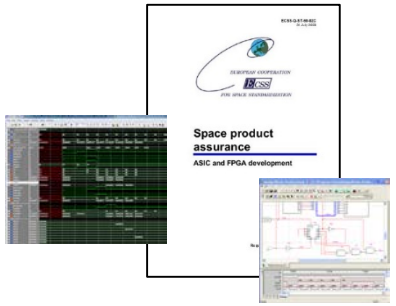
- A Change Request identifies the need of the introduction of a coordinated HW (FPGA) and SW process:
  - Early identification of the decomposition between Hardware and Software and required interfaces (e.g. similar to the SSS/IRD in ECSS-E-ST-40C): benefit of the best of the two worlds.
  - Identification of key points/deliveries during the design and development phases: what is defined/delivered and when, ensure a smooth development.
  - Identification of co-verification and co-validation activities: ensure the early identification of issues and the best correction to apply.
  - Definition of a consistent configuration management: ensure that both hardware and software are made to work together.
  - ...

ECSS-E-ST-20-40 “ASIC/FPGA Engineering” status

WG kick-off in July 2019:

2 years of work planned to have draft for public review by July 2021

WG is 12 members (ESA, CNES, ADS, TAS, IMEC, Ariane Group, GMV, RAL)
+ 12 experts (TESAT, ADS, ESA Microelectronics section)



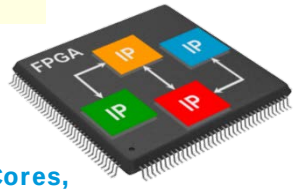
120 requests for changes gathered between 2017 and 2019 from industry and agencies => Engineering chapters and requirements of ECSS-Q-ST-60-02C under thorough review/improvement to create a new ECSS-E-ST-20-40. Revised ECSS-Q-ST-60-02 will have only PA requirements.

Progressing well, bit slower than planned, 5 new additional meetings were held to accelerate. 16 meetings held so far. Estimated 6 months delay due to WG debates to improve structure (new chapters) and content (better and new requirements), and working in two separate books: E (eng requirements) and Q (product assurance req) **-> PUBLIC REVIEW expected Q4-2021**

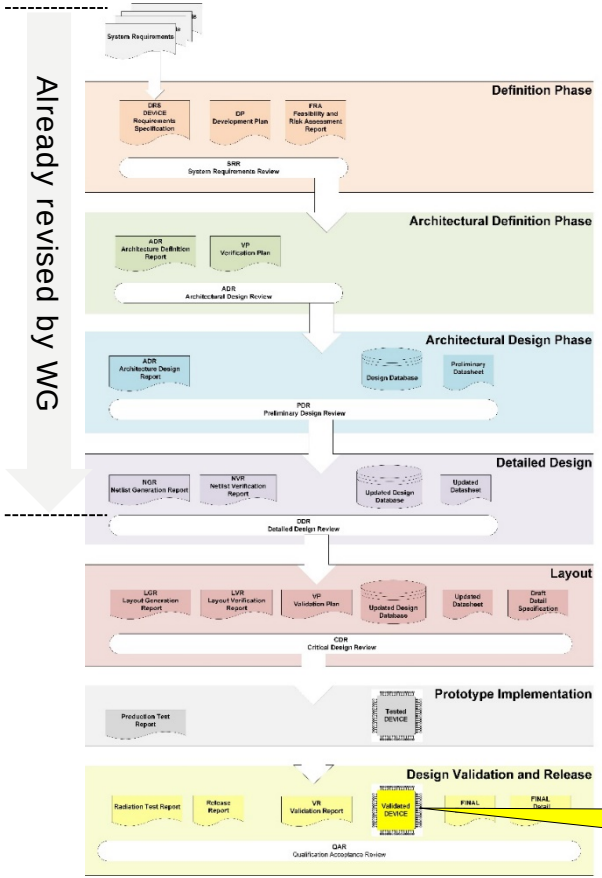
380 requirements under detailed review:

55% already reviewed and dispositioned

emphasis on: higher clarity,
covering different flows for ASIC (dig and analog), FPGAs and IP Cores,
consistent with SW ECSS-E-ST-40 /Q-ST-80 std,
facilitate tailoring per flow and criticality, including IP development



ASIC/FPGA Engineering development flow



The following **readable code and binary files** are used during several development steps of ASICs and/or FPGAs. - WARNING – they might look and feel like “software”, BUT THEY ARE NOT !

Hardware Description Language models of ICs (e.g. Verilog, VHDL, SystemC, etc.)

Scripts to automate and control many IC design steps (e.g. HDL code syntax analysis, behavioral simulations, netlist generation, timing analysis, place and route design rules checks, test mode insertion, mitigation techniques insertion, production test vectors generation, etc.) Those design steps involve many specialized CAD/EDA tools, requiring their own “constraint files” and/or “script languages”

Bitstreams to “programme” FPGAs (i.e. to fix the interconnections between all the FPGA internal circuits and the input/output pins)

the final product is an **Integrated Circuit**. The files above are used as “inputs” to the various “IC design tools” needed to create the microchip.

ECSS-E-ST-20-40
ASIC/FPGA/IP Engineering

Definition of term
“SOFTWARE”
final details to be
agreed for both
standards by both WGs

ECSS-E-ST-40
Software Engineering

3.2.23 software
set of instructions and data executed in a processing platform

NOTE 1 Hardware Description Language files are not software.

NOTE 2 integrated circuit models intended to generate HDL models are not software.

NOTE 3 bit streams used to programme FPGAs are not software.

3.2.29 software
set of instructions and data executed on a processing platform

NOTE 1: A processing platform can be hardware, e.g. a processor or software, e.g. a virtual machine or an interpreter.

NOTE 2: Some processing platforms only require data, e.g. configuration of state machines or configuration data of a neural network

Integrated Circuits embedding processing cores

So… what happens if the Integrated Circuit to be developed contains one (or more) "**processing core(s)**" that will use:

CASE 1: ANY "sets of instructions and data" compatible with the processor core(s) architecture(s), to be developed by an a-priori-not-known software team, to perform many functions (not known yet).

CASE 2: Several "set of instructions and data" to be developed by a known or unknown software teams, to perform several functions within several systems, in a several missions.

CASE 3: A specific "set of instructions and data" to be developed by a known software team, to perform certain functions within a specific system, in a specific mission

EXAMPLES

Standard ASIC/Microprocessor /FPGA **products:**
- AT697
- AT7913 (SpW-RTC)
- AT7991 (AGGA4)
- GR712
- GR740

→ LEON cores

- Blank NX1H140TSP (NG-Large) FPGA
- Blank NX2H540TSC (NG-Ultra) FPGA

→ ARM cores

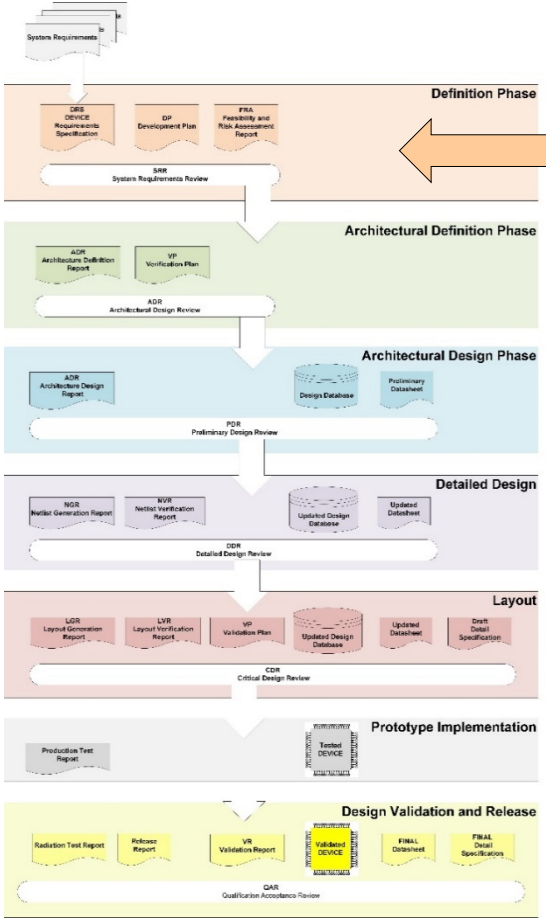
Proprietary ASICs/processors:
- MDPA ASIC
- EPICA ASIC
- SCOC3 ASIC

→ LEON cores

- DPC ASIC – OpenMSP430 cores
- HPDP ASIC – PACT XPP cores

FPGAs programmed with proprietary designs for a specific P/L or P/F system of a mission:
- Programmed NX Large or Ultra FPGAs – "hard" ARM cores
- Programmed Xilinx FPGAs – "hard" ARM cores
- Programmed Microchip, Xilinx or NX FPGAs with soft processor cores (e.g. LEON or RISC-V "soft" IPs)

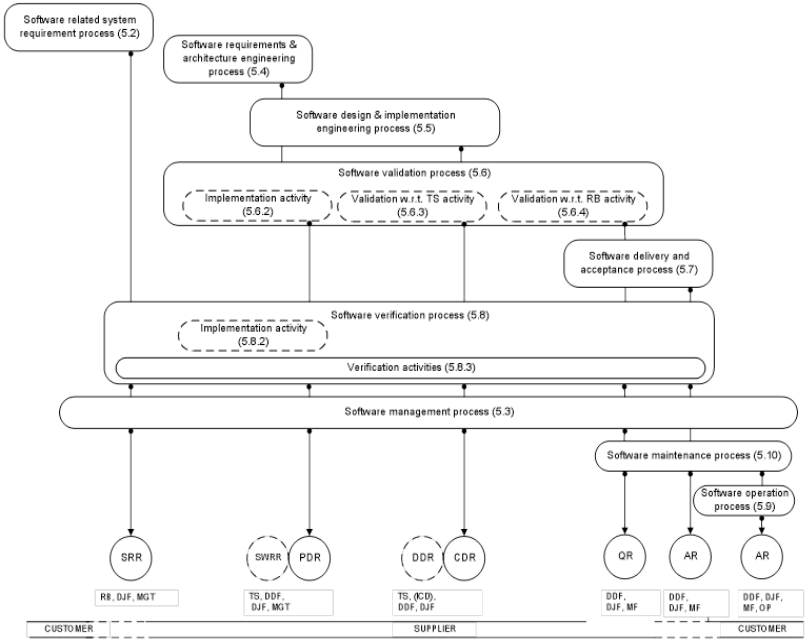
ASIC/FPGA Engineering development flow



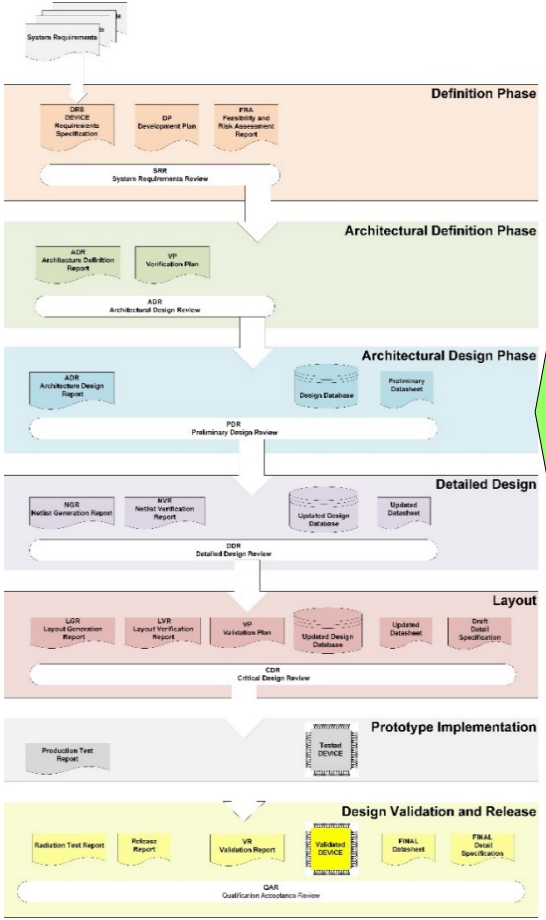
SOME FIRST QUESTIONS

1. standard product (multiple uses / SW) or proprietary (several or specific uses / SW)?
2. Which processor core(s)?
3. “hard cores” already embedded in an FPGA or “soft cores” to be introduced in the FPGA or new ASIC?
4. Do(es) the processing core(s) interact with other FPGA or ASIC functions? SW to take that into account…
5. Known or unknown SW developers?

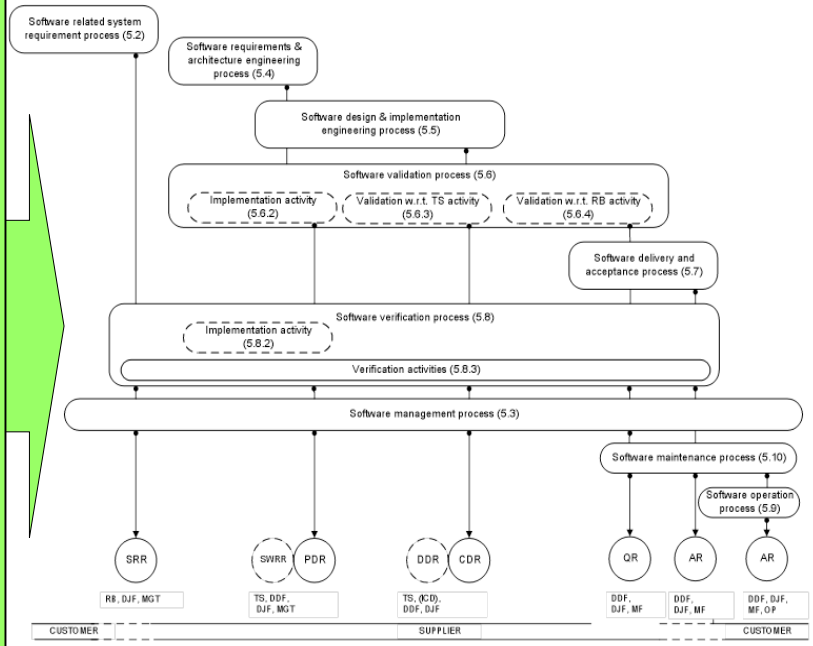
Overview of Software life cycle process



ASIC/FPGA Engineering development flow



Overview of Software life cycle process



SOME FIRST IDEAS for new requirements in both standards

1. Coordination and collaboration between the HW and SW development teams
2. Schedules of key milestones alignment whenever dependencies exist
3. Interaction of HW and SW teams for final definition (and compatibility) of HW and SW specifications
4. Co-simulation/verification of core processor(s) with SW
5. Co-validation (tests with IC prototypes) of IC and the SW