DOCUMENT

ESA IP Core Technical Requirements

Prepared by	**TEC-EDM IP Cores Service**
Reference	**TEC-EDM/2010.61/KM**
Issue	**3**
Revision	**1**
Date of Issue	**13/05/2022**
Status	**Issued**
Document Type	**TN**

European Space Agency
Agence spatiale européenne

APPROVAL

Title **ESA IP Core Technical Requirements**	
Issue 3	**Revision** **1**
Author **TEC-EDM IP Cores Service**	**Date** 13/05/2022
Approved by	**Date**

Table of contents:

1 LIST OF ACRONYMS

ASIC	Application Specific Integrated Circuit
COTS	Commercial Off The Shelf
EDA	Electronic Design Automation
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
HW-SW	Hardware-software
IP	Intellectual Property
RTL	Register Transfer Level
SEE	Single Event Effect
SEU	Single Event Upset
SoC	System-On-Chip
TLM	Transaction Level Model

2 INTRODUCTION

The objective of this document is to describe the technical requirements and the minimum set of deliverables expected, in order to allow a design to be reused as a synthesizable digital IP Core “Intellectual Property Core”, or soft IP Core, described at RTL level. These requirements can also be applied to high abstraction IP models of specified functionality (such as instruction and data processor, memories, busses, etc.) described in an appropriate modelling language such as SystemC/TLM. The term IP Core is used in this document to refer to both, RTL IP Cores and IP models.

The list of requirements is organised as follows: first a list of generic requirements applicable to both RTL IP Cores and IP Core models is provided. Then specific requirements for RTL IP Cores and IP models, in particular written in SystemC/TLM, are elaborated.

3 IP CORE TECHNICAL REQUIREMENTS

An IP Core should be portable to various designs and technologies, and easily re-usable by third parties, who were not involved into the development of the IP. It must therefore satisfy a number of requirements:

3.1 Generic IP Requirements

R1 Self-containment

The IP database (source code, test benches, scripts) must be self-contained. External dependencies (i.e. on elements, design units, packages, etc. which are not part of the same IP Core group of files and/or for which the user does not have the same IP rights) shall be avoided, unless these can be expected to be commonly and freely available to users, like e.g. IEEE packages generally supported by CAD tools, or FPGA specific simulation models (memory models etc.). For non-common external dependencies, such as ASIC technology simulation models: provide equivalent behavioural models allowing a limited execution of the self-standing IP.

Background IPR and Open Source code shall be considered an external dependency in case they cannot be delivered and further re-distributed with the same IP rights foreseen for the current development. Any other files or documents which are necessary to use the IP Core shall be identified in the IP Core documentation, detailing the versions to be used and the way to obtain them.

R2 CAD tool independence

The IP-Core code database shall not depend on any specific CAD tool. The compilation order and any library dependency shall be specified in a tool-independent way. Tool dependent constructs inside the code are only allowed if there is no other way to describe the same functionality, and they shall be appropriately signalled in code comments, documents and reviewed.

R3 Version Management

All the code and documents shall be available during the development on an internet-accessible concurrent version management system (such as Subversion or GIT). Release versions shall be properly tagged and identified with numeric values, for instance "_ver x.y" where 'x' is a major change/regressively incompatible and 'y' is minor/regressively compatible.

R4 Verification

Verification procedures shall exercise the IP Core code. Verification shall ensure that the implementation of both single components and full IP Core matches the intended functionality and meets the requirements.

R5 Code coverage

The verification tests shall cover 100% of the IP Core source code. Missed coverage shall be manually analysed and described in the Verification Report. For HDL code, coverage shall be measured on, at least, *statement*, *branch*, *FSM*, and *condition*.

R6 Testbenches

Testbenches shall be implemented as pass/fail tests, with automatic checking of the results and allowing batch and regression runs. Unlike what is specified in [2], testbenches might be written either using the VHDL or SystemC or another high-level verification language to be agreed upon; appropriate scripts have anyway to be provided to allow using the same testbenches (or a subset of those) with both VHDL and SystemC models. It shall be possible to set up and re-run all testbenches from scripts or makefiles with minimum user intervention.

R7 IP Verification Plan

The IP Verification Plan shall describe the complete verification environment and procedures in detail, including simulation testbenches and tests on a hardware prototype. All test cases shall be described with clear indications of the IP configurations on which they have been verified. The correspondence between the testbench code and the test cases described in this document shall be stated. A list of tools and operating systems (and their versions) used during the development/verification of the IP shall also be provided. The IP Verification Plan shall be based on the outline given in Annex E and F of [5]; in addition, a R*equirement Coverage Matrix* must be provided, specifying the correspondence between the requirements and the tests which verify them.

R8 IP Verification Report

The verification results shall be documented in a Verification Report, along with explanations for problems, and justifications for non-passed cases. Technology and environment related implementation and verification results, such as area, timing, SEU rates shall be summarised, and any problems which can be anticipated (e.g. memory initialisation during gate level simulation) shall be highlighted. Explanations for missed code coverage shall be provided.

3.2 RTL IP Core Requirements

R9 Coding Guidelines

The source code shall be written in conformance with modelling guidelines and coding style rules agreed between the developers and the Agency. For example, the ESA VHDL Modelling Guidelines [2] are applied for any new VHDL design as a baseline. Any exceptions to the agreed design and coding guidelines must be documented. In particular, exceptions related to the existence of asynchronous circuits, combinational inputs, or combinational outputs deserve special attention. Other modelling guidelines or coding style rules can be followed in agreement with the Agency. Such rules shall be documented and distributable to the Agency. The followed coding guidelines shall maximise design re-use and the amount of ASIC and FPGA technologies the RTL can be mapped to. With respect to what indicated in [2], the VHDL code of the IP Core shall have to adhere to the *IEEE Standard for VHDL Register Transfer Level (RTL) Synthesis*, [6]. Comments in the code shall be written to be readable by an automatic documentation system of choice (e.g. Doxygen, [10]).

R10 IP Database

The IP Core database shall contain, as a minimum:

- o Readme and Makefiles.
- o Complete, synthesizable source code for the design, and all its sub-blocks (in VHDL or Verilog).
- o VHDL/Verilog design examples that instantiate the IP core.

- Simulation scripts, compilation scripts and any command files used during the simulations.
- Testbenches, incorporating automated results checking, in the form of PASS/FAIL tests.
- Bus functional models/monitors used in the testbenches (if any).
- Simulation logs, code coverage reports.
- Implementation scripts, constraints files, and reports for the target technologies.

The organization of these files in folders and sub-folders shall be agreed with the Agency and shall be documented in the IP User Manual. The Agency has supported the development of a tool to facilitate the generation of a directory structure for IP Cores and the automation of the design flow. The tool is called Abeto and can be provided to IP Core developers. More information [13].

R11 **Clock and Reset**

Clock and Reset signals shall contain no logic; if really necessary, such logic shall be grouped in a well-defined module, such as a clock gating unit. All Flip-Flops shall be designed to have both synchronous and asynchronous reset signals. At IP-Core instantiation the user shall select (for example through *generics*) which of the two reset flavours to use; the selection shall be, if possible, the same for the whole IP-Core (i.e. it shall be avoided that in the same instantiation synchronous and asynchronous reset signals are mixed).

R12 **Clock Domain Crossing (CDC)**

Appropriate measures shall be taken for signals that cross clock domains to avoid metastability. These shall be documented in the Detailed Design Description. The use of CDC verification tools is encouraged. CDC shall if possible be localized in dedicated design units (synchroniser /FIFOs), rather than being 'diluted' in larger functional blocks.

R13 **IO pin requirements**

By default, there shall be no bidirectional pins (for external I/O’s, the IP must have separate input, output and enable pins). All outputs shall be registered. Deviations from these defaults must be avoided and, if they are really necessary, they must be documented in the IP Datasheet (see R21 below).

R14 **Functional verification**

Functional correctness of a design shall be demonstrated by running a sufficient set of simulation test benches. Test benches shall be made available in full source code, including auxiliary scripts, pattern files and reference models (e.g. in C, SystemC, or Matlab) used for test pattern generation or analysis. Console output during simulation shall be verbose and shall give clear reference to the test cases defined in the IP Verification Plan (see R7 above). Test benches shall be self-checking pass-fail wherever possible; when not possible, reference (“golden”) patterns are to be provided along with scripts comparing them with the simulation output. FPGA prototyping based on a COTS evaluation board is encouraged in order to complement and accelerate functional verification.

R15 Formal equivalence checking

It is encouraged to employ formal equivalence checkers to ensure correspondence between the behavioural, synthesized, mapped and P&R models. Using these tools can reduce the amount of post-synthesis or post-place and route simulations required. Formal equivalence checkers shall be always used when the target for IP Core implementation is ASIC.

R16 Assertion-based verification

Assertion-based verification is encouraged to maximise the functional coverage of a design.

R17 Timing Constraints

Timing constraints shall be provided, typically in SDC format, defining tentative IO constraints, clock constraints, clock relationships, false paths and multicycle paths. Gate-level simulations should be performed post-P&R for FPGA targets, and post-synthesis for ASIC targets. The list of technologies for which gate-level simulation is performed shall be agreed with the Agency.

R18 Verification of configurable IPs

In many cases, the IP Core will have configuration options (hard-coded or software programmable), which make an exhaustive verification of all combinations of these options impossible. A reasonably chosen subset of configurations has to be verified. Code coverage results from the different configurations should be merged in order to meet the overall coverage goals. To avoid leaving the user in doubt, it is crucial that the chosen subset is fully documented and justified in the IP Verification Plan document (see R7 above).

R19 Technology independence

The IP Core shall be developed as technology independent, synthesisable VHDL code. However, for certain functionalities (e.g. embedded memories) technology specific macrocells may need to be used. The choice of target technologies is then subject to a mutual agreement between the contractor and the Agency. These macrocells need to be clearly identified in the source code (via proper encapsulation) and the documentation, in order to ensure easy portability to other technologies. In any case, a ‘generic’ version of the IP shall always be provided, where the macrocells are replaced by behavioural simulation models.

R20 Target Technologies for Implementation

The IP-core shall be compatible with different ASIC and FPGA technologies. As a minimum, the implementation flow (synthesis for ASIC and FPGA, layout for FPGA) including timing verification shall be exercised for one ASIC technology, one reprogrammable FPGA and one

one-time programmable FPGA, see Section 5. The final choice regarding the target technologies shall be discussed and agreed upon between the contractor and the Agency.

The IP Core shall feature an easy way of configuring the target technology to be used, possibly integrated in the build-system.

R21 **IP Core Implementation Tables**

The technology mapping results for the IP Core shall be documented. All implementation details (FPGA resource use, total and by module, frequency achieved), shall be provided in tabular format. These tables shall provide evidence that the design can be appropriately mapped and the timing requirements are met for the target technologies, as agreed with the Agency.

R22 **IP Datasheet**

The Datasheet shall provide all architectural and implementation details of the IP Core, including descriptions of the functionality, interfaces, top level module, and every sub-module and their interconnections. It shall also describe the SEE Mitigation approach, see R24. It must also provide all the necessary information to allow the designer to integrate the IP Core into the target system. The IP Datasheet shall be distributable to the public. The IP Datasheet shall follow the outline depicted in [4], Appendix C.5, with the following amendments:

- The hard configuration options (constants or generic parameters to be defined prior to synthesis), which do not exist at component level, have to be documented in a dedicated section, ensuring clear separation between hard (at SoC design time) and soft (programmable registers or input pins) configuration.
- Any design techniques, implemented in the code, optimising certain electrical and environmental properties (e.g. fault tolerance by design, power saving modes etc.) shall be described along with the functional description.
- The pin list shall specify the timing relation for every pin, such as ‘synchronous with clock X’, ‘registered output’, ‘combinational through’, etc.
- The pin list shall indicate (in a typical application), which of the pins are for on-chip signals, and which are for external pads. In the latter case, the pad type shall be indicated (e.g. Schmitt trigger, open-drain, LVDS, pull-up).
- Logical interfaces (e.g. groups of pins representing an address, or the interface to a data bus) shall be clearly identified, and all the relevant characteristics specified (e.g. endianness, etc.).
- Mechanical, electrical (DC, AC, power) and environmental (thermal, radiation) specs are not applicable for a technology independent IP.
- ASIC layout and/or FPGA place and route guidelines should be provided for any timing critical signals.
- Recommendations regarding the clocking, CDC and reset strategies needed to drive the IP Core should be provided.

Annex G of [5] shall also be taken into account during the preparation of the datasheet.

R23 IP User Manual

The VHDL model hierarchy, and the corresponding file and directory structure, shall be described. Furthermore, the correspondence between the VHDL model structure and the design architecture described in the datasheet shall also be stated. There shall be a straightforward correspondence between the block names described in the documentation, and the directory/library structure and file names in the design database. Instructions on how to setup the design database shall be provided, and any specific paths or variable settings required shall be documented. Furthermore, instructions shall be given on how to initiate "design operations", e.g. compilation, simulation, synthesis, and timing analysis. An elegant solution is to provide a Makefile which, when called without argument, simply prints a list of all available targets starting up the various test benches, verification steps, etc. Brief explanation of the testbenches and detailed instructions on the steps necessary to run them shall be given. The IP User Manual shall be distributable to the public.

R24 Single Event Effects Mitigation

The effects of a SEE in the IP Core shall be studied in order to identify which parts of design are vulnerable and need protection. This can be done by analysis or by fault injection during simulation.

The design shall provide instruments to enable SEE hardening/mitigation measures implemented at either RTL level or during implementation (see [9], Section 7). Hardening/mitigation measures shall be implemented with configuration options ensuring that a non-hardened design can be created for implementation of radiation hardened technology or for prototyping purposes.

As a minimum, the following mitigation techniques shall be considered, as options that can be activated or inhibited, depending on the user needs:

a. Triple Modular Redundancy and Majority Voting in all flip-flops and latches.
b. Error Detection and Correction (EDAC) and/or parity bits for internal and externally used memory banks.

In addition, all Finite State Machines in the IP shall not have any redundant, unreachable or deadlock state (a state without output transitions to other states), and shall go back to a known state in the event of entering an illegal state or experimenting an illegal transition caused by an SEU.

R25 IP Database description

The IP database description will be shipped to the customer along with the database and shall effectively enable him to use the database and make the link between code and documentation. The database structure (directories, filenames, signals names) shall be sufficiently self-explanatory, so the database description is reduced to a minimum and becomes a simple README text file in the root directory of the database. As a minimum, the following information shall be available to the user:

- Setup of database, any required specific path or variable settings.

- How to start “design operations” (simulation, synthesis etc.) – an elegant solution is to provide a Makefile which, when executed without argument, simply prints a list of all available targets starting up the various test benches, verification steps etc.
- Link between VHDL model structure and the architecture described in the datasheet. In ideal case, there is an “easy-to-guess” correspondence between the hierarchy and block names described in the documentation and the directory/library structure and file names in the database.
- Link between test bench code (which test bench/generic parameter) and the test cases described in the verification document. List of tools and operating systems (and their versions) used during development/verification of the IP.

3.3 SystemC IP Core Requirements

R26 **Functional Behaviour**

The RTL version of the IP Core and the SystemC one shall have identical functional behaviour, i.e. given the same inputs, they shall produce the same outputs. Bit exact SystemC models are generally desired. Deviations can be accepted in order to accelerate simulations or the modelling itself, if justified by the development needs. In such case, the differences between the output of the VHDL model and the SystemC model shall be quantified, I.e. measuring the SNR, and kept below a margin to be agreed with the Agency.

The SystemC model shall include the effects of configuration registers and any corner case if present. Functional verification shall be performed using the developed test-benches as specified at requirement R6 and or any reference model of the functionality.

R27 **Timing Behaviour**

The difference in timing at the interfaces between the VHDL RTL version of the IP Core and the SystemC shall be measured.

R28 **SystemC IP Core User’s Manual**

The User’s Manual has a similar function for the SoC designer as the component datasheet has for the board designer, as it should allow him to design and integrate the IP/component into the system. It should clearly indicate the SystemC IP Core configuration options which have to be documented in a dedicated section.

R29 **Build System**

The SystemC IP Core shall include compilation scripts and shall use a scalable, configurable, multi-platform build system. The use of autotools is discouraged due to their difficult use in multi-platform environments.

R30 **Code Quality**

All the code shall be written according to ESA BSSC(2000)1 standard ([8]). The code elements (classes, attributes, methods, etc.) shall be thoroughly documented with a style compatible with the Doxygen (see [10]) documentation system; in addition documentation shall be added inside the body of methods, functions and procedure in order to ease the understanding of the behaviour of the code.

4 LIST OF DELIVERABLES

The following table lists the minimum set of deliverables to be expected as part of the IP core package (design database).

4.1 RTL IP database

Item	Description	Reference
Functional RTL description	Complete, synthesizable source code for the design, and all its sub-blocks (in VHDL or Verilog). VHDL/Verilog design examples that instantiate the IP core. Simulation scripts, compilation scripts and any command files used during the simulations.	R9, R1, R2, R13, R24
Verification environment	Testbenches, incorporating automated results checking, in the form of PASS/FAIL tests. Bus functional models/monitors used in the testbenches (if any). Simulation logs, code coverage reports.	R6, R14 to R18
Implementation environment	Synthesis scripts, constraints files, and reports for multiple technologies. Refer to Section 4.4 for further details.	Section 4.2, R19

4.2 RTL IP documentation

Document	Distribution	Description and reference requirement
Datasheet and User Manual	Public	R9 to R13, R21, R24, R23,R25,
Verification Document	ESA Internal, IP Core Users	Includes verification plan (R7) and report (R8). R14 to R18
IP Core Implementation Tables	Public	Includes Technology Mapping Results. R19, Section 4.4 Can be merged with Datasheet.

It must be possible to distribute the Datasheet and the User Manual to the public, without the need to sign any license/NDA. All documents must be provided also in an editable format.

4.3 SystemC IP database

Item	Description	Reference
Source code	Complete, source code for the design. SystemC design examples that instantiate the IP core. Simulation scripts, compilation scripts and any command files used during the simulations.	R29, R30
Verification environment	Testbenches, incorporating automated results checking, in the form of PASS/FAIL tests. Simulation logs, code coverage reports.	R6

4.4 SystemC IP documentation

Document	Distribution	Description and reference requirement
Datasheet and User Manual	Public	R28
Verification Document	ESA Internal, IP Core Users	Includes verification plan (R7) and report (R8). R29, R30

It must be possible to distribute the User Manual to the public, without the need to sign any license/NDA. The documents must be provided also in an editable format.

5 TARGET TECHNOLOGIES

As an indication, the following are some of the most commonly used technologies for space-related ASIC and FPGA developments:

- ASIC technologies
  - i. Microchip ATMX150RHA
  - ii. DARE rad-hard library on (UMC 0.18 um)
  - i. DARE 65 nm.
- FPGA
  - i. Microchip RTAX, RTSX (one-time programmable)
  - ii. Xilinx Virtex-4/5, Kintex Ultrascale KU060 (SRAM based reprogrammable)
  - iii. Microchip Pro-ASIC flash-based, RGT4, PolarFire (non-volatile reprogrammable)
  - iv. NanoXplore BRAVE FPGA Family (SRAM rad-hard reprogrammable)

6 REFERENCE AND APPLICABLE DOCUMENTS

[1] Keating, M.; Bricaud, P., "Reuse Methodology Manual for System-on-a-Chip Designs", 3rd edition, Kluwer Academic Publishers, 2002.

[2] "VHDL Modeling Guidelines", issue 1, September 1994, ESA/ESTEC, https://amstel.estec.esa.int/tecedm/website/docs_generic/ModelGuide.pdf

[3] Design and VHDL Handbook for VLSI Development CNES Edition https://logiciels.cnes.fr/sites/default/files/DCT-TV-AV-2015-04424-handbook_CNES_xml.pdf

[4] "*ASIC Design and Manufacturing Requirements*", ESA document WDN/PS/700 http://microelectronics.esa.int/asic/DesignReq.pdf

[5] ECSS-Q-ST-60-02C – "ASIC and FPGA development" http://www.ecss.nl

[6] IEEE Standard for VHDL Register Transfer Level (RTL) Synthesis (IEEE Std 1076.6™-2004 or later).

[7] 1666-2005 - IEEE Standard SystemC® Language Reference Manual http://standards.ieee.org/findstds/standard/1666-2005.html http://www.systemc.org/downloads/standards/

[8] ESA C and C++ Coding Standard BSSC(2000)1 ftp://ftp.estec.esa.nl/pub/wm/anonymous/wme/bssc/bssc2000(1)i10.PDF

[9] Lessons Learned from FPGA Development Version 0.2, September 2002 http://microelectronics.esa.int/asic/fpga_001_01-0-2.pdf

[10] Doxygen http://www.doxygen.org/

[11] Blue Pearl Software including ESA Coding guidelines https://www.bluepearlsoftware.com/european-space-agency-blue-pearl-software-and-adiuvo-engineering-partner-contract-to-improve-the-usability-of-esa-soft-cores/

[12] VHDL Tool for code quality, CNES https://logiciels.cnes.fr/en/content/vhdltool https://github.com/VHDLTool/Sonarqube-Rulechecker-Demo

[13] Automated benchmarking tool for ESA IP Cores https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Microelectronics/Automated_BEnchmarking_TOol_ABETO_for_ESA_IP_Cores