IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 25, 2021, accepted September 9, 2021, date of publication September 22, 2021, date of current version October 1, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3114502*

# Development and Testing on the European Space-Grade BRAVE FPGAs: Evaluation of NG-Large Using High-Performance DSP Benchmarks

**VASILEIOS LEON[1], IOANNIS STAMOULIAS[1,2], GEORGE LENTARIS[1], DIMITRIOS SOUDRIS[1], (Member, IEEE), DAVID GONZALEZ-ARJONA[3], RUBEN DOMINGO[3], DAVID MERODIO CODINACHS[4], AND ISABELLE CONWAY[4]**

[1]School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Athens, Greece
[2]Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 15784 Athens, Greece
[3]Space Segment and Robotics Business Unit, GMV Aerospace and Defence SAU, 28760 Madrid, Spain
[4]European Space Research and Technology Centre, European Space Agency, 2201 AZ Noordwijk, The Netherlands

Corresponding author: Vasileios Leon (vleon@microlab.ntua.gr)

This work was supported in part by the National Technical University of Athens via the Project i-CoVo: Implementation of Computer Vision Algorithms on Embedded Architectures under Grant 950032, and in part by the European Space Agency via the Activity QUEENS2: Quality Assessment of The New European Large BRAVE FPGA Software Tools under Grant 4000128041/19/NL/AR/va.

**ABSTRACT** The advent of space applications with increased computational requirements has led the space industry to consider innovative chips and avionics architectures for high-performance on-board data processing. In a relatively limited market, the European *BRAVE* family of Field-Programmable Gate Arrays (FPGAs) offers such novel radiation-hardened solutions. Towards verification, the current work devises and applies a methodology to thoroughly assess the BRAVE FPGAs and their SW tools. The paper focuses on NG-Large, i.e., the largest FPGA of the 65nm Radiation-Hardened-By-Design (RHBD) technology of NanoXplore to date. The proposed approach comprises a number of customized steps to systematically evaluate the entire FPGA design flow. Initially, we carefully select and tune a set of high-performance Digital Signal Processing (DSP) & Computer Vision (CV) benchmarks, which were originally developed as Hardware Description Language (HDL) IPs in past projects of the European Space Agency (ESA). Subsequently, we perform exhaustive exploration of the Synthesis, Placement, and Routing stages of the SW tools, as well as testing on actual HW boards. At each step, we generate and analyze a variety of results, while we also compare them to 3rd-party solutions. The results show that NG-Large provides sufficient programmability and performance, e.g., classic CV IPs for feature detection on megapixel images can achieve a throughput of 5–10 frames per second, while the on-chip memory utilization is up to 56% better than that of 3rd-party FPGAs. As a highlight, at system-level, we successfully implement and execute an entire HW/SW algorithmic pipeline for Vision-Based Navigation (VBN) involving SpaceWire data transfers with LEON CPU & NG-Large co-processing.

**INDEX TERMS** Assessment methodology, computer vision, digital signal processing, European FPGA, HW benchmarking, NanoXplore, radiation-hardened, space applications, tool testing.

## I. INTRODUCTION

The emerging need for increased computational power and fast data transfers in modern space applications makes the use of high-performance devices in on-board processing systems critical. Radiation-hardened CPUs, such as the PowerPC-based RAD750 (12W@200MHz) and LEON2-based AT697F (1W@100MHz), seem unable to meet the performance requirements of future missions. Towards more power-efficient and high-performance computing, the space industry explores new processing platforms. In general, the RHBD Application-Specific Integrated

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

Circuits (ASICs) provide the best performance, however, they are very costly and lack in terms of re-configuration and flexibility. The FPGAs are an attractive alternative solution providing excellent performance-per-power ratio and re-configuration capabilities [1]. In this context, the new *NanoXplore* [2] RHBD family of FPGAs [3], [4], also known as *BRAVE*,[1] is expected to play a key role in upcoming space missions, especially in Europe. The BRAVE family offers high resource density on FPGA and SW tools for end-to-end development and seamless re-configuration.

The enhanced performance of FPGAs will facilitate embedded computing in space, e.g., for Earth Observation (EO) and Vision-Based Navigation (VBN), towards a decreased need for downlink transmission of sensor data to the ground stations. The space community has already conducted relevant research on space-grade and Commercial-Off-The-Self (COTS) FPGAs. More specifically, FPGAs are constantly being evaluated for future missions, either as main accelerators [5]–[8] or framing processors & heritage accelerators [9], [10]. The literature also includes numerous works with FPGA-based co-processing architectures for space avionics [11], [12]. In terms of algorithms, FPGAs are used for accelerating DSP/CV functions, as well as for implementing data transcoding for instruments/sensors (e.g., via SpaceWire/SpaceFibre [13]) and data compression [14], [15].

The number of available space-grade FPGAs is limited and becomes even smaller when opting for European high-density chips. The major space-grade FPGAs, in the market today, are categorized per vendor as follows:

- Xilinx [16]: Virtex-4QV (SRAM, 90nm), Virtex-5QV (SRAM, 65nm), RT Kintex UltraScale (SRAM, 20nm).
- Microsemi [17]: RTSX-SU (anti-fuse, 250nm), RT ProASIC3 (flash, 130nm), RTAX-S/SL (anti-fuse, 150nm). RTG4 (flash, 65nm), RT PolarFire (SONOS, 28nm).
- Atmel [18]: AT40K (SRAM, 350nm), ATF280 (SRAM, 180nm).

All the above FPGAs vary with respect to resource density, performance capabilities and radiation resilience. The BRAVE family constitutes a new promising addition in this pool. NanoXplore [2] provides multiple low-end and high-end RHBD SRAM-based FPGAs [3], [4], i.e., *NG-Medium* (65nm), *NG-Large* (65nm) and *NG-Ultra* (28nm), with all the traditional FPGA programmable logic resources. NG-Large and NG-Ultra integrate the single-core ARM Cortex-R5 and quad-core ARM Cortex-R52 processors, respectively, with the latter implementing the DAHLIA System-on-Chip (SoC). Furthermore, the BRAVE FPGAs include features that are essential for on-board embedded computing in space, such as the SpaceWire interface for fast I/O data transfers and memory scrubbing to ensure continuous error-free functionality. The BRAVE FPGAs can be configured via multiple interfaces, i.e., JTAG, SPI/flash, and SpaceWire.

[1]BRAVE: Big Re-programmable Array for Versatile Environments.

Table 1 presents an overview of the most prominent space-grade FPGAs, including BRAVE. The Atmel FPGAs are omitted here due to their limited on-chip resources. The examined FPGAs can be categorized in three classes; each class includes chips from different vendors but with similar resources, e.g., {Virtex-5QV, RTG4, NG-Large}. We note that only the BRAVE family offers hard embedded processors, and hence, increases the development flexibility by facilitating HW/SW co-design. In terms of radiation resilience, similarly to Virtex-5QV, the advantage of BRAVE is their RHBD SRAM-based chips: they are fabricated with 12-transistor (12-T) configuration memory cells outperforming the simple SRAM cells and competing with anti-fuse and flash technologies. This is extremely important, considering that, for example, Virtex-4QV would require full Triple Modular Redundancy (TMR) and configuration memory scrubbing to achieve reliable operation.

The efficient utilization of a new FPGA family such as BRAVE, as well as the full exploitation of the associated SW tools, require a systematic and disciplined approach. For this reason, ESA is supporting a set of research activities, which involve HW benchmarking on the BRAVE FPGAs and testing of the BRAVE SW tools. These activities aim to improve the NanoXplore tools and devices, and evaluate the suitability of the BRAVE solution as on-board data processor. In the ''QUEENS-FPGA'' activity [19], we evaluated the NG-Medium FPGA based on our assessment methodology. In the current paper, we enhance our methodology and evaluate the next chip of the BRAVE series, i.e., NG-Large. The work is performed in the context of the ''QUEENS2'' activity of ESA.

The proposed quality assessment methodology is based on the systematic verification and testing of the NanoXplore tools for FPGA development & configuration. One of the key aspects in our methodology is the high-performance benchmarking with HDL IP cores from the signal processing and computer vision domains, which are representative of the performance requirements in rovers/spacecraft. The contribution of this paper lies in (i) introducing an enhanced version of our assessment methodology for evaluating new devices/tools, (ii) evaluating the NanoXplore SW tools by examining the available options throughout the entire FPGA design flow, (iii) evaluating the NG-Large capabilities as on-board processor with representative VBN benchmarks.

The paper is organized as follows. Section II includes related work with BRAVE and other space-grade FPGAs. Section III presents the BRAVE FPGA architecture and SW tool. Section IV describes our quality assessment methodology. Section V presents experimental results with DSP/CV benchmarks. Section VI presents system-level results for an entire VBN pipeline. Section VII draws the conclusions.

## II. RELATED WORK

### A. EVALUATION OF SPACE-GRADE FPGAs

Space applications impose strict requirements for radiation, thermal & vibration resilience compared to the terrestrial

**TABLE 1. Overview of space-grade FPGAs [2], [16], [17].**

| Vendor | FPGA | Rad. Resilience | Technology | Processor[1] | Logic Cells | Total RAM | DSPs | User I/Os | SERDES[2] | TID / SEL[3] |
|---|---|---|---|---|---|---|---|---|---|---|
| Xilinx | Virtex-4QV | Tolerant | SRAM, 90nm | – | 55–200K | 4.1–9.9Mb | 32–192 | 640–960 | – | 300 / 125 |
| | Virtex-5QV | Hard | SRAM, 65nm | – | 131K | 12.3Mb | 320 | 836 | 18@4.3Gbps | 1000 / 125 |
| | RT Kintex US | Tolerant | SRAM, 20nm | – | 726K | 38Mb | 2.7K | 620 | 32@12.5Gbps | 100 / 80 |
| Microsemi | RTAX-S/SL | Tolerant | anti-fuse, 150nm | – | 2–40K | 0.05–0.5Mb | 0-120 | 198–840 | – | 300 / 117 |
| | RTG4 | Tolerant | flash, 65nm | – | 151K | 5Mb | 462 | 720 | 24@3.1Gbps | 100 / 103 |
| | RT PolarFire | Tolerant | SONOS, 28nm | – | 481K | 33Mb | 1.4K | 584 | 24@10.3Gbps | 100 / 80 |
| NanoXplore | NG-Medium | Hard | SRAM, 65nm | – | 34K | 2.9Mb | 112 | 566 | – | 100 / 60 |
| | NG-Large | Hard | SRAM, 65nm | Cortex-R5 | 137K | 10.1Mb | 384 | 684 | 24@6.3Gbps | 100 / 60 |
| | NG-Ultra | Hard | SRAM, 28nm | Cortex-R52 | 536K | 34Mb | 1.3K | 744 | 32@12.5Gbps | 50 / 60 |

[1] Hardcoded embedded processor.
[2] High-performance serialization/deserialization (SERDES) transceivers.
[3] Radiation testing: Total Ionizing Dose (TID) in Krad (Si) and Single-Event Latch-up (SEL) immunity to linear energy transfer in MeV-cm$^2$/mg.

ones. To achieve increased reliability, space-grade FPGAs are used instead of their COTS counterparts, sacrificing the performance of the latest COTS technology node. The literature includes numerous works with space-grade FPGAs. These works focus on the implementation of functions used in on-board computing systems, the development of new space applications, and the design of novel processing architectures.

In [20], the CCSDS 1.2.3 standard for compressing hyperspectral images is implemented on Virtex-4QV, achieving real-time compression for sensors such as AVIRIS ($680 \times 512 \times 224$ image), while consuming 1/3 of the chip resources and limited power. Similarly, the Virtex-5QV and RTG4 FPGAs have been used for the implementation of the SHyLoC 2.0 CCSDS 121 and 123 lossless compression standards [14], providing up to 138 and 81 MSamples/s, respectively, for the AVIRIS sensor. In [21], the authors implement a single-chip payload data processing unit on the Virtex-5QV FPGA, integrating both the instrument system supervisor and data processing functions. The proposed architecture supports self-configuration management and mitigation techniques to provide fault-tolerance. In [22], the new SCCC-X telemetry transmitter, which is an extension of the CCSDS 131.2-B-1 standard, is implemented and evaluated on RT Kintex UltraScale and RTG4, delivering more than 450 MSym/s and 250 MSym/s, respectively. Very recently, radiation-tolerant FPGA-based platforms for AI applications have been proposed. In [23], a deep learning architecture for RT Kintex UltraScale is proposed, which is based on Xilinx's deep learning processing unit, TMR MicroBlaze subsystem, and Single Event Upset Mitigation (SEM) IP. In [24], the authors use the VectorBlox software development kit to deploy AI models on a matrix processor implemented on the RT PolarFire FPGA.

The literature includes several works involving the BRAVE FPGAs. In [25], the re-configuration capabilities of NG-Medium via the SpaceWire interface are evaluated, while in [19], benchmarking results for NG-Medium are reported and re-configuration scenarios with different algorithms are examined. Regarding NG-Large, benchmarking and tool testing results are provided by the current paper's authors in [26]. Moreover, NG-Medium and NG-Large have been used for the implementation of HW/SW pipelines for rover localization and mapping [27]. In [14], both FPGAs have been evaluated for hyperspectral image compression.

### B. USE OF SPACE-GRADE FPGAs IN MISSIONS

A review of the FPGA usage in space missions is crucial to understand the flight heritage and the target applications, as well as the evolution towards the future of the industry.

Starting with Microsemi, RTSX-SU has flight heritage since 2005, as it was on-board the Mars Reconnaissance Orbiter spacecraft. Other missions using this chip are the GPS-2RM, New Horizons, SAR-Lupe 1 and 2, Galileo GIOVE-A, and TerrSar X. RTAX is also widely used, having flight heritage since 2007 in missions such as the Mars Science Lab Curiosity rover. Moreover, RTAX-S is on-board in COSMO SkyMed 1 and Mars Phoenix. RT-ProASIC3 has flight heritage since 2013, i.e., in NASA's IRIS mission, being the first radiation-tolerant flash-based FPGA used in space.

Many FPGAs are used for implementing mass memory controllers and SpaceWire links and routers. Exomars 2022 OBC1 [28], developed by Crisa, includes two (four for redundancy) RTAX2000S for interfacing, reconfiguration and mass memory control. OBC1 manages the whole ExoMars 2022 mission during Cruise, EDL and Mars Surface Operation phases, and runs the mission's SW (developed by TAS) including the Guidance, Navigation and Control (GNC) subsystem (implemented by GMV). Moreover, RTAX2000 is used for the MDP/MDR in ARASE mission [29], which is carried out by JAXA. The Mission Data Recorder (MDR) and Mission Data Processor electrical box (MDP-e) are connected to the SpaceWire bus. When targeting higher performance and larger capacity, the RTG4 FPGA is a good solution, offering re-configuration capabilities up to a certain number of cycles (~300). This relatively new FPGA is incorporated in different on-going missions, e.g., in ESA's HERA, where it implements the interfaces and the SpaceWire router controller of the main On-Board Computer (OBC). Moreover, it is employed in JenaOptronic RVS3000-3D [30], which is a pose estimation high-performance in-orbit computing platform, integrating LIDAR and image processing into a single box. RTG4 is the main processing device,

taking over the LIDAR data extraction and algorithm execution.

Regarding the SRAM-based solutions, Xilinx FPGAs are employed in the Dawn Framing Camera (DawnFC) [31] and Venus Express Monitoring Camera (VMC) [32]. In DawnFC, Virtex-4QV is used for implementing Data Processing Unit (DPU) functions, such as the LEON2 processor running the RTEMS real-time operating system, application-specific co-processor for image processing, and sensor interface ports. A similar FPGA-based DPU processor is implemented in VMC. In ESA's Solar Orbiter, a multiprocessor architecture for high-performance floating point operations is implemented in the PHI instrument, which carries out the scientific analysis of the mission [33]. This SIMD architecture is used as an accelerator within the DPU, which is based on the LEON processor and two Virtex-4QV XQR4VSX55 FPGAs. Furthermore, both Virtex-5QV and NG-Medium are used in the HERA mission [34], [35], where GMV is in charge of developing the GNC subsystem. The hardware solution proposed by GMV is the Image Processing Unit (IPU), which is a dual-purpose avionics processing board for image processing and interface control (including the management of the interfaces with OBC and in-flight re-programming). The IPU provides isolation for the image processing and interface functions, relying on an FPGA-based architecture with allocated external volatile and non-volatile memories. NG-Medium is responsible for controlling and monitoring the interfaces, and Virtex-5QV executes computer vision algorithms, i.e., feature detection & tracking and calculation of brightness centre & maximum correlation with a Lambertian sphere, to provide the position of the mission's target asteroid in the field-of-view of the camera.

Xilinx space-grade FPGAs have been extensively used in Mars missions, and specifically, the Mars rovers [36], e.g., in Mars Exploration Rover (MER) Opportunity and Spirit. In this mission, NASA used space-grade Virtex FPGAs for both landing and on-surface operations of the rover, as well as for supervising the wheel motors, cameras, and arms. In the Curiosity rover of NASA, radiation-tolerant Virtex FPGAs implement the imaging pipelines and a MicroBlaze soft-processor core. More recently, in the Mars2020 rover (Perseverance) [37], NASA uses Virtex-5QV as Computer vision Accelerator Card (CVAC) to aid in landing navigation and autonomous driving on the Martian surface by accelerating stereo and visual tasks, e.g., image rectification, filtering, detection and matching. In the same rover, Virtex-4QV XQR4VFX60 is integrated in the Planetary Instrument for X-ray (PIXL) [38], aiming to identify chemical elements at a tiny scale. In this context, Virtex-5QV is also part of several components [39], such as the Iridium Next reconfigurable processor, NovaSAR 2018 spaceborne radar, 2019 Hellas-Sat 4/SGS1 communication satellite, 2022 NISAR–NASA/ISRO radar imaging, 2016 OSIRIS-Rex, 2023–2025 NASA/JPL Europa mission, and ExoMARS 2020 European rover (scheduled to launch in 2024).

All the aforementioned missions use space-grade FPGAs that are provided by US vendors. As an alternative promising solution, the European RHBD BRAVE FPGAs, which have been prototyped very recently (e.g., NG-Medium in 2016 and NG-Large in 2019), are gaining momentum. As mentioned before, NG-Medium is already included in the IPU of HERA [35]. In parallel to the HERA mission, GMV is examining a full European IPU, by replacing Virtex-5QV with NG-Large (the DC/DC converter will be European as well). NG-Large will accommodate the implementation of an edge detector IP module, feature tracking, centre of brightness calculation, and a long-range image processing technique to detect faint targets of pixel/subpixel sizes in the camera image. NG-Medium's first potential use in space will be on-board of the Payload-X/XL [40], which should have been already launched, however, it is on-hold waiting to launch with the optical EO satellite Amazonia-1.

Finally, several missions are considering COTS FPGAs due to their higher performance, and also, because the Single-Event Upset (SEU) rates are decreased along with the transistor feature sizes [41]. This kind of "New Space" approach for on-boarding COTS FPGAs is mostly found in CubeSats for Low Earth Orbit (LEO) non-critical operations [42]. Examples of COTS-based processing platforms are Xiphos Q8S (includes Xilinx's Zynq UltraScale+), AAC Clyde Space Kyrten-M3 (includes Microsemi's SmartFusion SoC), and Innoflight cfc-400 (includes Xilinx's Zynq UltraScale+).

## III. NanoXplore BRAVE FPGAs & SW TOOLS

### A. NG-LARGE FPGA

NG-Large is a RHBD SRAM-based FPGA manufactured on the 65nm STM C65-Space process technology [3]. Its fabric architecture is illustrated in Figure 1.

The die features 7 rows of 48 Tiles, with a single Tile consisting of 384 4-input Look-Up-Tables (LUTs), 384 D Flip-Flops (DFFs), 96 1-bit Carry Logic (CYs), 24 X-LUTs, and 2 64 × 16-bit Register Files (RFs). Each Tile includes 384 Functional Elements (FEs), with a single FE integrating 1 LUT–DFF pair and additional logic for CYs, X-LUTs and RFs. The CYs of NG-Large combine 1 LUT with carry propagation logic to support up to 96-bit carry chain. To allow the implementation of up to 16-input logic functions, 4 LUTs drive the inputs of another LUT (called X-LUT) without routing through the interconnect network. The RF of NG-Large is a synchronous dual-port SRAM with read-only & write-only ports and optional output register.

Furthermore, the chip die features 4 rows of 48 48-Kbit RAM Blocks (RAMBs) and 4 rows of 96 Digital Signal Processors (DSPs). Each RAMB is a true dual-port SRAM with optional output register and supports multiple memory configurations, i.e., 48K × 1-bit, 24K × 2-bit, 12K × 4-bit, 6K × 8-bit, 4K × 12-bit and 2K × 24-bit, as well as Error Detection and Correction (EDAC) configuration. A single DSP includes a 19 × 24 multiplier, a 56-bit ALU, an 18-bit

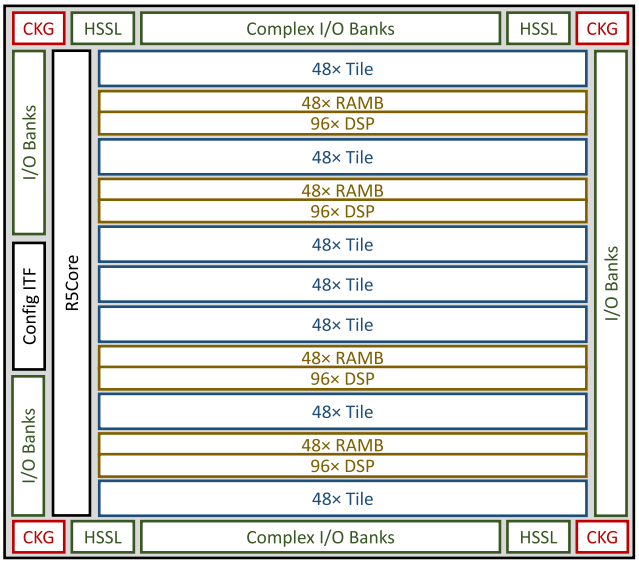



**FIGURE 1.** The fabric architecture of NanoXplore NG-Large FPGA [2].

pre-adder, as well as pipeline registers. The DSPs operate either in signed or unsigned mode, and can cascade up to 96 blocks.

NG-Large includes 4 Clock Generators (CKGs), one at each die corner. The CKG block includes 1 Phase-Locked Loop (PLL) and 10 Waveform Generators (WFGs), i.e., frequency dividers. Moreover, it has 4 High Speed Serial Links (HSSLs) of 6 lanes, providing up to 6.25Gbps data rate.

Overall, the total resources of NG-Large are summarized as: 137088 LUTs, 129024 DFFs, 32256 CYs, 384 DSPs, 192 RAMBs, 672 RFs, 4 PLLs. Compared to its predecessor, i.e., NG-Medium, it is ∼4× larger [3], e.g., NG-Medium has 34272 LUTs, 112 DSPs, 56 RAMBs. Correspondingly, NG-Large is ∼4× smaller compared to NG-Ultra.

### B. NXmap SW TOOL

*NXmap* is the design suite provided by NanoXplore to support all classical FPGA development stages except from simulation, which is currently performed with 3rd-party tools (ModelSim/QuestaSim). The tool supports both Verilog and VHDL hardware description languages, and includes functions for timing analysis and power estimation. It is divided in two components: (i) the graphical interface, which allows the user to compile existing projects, view the floorplan and inspect the implemented design, (ii) the Python wrapper, which allows the user to build projects by compiling Python scripts with the desired tool settings and functionalities.

The Python wrapper of NXmap supports the Python syntax and structures, and includes a plethora of NanoXplore routines/modules for each stage of the FPGA development flow. These Python routines can be categorized as follows:

- project-related: e.g., `createProject`, `addFiles`
- tool-related: e.g., `setOptions`

```python
# project creation
project = createProject(my_directory)
project.setVariantName('NG-LARGE')
project.setTopCellName('top_module')
project.addFiles(['design_file1.vhd', 'design_file2.vhd'])

# general tool settings
project.setOptions(['MappingEffort': 'Medium',
                    'RoutingEffort': 'High',
                    'DisableDSPRegisters': 'Yes',
                    'DefaultROMMapping': 'LUT',
                    'ManageUnconnectedOutputs': 'Ground'])

# custom mapping targets
project.addMappingDirective('getModels(.*mult.*)', 'MUL', 'DSP')
project.addMappingDirective('getModels(add_9u_9u)', 'ADD', 'CY')

# custom mapping locations
project.addDSPLocation('*mult_L267*', 'CGB[1x8]:L')
project.addRAMLocation('*RAM_INST0*', 'CGB[48x20]')

# clock constraint
project.createClock('getClockNet(clk)', 'clk', 40000)

# I/O signals and pads pairing
project.addPad('clk', {'location':'IO_B18D02P',
                       'standard':'LVCMOS', 'drive':'2mA'})
# project save
project.save('project.nym')

# synthesis
project.synthesize()
project.save('synthesis_netlist.vhd')

# place
project.place()
project.save('place_netlist.vhd')

# route
project.route()
project.save('route_netlist.vhd')

# reports
project.reportPorts()
project.reportInstances()

# static timing analysis
analyzer = project.createAnalyzer()
analyzer.launch()

# bitstream generation
project.generateBitstream('bitstream.nxb')
```

**CODE 1.** Example of python script for building a project in the NXmap tool.

- mapping-related: e.g., `createRegion`, `addRAM Location`
- stage-related: e.g., `synthesize`, `generate Bitstream`
- board-related: e.g., `addPads`, `addBanks`
- timing-related: e.g., `createClock`, `addFalse Path`

The NXmap routines take numerous arguments as input, providing a wide range of functionalities. NXmap also offers routines for monitoring the design flow and defining the verbosity of the reports. A Python code snippet demonstrating some basic NXmap functionalities is attached in Code 1.

## IV. PROPOSED ASSESSMENT METHODOLOGY

The proposed methodology for evaluating the NanoXplore SW tools and FPGAs is divided in 5 steps: (i) benchmark selection, (ii) definition of rating/evaluation method, (iii) *Synthesis* assessment, (iv) *Place & Route* (*P&R*) assessment, and (v) *Bitstream Generation* assessment. Currently, the methodology is executed manually by the developer/tester, however, some segments, such as the exploration of the tool settings, could be performed in an automatic fashion. Our methodology can also be adopted to test other FPGAs, or used to facilitate the development on new devices/tools.

### A. SELECTION OF BENCHMARKS

The first step of our approach is to create a pool of HDL benchmarks with varying complexity [19], i.e., simple circuits (e.g., arithmetic/memory units), designs of medium complexity (e.g., controllers), and high-performance benchmarks (e.g., image processing). Via the simple circuits, we aim to target specific tool options and FPGA primitives, while via the high-performance benchmarks, we stress the SW tool with demanding algorithms from real-world space applications. Moreover, our benchmarks impose varying constraints in I/O, memory and computations, thus, our evaluation is diverse and covers a wide range of requirements of on-board processing systems.

Regarding the high-performance benchmarks, our initial pool consists of various HDL IPs from the DSP and CV domains. To examine their suitability for our assessment methodology, we create multiple configurations for each benchmark by customizing its algorithmic parameters (e.g., image size, data bit-width, internal mask size), and perform an extensive design space exploration on the Electronic Design Automation (EDA) tools of vendors such as Intel/Altera, Xilinx, Microsemi and Synopsys. This exploration does not involve the NanoXplore tools and devices, namely, it is BRAVE-agnostic. We note that we use 3rd-party FPGAs that have similar features (resources, space-grade, technology node) with the examined BRAVE FPGA. Based on these results, as well as by considering other criteria, e.g., throughput/activity, parameterization/scalability, use of vendor's IP blocks, we select a final set of benchmarks that will be used for the evaluation of the BRAVE FPGA and SW tool.

### B. DEFINITION OF RATING/EVALUATION METHOD

The next step is to identify the evaluation metrics [19]. To define the metrics, we use two groups of FPGA engineers, i.e., ''black-box'' engineers that have not used the NanoXplore tools/devices and ''grey-box'' engineers that have only started using them. Indicatively, such metrics are the resource utilization, maximum clock frequency, power consumption, tool runtime & memory requirements, tool reports, tool options & attributes, floorplan capabilities, and GUI flexibility.

Next, we introduce a process to rate/evaluate the BRAVE devices/tools compared to 3rd-party solutions. The rating method [19] is illustrated in Figure 2. For each one of the measurable evaluation metrics, we calculate a reference value, which is the average value of all the results obtained from the 3rd-party tools. Then, we rate the NanoXplore SW tool by comparing its value with the reference value. Our rating process has 5 ranges, which are defined by thresholds: deficient (D), if NanoXplore is more than 20% worse, acceptable (A), if NanoXplore is 20%–5% worse, good (G), if NanoXplore is less than 5% worse, very good (V), if NanoXplore is 0.1%–5% better, excellent (E), if NanoXplore is more than 5% better. We note that this rating system is applied at each

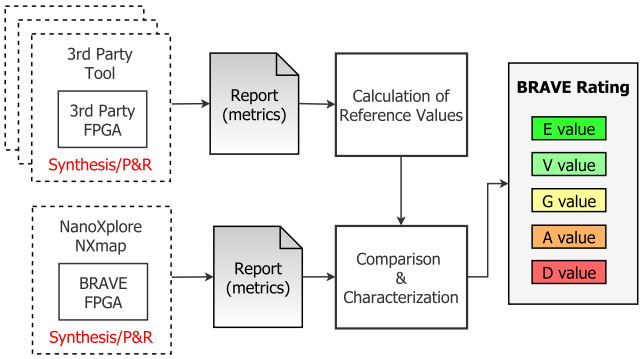



**FIGURE 2.** The proposed method for evaluating BRAVE tools & FPGAs [19].

one of the following methodology steps, which assess the typical stages of the FPGA design flow, i.e., Synthesis, P&R and Bitstream Generation.

### C. ASSESSMENT OF SYNTHESIS STAGE

The Synthesis assessment aims to: (i) explore and test the correct functionality of all the NXmap's settings and attributes, (ii) examine the quality of the results for different NXmap settings, (iii) evaluate the ability of the synthesizer to map efficiently the RTL designs on the FPGA primitives, (iv) evaluate the quality of the Synthesis reports, (v) rate the resource utilization via systematic comparisons to state-of-the-art 3rd-party tools.

The proposed methodology for realizing the aforementioned goals is illustrated in Figure 3. Initially, we adapt the algorithmic parameters of the selected benchmarks according to the features of BRAVE (resources, architecture of FPGA primitives). Next, we perform a preliminary Synthesis with the default NXmap settings to retrieve the ''default'' reports and detect potential issues. This step is also considered as test for the synthesizer's flexibility to automatically balance the resource utilization and provide a viable solution.

The preliminary Synthesis is followed by the phase of ''Program-Agnostic Tuning'', which explores the available settings that are related to Synthesis, and assesses their capability to drive the Synthesis process according to the user's choices and preferences. In this phase, it is not required for the user to be familiar with the HDL code of the benchmarks. Indicatively, we mention that the tool settings involve choices regarding the mapping effort of the synthesizer, the mapping target of the arithmetic/memory components, the DSP utilization ratio, the register duplication, and the style of the finite-state machine encodings. The evaluation is performed in both standalone and combinatorial fashion. Subsequently, we compare the NXmap results with the results of the 3rd-party tools based on our rating methodology. In case spikes are observed, a lower level exploration takes place by recursively decomposing the benchmark architecture to smaller building blocks and testing them individually. This is an essential modification of the initial ''QUEENS-FPGA'' methodology [19], which allows for in depth investigation of

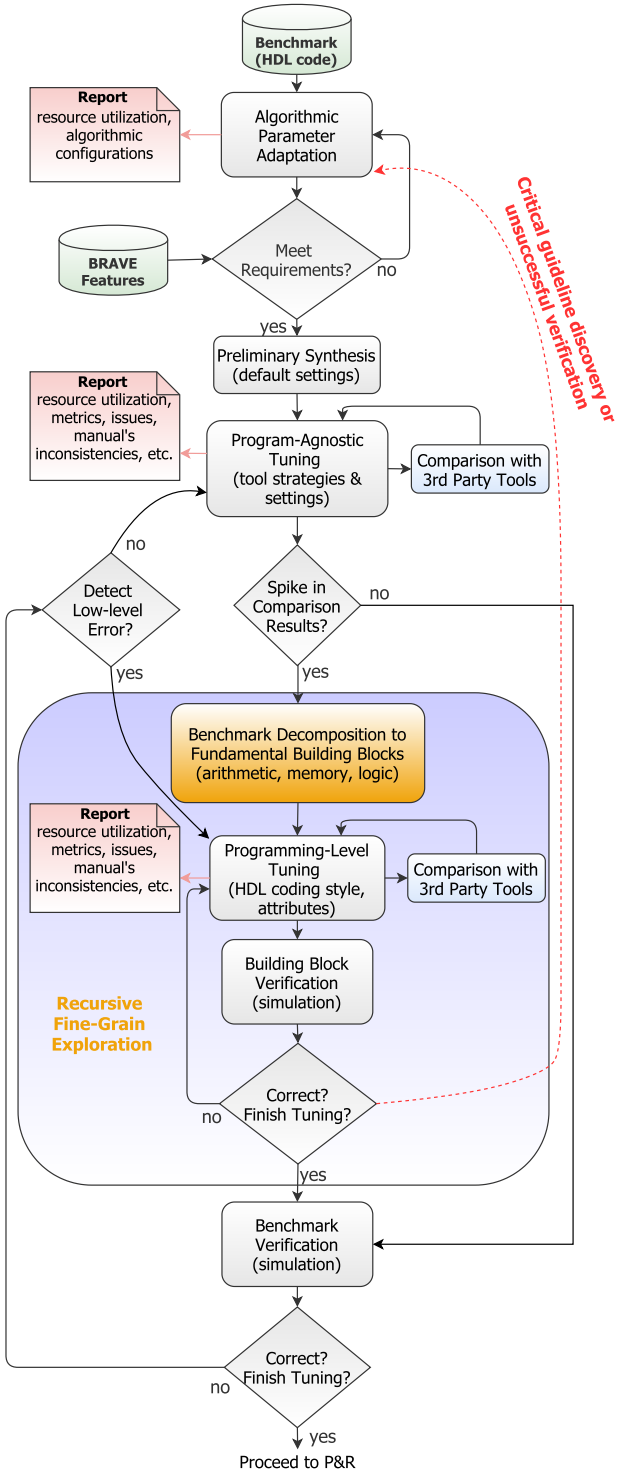



**FIGURE 3.** The proposed assessment methodology for Synthesis [26].

various optimization issues and/or errors, which, otherwise, would be very difficult to be detected at higher level.

The exploration with the building blocks is performed at the phase of "Programming-Level Tuning", which investigates the capability of the synthesizer to efficiently map the HDL code on the underlying BRAVE architecture. In this phase, we use standard template-based coding

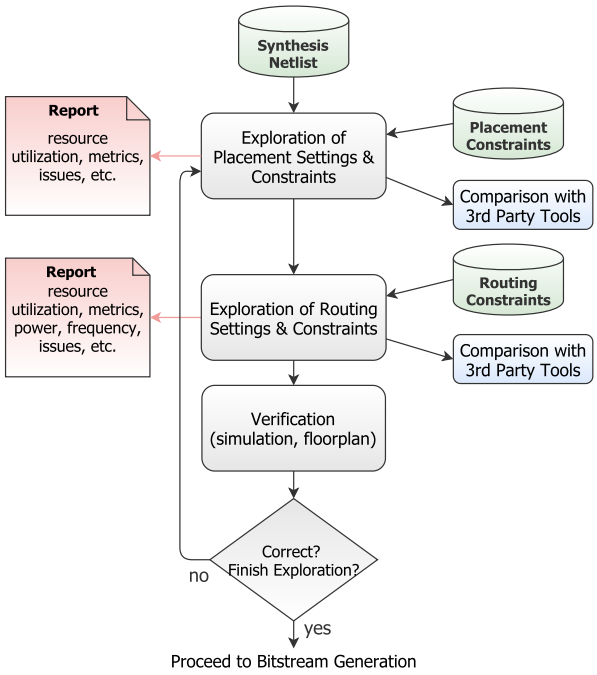



**FIGURE 4.** The proposed assessment methodology for Place & Route [19].

and attributes/directives to express memories, finite-state machines, and arithmetic components. If we identify a type of HDL coding that leads to improved results, or notice inability of the design to fit in the FPGA in spite of our efforts, we use our feedback loop (red dashed line) to re-customize the algorithmic parameters and proceed with a new benchmark configuration. Similarly to the previous phase, every exploration is followed by systematic 3rd-party comparison.

In both main exploration phases, we keep records of the examined metrics, i.e., resource utilization, provided features, efficiency of the existing Synthesis settings, proposed design guidelines, as well as the detected issues. Moreover, both phases include functional verification via simulations with 3rd-party simulators (ModelSim/QuestaSim). More specifically, the post-Synthesis netlists of the benchmarks and/or the basic building blocks are simulated, and the outputs are compared to the ground-truth data obtained from the behavioral RTL simulation.

### D. ASSESSMENT OF PLACE & ROUTE STAGE

A similar assessment methodology is designed for the P&R stage. This methodology, illustrated in Figure 4, takes as input the post-Synthesis netlist of either the entire benchmark or a small building block with problematic behavior. The testing of the building blocks as standalone components is another addition to our initial P&R methodology [19]. It becomes crucial by enabling the isolation of individual components, which may result in malfunction of the benchmark when tested on HW. For both cases, we evaluate the P&R-related metrics (e.g., resources, frequency, power) by exploring the various settings and physical constraints of the NXmap tool.

The main exploration/evaluation procedure consists of two phases.

Firstly, we evaluate all Placement options. In this phase, we assess the tool's capability to perform location specific placements via constraints regarding the targeted region and the placement of groups, either at fine-grain (LUTs, DFFs) or coarse-grain (Tiles, DSPs, RAMBs) level. Furthermore, we examine the quality of the reports and the efficiency of the more general Placement settings, e.g., the placing effort.

Secondly, we evaluate all the Routing-related settings. We examine if the tool is capable of delivering efficient solutions under stressing the implementation towards performance and/or increased routing congestion. Our exploration targets the timing constraints (e.g., timing driven, set false path, set max delay, create clock) and router's settings (e.g., router effort, router mode). We note that we combine the Routing settings with different Placement constraints from the previous phase. Moreover, in this phase we assess the Static Timing Analysis (STA) reports.

Similarly to our Synthesis assessment methodology, every experimentation is accompanied by: (i) systematic comparison with the 3rd-party tools, (ii) functional and timing verification via post-Place and post-Route netlist simulations, and (iii) floorplan inspection. Additionally, considering that Synthesis and P&R are tightly coupled (different Synthesis netlists may lead to different P&R results), we explore various scenarios by combining tool settings from both stages.

### E. ASSESSMENT OF BITSTREAM GENERATION STAGE

The final step of our methodology is to evaluate the Bitstream Generation stage and the FPGA configuration. More specifically, in this evaluation phase, we examine the correct bitstream generation for all the relevant tool options, the bitstream size, the programming speeds via the available configuration interfaces (JTAG, SpaceWire, EPROM), the correct functionality of the FPGA after multiple successive re-configurations, and the correctness of the actual HW execution on the BRAVE FPGAs. The latter is performed by establishing an I/O communication between the BRAVE FPGA and the host-PC (e.g., via UART, SpaceWire) and comparing the FPGA outputs with ground-truth data obtained from behavioral or post-P&R NXmap netlist simulations.

## V. EVALUATION

### A. DSP AND COMPUTER VISION BENCHMARKS

For the evaluation of NG-Large, according to our benchmark selection methodology, we created a set of 12 HDL benchmarks and selected 5 of them, which can be classified as follows: (i) *FIR Filter* for 1D signal processing, (ii) *Harris Corner Detector* and *Canny Edge Detector* for feature extraction (corners and edges, respectively), (iii) *GAD-Disparity* and *Spacesweep* for stereo matching (depth extraction in 3D scene reconstruction) [7]. Below, we present in brief one benchmark from each class:

**TABLE 2. Configuration of benchmark parameters.**

| Benchmark | Data | | | Mask | |
|---|---|---|---|---|---|
| | Input Size | Input Partition | I/O Bits | Size | Bits |
| FIR | $N\times1$ | continuous | 16/16 | $64\times1$ | $16\times16$ |
| Harris | $1024\times1024$ | $1024\times32$ | 8/32 | $7\times7$ | $8\times14$ |
| Canny | $1024\times1024$ | continuous | 8/4 | $3\times3$ | $8\times3$ |
| GAD-Disparity | $1024\times1024$ | $1024\times32$ | 8/10 | $7\times7$ | $8\times7$ |
| Spacesweep | $1024\times1024$ | $1024\times16$ | 8/32 | $13\times13$ | $8\times8$ |

1) Signal Filtering (*FIR Filter*): The component inputs samples in a streaming fashion (1 sample per clock cycle) and performs 1D signal filtering. More specifically, it is a deeply pipelined filter, which is implemented with a big sequence of registers and without memory resources, i.e., all the computations are performed in-place.
2) Feature Extraction (*Harris Corner Detector*): The component inputs a grayscale image and outputs a set of "corners", i.e., the coordinates $(x,y)$ of the most salient features depicted in the image. The image is divided in horizontal stripes, which are downloaded to the FPGA and processed successively by resource reusing. Functionally, in a loop over all pixels, Harris employs Gaussian-smoothed products of image derivatives to define an auto-correlation matrix, whose eigenvalues capture the principal intensity changes in the examined point's neighborhood. Corners are detected on pixels whose "cornerness" is sufficiently high and exceeds that of its neighboring pixels in a $3 \times 3$ region (via non-maximum suppression). All these operations are implemented via a succession of deep, fine-grained, pixel-based pipelines connecting memories that store intermediate results.
3) Depth Extraction (*GAD-Disparity*): The component inputs a pair of stereo images and outputs a two-dimensional disparity map, which can be transformed to depth map via simple calculations involving the focal length and baseline of the camera. The images are divided in horizontal stripes, which are downloaded successively to the FPGA and processed by resource reusing. In an iterative fashion, by implementing an outer loop over all examined disparities and an inner loop over all pixels, we match all $7 \times 7$ image blocks between images by minimizing a Gauss-aggregated sum of absolute differences. The on-chip memory is mostly used to store the pixels and their corresponding intermediate aggregated values (which are continuously compared/updated). On the other hand, the on-chip logic is mostly used to calculate the Gauss-aggregated values.

All the benchmarks are developed with parametric VHDL code. Therefore, at compile time, we can change various parameters, e.g., the input image size, the datapath bit-width, or certain parallelization factors. For the current evaluation, the benchmarks are configured as shown in Table 2. For example, Harris inputs a $1024 \times 1024.8$-bit image, partitioned

in 1024 × 32 pixel stripes (i.e., it processes 32 such bands), and performs convolutions with 7 × 7 14-bit kernels to output 32-bit corners.

### B. EXPERIMENTAL SETUP

The benchmarking results are produced by NXmap3 v5.0.4. Our evaluation is performed at two levels: (i) at tool level, by evaluating the NXmap settings and examining the resource utilization and the tool requirements, (ii) at HW level, by evaluating the chip's maximum frequency, the power consumption, the benchmarks' throughput and the bitstream configuration times. Overall, we report various experimental results for NG-Large, comparative results to 3rd-party FPGAs and NG-Medium, and system-level results from the implementation of an entire HW/SW algorithmic pipeline.

The functional verification of the benchmarks was performed via post-Synthesis and post-P&R simulations on realistic datasets. A natural signal sampled at 110 KSamples/s was used for FIR, 1024 × 1024 synthetic stereo images depicting a rover's view on Martian terrain were employed for GAD-Disparity and Spacesweep, 1024 × 1024 images depicting rocky Martian terrains were used for Harris and Canny. The derived results were compared with the outputs of the behavioral RTL simulation and revealed a fully functional error-free operation. The benchmarks were also successfully ported and executed on the NG-Large HW, utilizing a UART communication for transmitting/receiving the I/O data.

### C. EVALUATION OF NXmap SW TOOL

Regarding the Synthesis process, Table 3 presents the resource utilization when using the default settings/options of the NXmap tool. With this setup, zero DSPs are employed for FIR and all the arithmetic operations are mapped onto CY units (61% utilization). Similarly, for Harris, the majority of multiplications are mapped onto CYs instead of DSPs. To balance the utilization, we used our Synthesis exploration procedure (see Figure 3) to customize the tool options according to the requirements of each benchmark. More specifically, we shared the arithmetic operations between DSPs and CYs using the corresponding NXmap Python routines. The results of our customization decisions are reported in Table 4. Employing DSPs in FIR decreased the CY utilization from 61% to 4% and increased the DSPs by 17%. For Harris, we get a balanced use between CY and DSP blocks, i.e., from 43% and 8% to 23% and 22%, respectively. For GAD-Disparity, the default tool settings deliver the same resources with our custom settings that balance the DSP and CY utilization. For Spacesweep, the tool employs by default RF resources for all the small memory components, thus, it is capable of making decisions with respect to the memory size. To test the custom mapping of memories, we forced the tool to map the small memories onto RAMBs.

Before proceeding to the P&R evaluation, we also examined how our mapping choices affect the timing performance. The Harris and Canny benchmarks achieve better clock frequency when the multiplication operations are mapped onto DSPs. On the other hand, FIR, GAD-Disparity and Spacesweep achieve better timing when the default mapping is used. Specifically, the custom mapping in FIR decreased the frequency by 44%, while in GAD-Disparity and Spacesweep, the decrease was only 1MHz and 0.65MHz, respectively. Therefore, during the P&R evaluation, we decided to remove the custom mapping in these benchmarks, and tune only the P&R-related options.

Subsequently, we used our P&R exploration procedure (see Figure 4) to experiment with the tool options and achieve the best possible timing performance. Table 5 reports the P&R resource utilization for the implementations with the default P&R option values, while Table 6 reports the results for the customized P&R option values. The memory and arithmetic resources, i.e., RFs/RAMBs and CYs/DSPs, remain equal to those generated by Synthesis. Moreover, the variations in resources for different P&R options

**TABLE 3.** Synthesis resources of NG-Large with default NXmap settings.

| Benchmark | LUT | DFF | CY | DSP | RF | RAMB |
|---|---|---|---|---|---|---|
| FIR | 0 | 7136 | 19440 | 0 | 0 | 0 |
| | (0%) | (6%) | (61%) | (0%) | (0%) | (0%) |
| Harris | 6210 | 16398 | 13794 | 27 | 0 | 69 |
| | (5%) | (13%) | (43%) | (8%) | (0%) | (36%) |
| Canny | 1845 | 2348 | 1167 | 2 | 0 | 177 |
| | (2%) | (2%) | (4%) | (1%) | (0%) | (93%) |
| GAD-Disparity | 1000 | 3628 | 4548 | 4 | 0 | 85 |
| | (1%) | (3%) | (15%) | (2%) | (0%) | (45%) |
| Spacesweep | 5500 | 10222 | 6277 | 50 | 8 | 74 |
| | (5%) | (8%) | (20%) | (14%) | (2%) | (39%) |
| **FPGA Resources** | 137K | 129K | 32K | 384 | 672 | 192 |

**TABLE 4.** Synthesis resources of NG-Large with tailored NXmap settings.

| Benchmark | LUT | DFF | CY | DSP | RF | RAMB |
|---|---|---|---|---|---|---|
| FIR | 2 | 7170 | 1008 | 64 | 0 | 0 |
| | (1%) | (6%) | (4%) | (17%) | (0%) | (0%) |
| Harris | 6110 | 15304 | 7112 | 81 | 0 | 69 |
| | (5%) | (12%) | (23%) | (22%) | (0%) | (36%) |
| Canny | 1845 | 2299 | 1086 | 4 | 0 | 177 |
| | (2%) | (2%) | (4%) | (2%) | (0%) | (93%) |
| GAD-Disparity | 1000 | 3628 | 4548 | 4 | 0 | 85 |
| | (1%) | (3%) | (15%) | (2%) | (0%) | (45%) |
| Spacesweep | 5499 | 10222 | 6277 | 50 | 0 | 79 |
| | (5%) | (8%) | (20%) | (14%) | (0%) | (42%) |
| **FPGA Resources** | 137K | 129K | 32K | 384 | 672 | 192 |

**TABLE 5.** P&R resources of NG-Large with default NXmap settings.

| Benchmark | LUT | DFF | CY | DSP | RAMB | MHz |
|---|---|---|---|---|---|---|
| FIR | 0 | 7136 | 19440 | 0 | 0 | 214 |
| | (0%) | (6%) | (61%) | (0%) | (0%) | |
| Harris | 6205 | 16516 | 13794 | 27 | 69 | 31 |
| | (5%) | (13%) | (43%) | (8%) | (36%) | |
| Canny | 1844 | 2412 | 1167 | 2 | 177 | 35 |
| | (2%) | (2%) | (4%) | (1%) | (93%) | |
| GAD-Disparity | 994 | 3664 | 4548 | 4 | 85 | 47 |
| | (1%) | (3%) | (15%) | (2%) | (45%) | |
| Spacesweep | 5493 | 10280 | 6277 | 50 | 74 | 51 |
| | (5%) | (8%) | (20%) | (14%) | (39%) | |
| **FPGA Resources** | 137K | 129K | 32K | 384 | 192 | – |

**TABLE 6.** P&R resources of NG-Large with final tailored NXmap settings.

| Benchmark | LUT | DFF | CY | DSP | RAMB | MHz |
|---|---|---|---|---|---|---|
| FIR | 0 (0%) | 7136 (6%) | 19440 (61%) | 0 (0%) | 0 (0%) | 214 |
| Harris | 6105 (5%) | 15413 (13%) | 7112 (23%) | 81 (22%) | 69 (36%) | 40 |
| Canny | 1843 (2%) | 2349 (2%) | 1086 (4%) | 4 (2%) | 177 (93%) | 38 |
| GAD-Disparity | 998 (1%) | 3672 (3%) | 4548 (15%) | 4 (2%) | 85 (45%) | 50 |
| Spacesweep | 5458 (5%) | 10276 (8%) | 6277 (20%) | 50 (14%) | 79 (42%) | 52 |
| **FPGA Resources** | 137K | 129K | 32K | 384 | 192 | – |

(*DensityEffort*, *CongestionEffort*, *PolishingEffort*, *RoutingEffort*, *BypassingEffort*) are negligible for all the benchmarks, i.e., ±10 LUTs. Ultimately, we observed that, with exception of FIR, all the other benchmarks can achieve better timing performance by changing some of the default P&R option values. We combined different values for the aforementioned tool options, and report those that provide the maximum frequency according to NXmap's STA. For Harris, the *PolishingEffort* option was set to "low" rather than "medium", giving an increase of 9.5MHz. For GAD-Disparity, the *PolishingEffort* option was set to "low" rather than "medium" and the *DensityEffort* to "medium" rather than "low", delivering an increase of 2.6MHz. For Canny, the *PolishingEffort* option was set to "high" rather than "medium", providing an increase of 2.7MHz. For Spacesweep, the *CongestionEffort* option was set to "medium" rather than "high", delivering an increase of 0.7MHz. We also notice that for FIR, the tool achieves almost the double maximum frequency, i.e., 214MHZ from 121MHz, when we do not employ a custom mapping for the arithmetic operations.

Overall, with respect to the reported resource utilization, we conclude that NXmap maps our designs as expected. For FIR, the tool correctly omits the RAMBs, because it is a deeply pipelined filter with a big sequence of registers. Moreover, with our custom mapping, it correctly occupies 64 DSPs, coinciding with the 64 filter's taps/coefficients. Regarding Harris, Canny, GAD-Disparity and Spacesweep, several of the available RAMB configurations are employed, e.g., 24K×2, 12K×4, and thus, reasonable RAMB utilization is derived for 1024-pixel-wide images (36%, 93%, 45% and 42%, respectively). We also note that NXmap successfully recognizes and reports all the arithmetic & memory components and finite-state machines of our benchmarks.

Next, we examine the system requirements of NXmap for compiling our benchmarks. As shown by the overall runtime and memory usage in Table 7, NXmap is a lightweight tool. Apart from FIR, all the other benchmarks are very demanding, but even for those, both runtime and peak memory usage remain in relatively low levels, manageable even by low-end CPUs. We also notice that even the total real-world

**TABLE 7.** NXmap system[1] requirements for the benchmark implementations on NG-Large.

| Benchmark | Synthesis Runtime | P&R Runtime | Total Runtime | Peak Memory |
|---|---|---|---|---|
| FIR | 8s | 55s | 118s | 1099KB |
| Harris | 141s | 270s | 577s | 1576KB |
| Canny | 1334s | 104s | 1521s | 1225KB |
| GAD-Disparity | 47s | 117s | 287s | 1291KB |
| Spacesweep | 95s | 158s | 382s | 1445KB |

[1] System: Intel Xeon E5-2650 @2.60GHz ×16, 64GB RAM.

elapsed time for all the processes, i.e., up to the bitstream generation, is just a couple of minutes (<6). Canny is the only benchmark that required several minutes (~25), and the reason is the high RAMB utilization, which is at 93%.

### D. COMPARISON TO 3rd-PARTY TOOLS

Following our NXmap testing, we compare NG-Large with 3rd-party FPGAs Depending on the benchmark, NXmap provides comparable P&R resource utilization for certain primitives and even improved in other cases.

For the Harris benchmark, NXmap provides a good LUT utilization, i.e., 3.2× less LUT versus the reference value. When considering the pass-thru LUTs from the CY utilization, the total number of LUTs increases. The LUT utilization should be examined along with the number of employed DSPs, where NXmap utilizes 1.5× less. Regarding the RAM resources, NXmap delivers less RAMBs (it has larger RAMB size) and the total RAMB Kbits are less than the reference value. Specifically, Harris utilizes 56% less RAMBs compared to the reference value. We note that the maximum clock frequency is less than the reference value, however, there is a small increase compared to the frequency of the previous tool versions. For Canny, NXmap provides acceptable LUT utilization with an increase of 6%, but if we also consider the route-thru LUTs from the CYs, it is increased by 48%. The DFF resources are increased by almost 50%, while the utilization of RAMBs and DSPs is very good, as NXmap provides the same number of resources with the reference values. Overall, we consider as acceptable the resources of Canny when taking into account that it is the only benchmark almost filling up all RAMBs.

For the GAD-Disparity benchmark, NXmap provides promising LUT utilization, as it employs a small number, even when considering the CY resources. Regarding RAM resources, NXmap is below the reference value in both blocks and Kbits. In terms of frequency, NXmap provides less MHz, however, this value was again increased by almost 15% versus the previous tool version, which means that the tool is improving. For Spacesweep, NXmap also provides a very good LUT utilization. It achieves better results by 52% and drops to comparable results when considering the CYs, with just a 3% LUT increase. The DFF utilization is also better by 5%. Finally, the DSP and RAMB utilization is excellent, as NXmap outperforms the average values of the other tools by 20% and 30%, respectively.

**TABLE 8.** Performance of NG-Large with tailored NXmap settings.

| Benchmark | Frequency | Runtime | Throughput[1] |
|---|---|---|---|
| FIR | 214MHz | continuous | 214 MSPS |
| Harris | 40MHz | 0.19s / frame | 5.3 FPS |
| Canny | 38MHz | 0.10s / frame | 10 FPS |
| GAD-Disparity | 50MHz | 6.7s / frame | 18 MPDS |
| Spacesweep | 52MHz | 10.8s / frame | 29 MPDS |

[1] Throughput excludes I/O and differs per benchmark: MSPS = mega samples per second, FPS = frames per second, MPDS = mega pixel disparities per second.

### E. EVALUATION OF NG-LARGE HW

Table 8 presents the maximum frequency of the benchmarks. Accordingly, a throughput metric is presented for each benchmark to highlight the potential of NG-Large for real-time operation. Overall, we note that NG-Large provide sufficient resources and frequency to all benchmarks. The achieved throughput of FIR facilitates a multitude of applications (e.g., in Telecom) and its resource utilization allows for complementary HDL components to be placed in the chip. The time required for a complete reconstruction using GAD-Disparity and Spacesweep could improve the conventional depth extraction of Mars rovers by 1 order of magnitude (in terms of resolution and speed). We note that, in the tested configuration, Spacesweep examines 3× more depth levels than GAD-Disparity (300 versus 100), and thus, provides much higher accuracy. Furthermore, given that most VBN applications require 1–10 FPS, we conclude that the throughput of Harris and Canny, i.e., 5.3 and 10 FPS, respectively, leaves enough room for the complementary components of an algorithmic chain to finish on time.

Next, we evaluate the power consumption of NG-Large by comparing it to one of the most prominent space-grade FPGAs (labeled as "3rd-party device"), which has similar technology node and resources. The static power consumption has been measured when the FPGAs are powered up and no bitstream is loaded, using the physical components and chipscope tools. The results are similar for both devices, i.e., 1.99W for NG-Large and 1.91W for the 3rd-party device. For the dynamic power consumption, which mainly depends on the number of utilized resources, the clock frequency and the toggle rates, we have performed various experiments in the same environment conditions using the provided power analyzer tools. The derived results, which are discussed in the next paragraph and illustrated in Figure 5, show that NG-Large provides comparable dynamic power consumption, and in some cases, even better.

Figure 5a shows the dynamic power consumption of the Logic Elements (LEs). We note that each LE of NG-Large consists of 1 LUT and 1 DFF. NG-Large delivers 5× higher power consumption than the 3rd-party device. However, this difference decreases for bigger LE utilization, and specifically, it is reduced up to 1.1× when almost all the LEs are utilized. Figure 5b reports the power consumption for different DSP utilization. In this case, NG-Large consumes 2.6× higher power than the 3rd-party device. We note that the DSPs of both FPGAs have similar architecture and

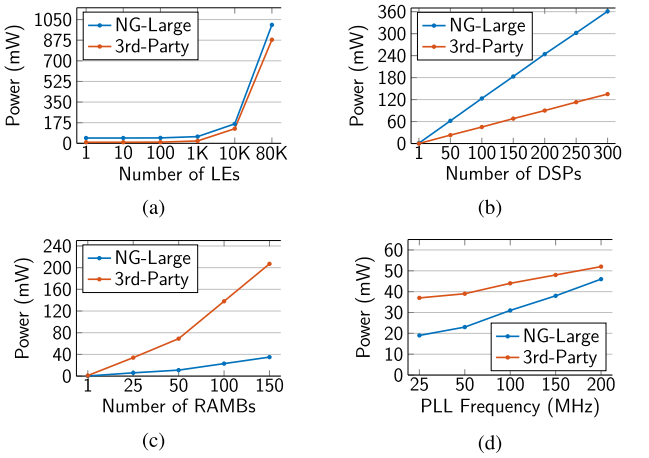



**FIGURE 5.** Power consumption of NG-Large and 3rd-party FPGA w.r.t. the (a) LE, (b) DSP, (c) RAMB utilization, and (d) clock frequency generated by PLL.

**TABLE 9.** Bitstream configuration results on NG-Large.

| Benchmark | Bitstream Size | JTAG Configuration Time[1] |
|---|---|---|
| FIR | 960KB | 2.5s |
| Harris | 1751KB | 4.5s |
| Canny | 2669KB | 5.9s |
| GAD-Disparity | 1563KB | 4.1s |
| Spacesweep | 1719KB | 4.3s |

[1] System: Intel Core i7-4500U @1.80GHz ×4, 8GB RAM.

data bit-width. Figure 5c illustrates the scaling of the power consumption with respect to the RAMB utilization. It is important to notice that NG-Large provides memory blocks of 48Kb, while the 3rd-party device provides smaller blocks. Despite that, NG-Large delivers 6× lower power compared to the 3rd-party device. For all the aforementioned experiments, we have used a clock frequency of 25MHz. The last step was to examine the power consumption of the PLL, when assigned to generate different clock frequencies (the input frequency is 25MHz). In Figure 5d, we observe the lower power values of NG-Large compared to the 3rd-party device. Nevertheless, the 3rd-party FPGA shows smaller increases (0.08mW/MHz) compared to NG-Large (0.16mW/MHz). This implies that NG-Large provides high power efficiency for low frequencies, which deteriorates for higher frequencies, but is still better than 3rd-party.

Finally, in Table 9 we report the bitstream size of each benchmark, and the time required for the NG-Large configuration/programming via the JTAG interface. According to the results, the configuration time of NG-Large is almost proportional to the bitstream size: 384KB per second are handled for FIR, 389KB for Harris, 452KB for Canny, 381KB for GAD-Disparity and 399KB for Spacesweep. We also observe that the bitstream of Canny is around 1KB larger than that of the other benchmarks, which is due to its 93% RAMB utilization, and thus, it requires more time (2-3 extra seconds) to be configured. Furthermore, we report that the size of the BRAVE bitstream is not fixed, like in other 3rd-party FPGAs, but depends on the design's size and complexity.

**TABLE 10. Resource utilization of NG-Large and NG-Medium FPGAs[1].**

| Bench.[2] | FPGA | LUT | DFF | CY | DSP | RAMB | MHz | NG-M Notes[3] |
|---|---|---|---|---|---|---|---|---|
| FIR | NG-L | 0% | 6% | 61% | 0% | 0% | 214 | different mapping |
| | NG-M | 1% | 23% | 13% | 58% | 0% | 136 | (mult. to DSPs) |
| Harris | NG-L | 5% | 13% | 23% | 22% | 36% | 40 | 4× smal. input part. |
| | NG-M | 19% | 38% | 73% | 100% | 63% | 11 | (1K×8 vs 1K×32) |
| Canny | NG-L | 2% | 2% | 4% | 1% | 93% | 38 | 4× smal. input img. |
| | NG-M | 7% | 8% | 13% | 5% | 90% | 50 | (0.25MP vs 1MP) |
| GAD-D. | NG-L | 1% | 3% | 15% | 2% | 45% | 50 | 2× smal. input part. |
| | NG-M | 4% | 10% | 18% | 41% | 86% | 52 | (1K×16 vs 1K×32) |
| Spacesw. | NG-L | 5% | 8% | 20% | 14% | 42% | 52 | 4× smal. input img. |
| | NG-M | 22% | 32% | 78% | 45% | 95% | 30 | (0.25MP vs 1MP) |

[1] NG-Large is 4× bigger in resources than NG-Medium.
[2] The same exact algorithms have been implemented on both FPGAs.
[3] Tool/Benchmark modifications to fit the design in NG-Medium.

### F. COMPARISON TO NG-MEDIUM

In this section, we compare NG-Large against its predecessor, i.e., NG-Medium, to evaluate the progress of BRAVE devices and examine if NG-Large provides significant advantage due to being 4× bigger in terms of resources/area. We implement the same benchmarks on NG-Medium, initially configured as shown in Table 1, and apply the final tailored tool settings that provided the best frequency on NG-Large. We note that we do not change the algorithm of the benchmarks, namely, the exact same HDL sources are implemented in both devices. Only in case of resource over-utilization or unexpected tool behavior in NG-Medium, we modify either the benchmark parameters (e.g., use smaller input image) or the tool settings (e.g., apply different mapping).

Table 10 reports the resource utilization of both devices. For the FIR implementation on NG-Medium, we share the arithmetic operations between CYs and DSPs, otherwise the tool over-utilizes the resources. This choice also leads to a clock frequency of 135MHz, which is 1.6× smaller than NG-Large. The memory resources of NG-Medium forced us to modify the parameters of the rest benchmarks. More specifically, in Harris and GAD-Disparity, we decreased the height of the input image band by 4× and 2×, respectively, while in Canny and Spacesweep, we decreased the size of the entire image by 4× (from 1024 × 1024 to 512 × 512). The derived results show that, even though the designs of NG-Large concern bigger image/band sizes, its resource utilization percentage is significantly better, leaving room for implementing complementary components, increasing the parallelization, or serving even bigger input images.

In Figure 6, we show how the benchmark runtime is improved in NG-Large compared to NG-Medium. For FIR, the performance improvement is almost 2×, which is due to the better clock frequency. Nevertheless, we note that we can improve the performance in NG-Large even more by exploiting its resources and increasing the parallelization, e.g., implement parallel MACs, adder trees. Harris in NG-Large is also better by 4.6×, as the clock frequency is increased by ∼4×, and NG-Medium has to process 4× more (smaller though) image bands. Canny achieves better clock frequency and runtime in NG-Medium, but for the 1/4 of NG-Large's input image. To process an 1024 × 1024 image, it will have to process each 1/4 of the image, send the partitioned edge map back to the CPU, and at the end, bring all the partitioned edge maps back to the FPGA. These extra steps will add an overhead of ∼60ms in the good scenario, i.e., when using the SpaceWire interface at 100Mbps. Thus, the total performance improvement in NG-Large is 1.4×. Regarding GAD-Disparity, both devices achieve almost the same clock frequency, i.e., around 50MHz, however, NG-Medium has to process 2× more (smaller though) image bands. This differentiation in the partition of the input image is translated to 1.2× performance improvement in NG-Large. For Spacesweep, considering that NG-Medium has to process smaller input image to avoid resource over-utilization, and also with a clock frequency decreased by 1.7×, NG-Large delivers around 1.7× better performance.

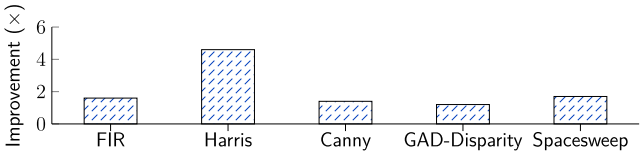



**FIGURE 6. Runtime improvement in NG-Large compared to NG-Medium for the same exact algorithms.**

Concluding, NG-Large provides increased flexibility due to offering more resources than NG-Medium, which is stressed (or is unable) to implement computer vision algorithms for typical image sizes such as 1 megapixel. In contrast, NG-Large delivers better performance, which can be further improved by exploiting its resources to increase the parallelization. Furthermore, NG-Large leaves a significant amount of resources, which can be exploited for implementing other complementary HDL components in the case of algorithmic pipelines, such as that presented in Section VI.

## VI. SYSTEM-LEVEL TEST: COMPLETE VBN PIPELINE

In this section, we evaluate NG-Large for implementing an entire HW/SW VBN system for Rover localization tasks. More specifically, we employ the "SPARTAN2" algorithmic pipeline, which is a vision-based autonomous navigation system from past ESA activities [6], [7], [43], and customize it for the BRAVE technology. The HW/SW co-processing architecture of "SPARTAN2" is illustrated in Figure 7. It includes SpaceWire communication interfaces to connect the FPGA accelerator (NG-Large) with an OBC (LEON-based GR740 of Cobham Gaisler), as well as a telemetry & telecommand CODEC IP, which is based on CCSDS Space Packet standard and PUS services. Regarding the image processing tasks, the Harris Corner Detector and SIFT Descriptor & Matching algorithms are implemented in VHDL, along with an arbiter, which controls their execution and handles the I/O. The floorplan of NG-Large with the implementation of the "SPARTAN2" algorithms (Harris and SIFT) is illustrated in Figure 8. For comparison purposes, we implement the same system on a prominent 3rd-party space-grade FPGA with similar resources. We also note that the input data are pairs of 512 × 512 stereo images

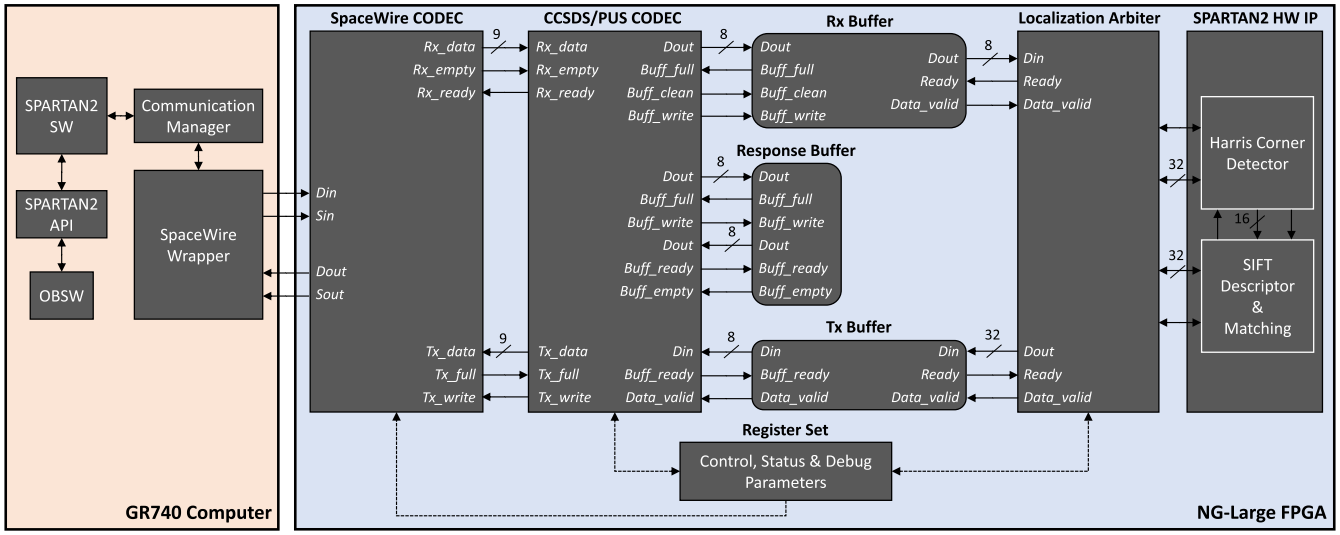



**FIGURE 7.** The block diagram of the "SPARTAN2" HW/SW architecture.

Regarding the implementation on NG-Large, the entire design could not be synthesized without directives, due to memory over-utilization and the tool's inability to complete the routing of the resources. Therefore, based on our methodology, we explored various Synthesis options, and ended up using the `addMappingDirective` routine, which defines custom mapping targets, with the following arguments:

- *'getModels(.*mult.*)', 'MUL', 'DSP'*, to force the tool to map every multiplier onto DSPs.
- *'getModels(add_31u_31u to add_36u_36u', 'ADD', 'DSP')*, to map all the adders with 31–36 bit-width onto DSPs and reduce the CY utilization under 80%, facilitating the tool to complete the routing of all the resources.
- *'getModels(FIFOs and small memories)', 'RAM', 'RF'*, to map small memories and FIFOs onto RF blocks and reduce the RAMB utilization, facilitating the tool to fit the entire design in NG-Large.

Table 11 reports the resource utilization and the maximum clock frequency of the two FPGAs. Due to a different LE and LUT architecture (the 3rd-party device integrates 4× more LUTs in a LE and has 6-input LUTs), NG-Large results in 6.5× and 2× higher LE and LUT utilization, respectively. Nevertheless, when considering the total number of available LEs and LUTs in both devices, we observe that the utilization percentage of LEs and LUTs is only 3% and 9% higher, respectively. Similarly, NG-Large utilizes 1.4× more DFFs, however, its utilization percentage is better by 4%. Regarding the DSPs, NG-Large delivers 2× higher utilization, but this is due to mapping arithmetic operations onto DSPs to save logic resources. In contrast, NG-Large utilizes half of the RAMBs employed by the 3rd-party device. This is explained by the bigger storage capacity of each NG-Large's RAMB. Finally, in terms of performance, both devices achieve comparable clock frequency, as there is only a small difference of 8MHz.

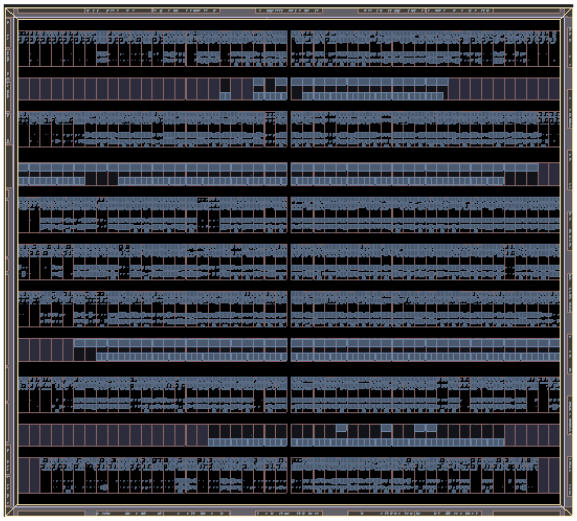


**FIGURE 8.** The floorplan view of "SPARTAN2" algorithms on NG-Large.

**TABLE 11.** Resource utilization of "SPARTAN2" HW system.

| FPGA | LE | LUT | DFF | DSP | RAMB | MHz |
|---|---|---|---|---|---|---|
| NG-Large | 98279 (76%) | 98279 (71%) | 56296 (44%) | 250 (65%) | 113 (58%) | 22 |
| 3rd-Party | 14894 (73%) | 50427 (62%) | 39008 (48%) | 129 (40%) | 228 (77%) | 30 |

Table 12 summarizes the performance results for the two implementations. Based on the maximum frequency reported by the static timing analysis of the tools, we configure the system clock of the FPGAs to the closest frequency that facilitates a seamless integration of the entire system to each device. Hence, the clock frequency is configured at 12.5MHz and 25MHz for NG-Large and the 3rd-party device, respectively. The first two table columns report the execution times of the Harris and SIFT algorithms for one of the pair's stereo images. As expected, due to having 2× faster clock, the 3rd-party device demonstrates around 2× less execution

**TABLE 12. Performance of "SPARTAN2" HW system.**

| FPGA | Harris | SIFT | SpW I/O | Total System | |
|---|---|---|---|---|---|
| | Time[1] | Time[1] | Time[1] | Time[2] | Throughput[2] |
| NG-Large | 208ms | 395ms | 28ms | 1251ms | 0.8 FPS |
| 3rd-Party | 104ms | 196ms | 28ms | 624ms | 1.6 FPS |

[1] refers to one 512×512 image.
[2] refers to a localization step with one 512×512 stereo pair.

time. We note that the execution time varies among different input images, as it is affected by the number of features detected. Regarding the I/O time via SpaceWire configured at 100Mbps, it is around 28ms for both implementations, as the transmission of each one of the 16 image bands requires 1.75ms. The last two table columns report the performance of the entire system (I/O + processing) for a pair of stereo images. Again, as expected, the 3rd-party device provides 2× throughput. In any case, the new NG-Large FPGA provides comparable performance, which is improved as the SW tools become more mature.

## VII. CONCLUSION & FUTURE WORK

In this work, we customized a methodology for evaluating the new European BRAVE FPGAs and their associated EDA tools. Our proposed methodology was applied on the NG-Large device. The experimental evaluation was based on HDL benchmarks developed in past ESA activities, it generated a variety of results regarding the SW tool and FPGA chip performances, while it also included comparisons to 3rd-party vendors. Indicatively, besides measuring improved tool flexibility and efficiency, we showed that image processing algorithms could achieve feature extraction with a throughput of around 10 FPS and depth extraction with a latency of around 10s, while more classical DSP functions could process more than 200 MSPS. Overall, NG-Large could implement high-performance and complicated algorithmic pipelines with sufficient HW metrics, i.e., resource utilization, throughput, and power consumption, which were all shown to be competitive with 3rd-party designs. Therefore, the BRAVE FPGAs constitute today an additional solution for space system engineers. Our future work includes: (i) benchmarking with HDL IPs from more application domains, e.g., Telecommunications and Artificial Intelligence, (ii) testing of new versions of SW tools, and (iii) evaluation of NG-Ultra, i.e., the next FPGA of the BRAVE series, which will also include the SoC's embedded processor.

## ACKNOWLEDGMENT

The authors would like to thank the NanoXplore Team for their support during the technical work.

## REFERENCES

[1] G. Lentaris, K. Maragos, I. Stratakos, L. Papadopoulos, O. Papanikolaou, D. Soudris, M. Lourakis, X. Zabulis, D. Gonzalez-Arjona, and G. Furano, "High-performance embedded computing in space: Evaluation of platforms for vision-based navigation," *J. Aerosp. Inf. Syst.*, vol. 15, no. 4, pp. 178–192, Apr. 2018.
[2] NanoXplore. *RHBD FPGAs, SoCs, eFPGAs*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.nanoxplore.com/
[3] J. L. Mauff and E. Lepape, "From eFPGA cores to RHBD system-on-chip FPGA," in *Proc. Space FPGA Users Workshop (SEFUW)*, 2018, pp. 1–53.
[4] J. L. Mauff, "NX RHBD FPGA solutions for OBDP applications," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2019, pp. 1–34.
[5] A. Perez, A. Rodriguez, A. Otero, D. G. Arjona, A. Jimenez-Peralo, M. A. Verdugo, and E. De La Torre, "Run-time reconfigurable MPSoC-based on-board processor for vision-based space navigation," *IEEE Access*, vol. 8, pp. 59891–59905, 2020.
[6] G. Lentaris, I. Stamoulias, D. Soudris, and M. Lourakis, "HW/SW codesign and FPGA acceleration of visual odometry algorithms for rover navigation on Mars," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 8, pp. 1563–1577, Aug. 2016.
[7] G. Lentaris, K. Maragos, D. Soudris, X. Zabulis, and M. Lourakis, "Single- and multi-FPGA acceleration of dense stereo vision for planetary rovers," *ACM Trans. Embedded Comput. Syst.*, vol. 18, no. 2, pp. 1–27, Apr. 2019.
[8] G. Lentaris, I. Stratakos, I. Stamoulias, D. Soudris, M. Lourakis, and X. Zabulis, "High-performance vision-based navigation on SoC FPGA for spacecraft proximity operations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 4, pp. 1188–1202, Apr. 2020.
[9] V. Leon, G. Lentaris, E. Petrongonas, D. Soudris, G. Furano, A. Tavoularis, and D. Moloney, "Improving performance-power-programmability in space avionics with edge devices: VBN on Myriad2 SoC," *ACM Trans. Embedded Comput. Syst.*, vol. 20, no. 3, pp. 1–23, Apr. 2021.
[10] J. E. Navarro, A. Samuelsson, H. Gingsjö, J. Barendt, A. Dunne, *L. Buckley*, D. Reisis, A. Kyriakos, E.-A. Papatheofanous, C. Bezaitis, P. Matthijs, J. P. Ramos, and D. Steenari, "High-performance compute board—A fault-tolerant module for on-board vision processing," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2021, pp. 1–7.
[11] C. Adams, A. Spain, J. Parker, M. Hevert, J. Roach, and D. Cotten, "Towards an integrated GPU accelerated SoC as a flight computer for small satellites," in *Proc. IEEE Aerosp. Conf.*, Mar. 2019, pp. 1–7.
[12] F. C. Bruhn, N. Tsog, F. Kunkel, O. Flordal, and I. Troxel, "Enabling radiation tolerant heterogeneous GPU-based onboard data processing in space," *CEAS Space J.*, vol. 12, no. 4, pp. 551–564, Dec. 2020.
[13] S. Parkes, C. McClements, D. McLaren, B. Youssef, M. S. Ali, A. F. Florit, and A. G. Villafranca, "SpaceWire and SpaceFibre on the microsemi RTG4 FPGA," in *Proc. IEEE Aerosp. Conf.*, Mar. 2016, pp. 1–8.
[14] Y. Barrios, A. J. Sanchez, L. Santos, and R. Sarmiento, "SHyLoC 2.0: A versatile hardware solution for on-board data and hyperspectral image compression on future space missions," *IEEE Access*, vol. 8, pp. 54269–54287, 2020.
[15] A. Tsigkanos, N. Kranitis, D. Theodoropoulos, and A. Paschalis, "High-performance COTS FPGA SoC for parallel hyperspectral image compression with CCSDS-123.0-B-1," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 11, pp. 2397–2409, Nov. 2020.
[16] Xilinx. *Space-Grade Portofolio*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.xilinx.com/products/silicon-devices/fpga/rt-kintex-ultrascale.html#productTable
[17] Microsemi. *Rad-Tolerant FPGAs*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.microsemi.com/product-directory/fpga-soc/1640-rad-tolerant-fpgas
[18] Atmel. *Atmel Aerospace: Reprogrammable FPGA With Built-in SEU and SET Protections*. Accessed: Aug. 24, 2021. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/4066I-FPGA-Reprogramable_US_051115_web.pdf
[19] K. Maragos, V. Leon, G. Lentaris, D. Soudris, D. Gonzalez-Arjona, R. Domingo, A. Pastor, D. M. Codinachs, and I. Conway, "Evaluation methodology and reconfiguration tests on the new European NG-MEDIUM FPGA," in *Proc. NASA/ESA Conf. Adapt. Hardw. Syst. (AHS)*, Aug. 2018, pp. 127–134.
[20] D. Bascones, C. Gonzalez, and D. Mozos, "FPGA implementation of the CCSDS 1.2.3 standard for real-time hyperspectral lossless compression," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 4, pp. 1158–1165, Apr. 2018.
[21] N. Kranitis, A. Tsigkanos, G. Theodorou, I. Sideris, and A. Paschalis, "A single chip dependable and adaptable payload data processing unit," in *Proc. IEEE 21st Int. On-Line Test. Symp. (IOLTS)*, Jul. 2015, pp. 138–143.
[22] M. Bertolucci, F. Falaschi, R. Cassettari, D. Davalle, and L. Fanucci, "A comprehensive trade-off analysis on the CCSDS 131.2-B-1 extended ModCod (SCCC-X) implementation," in *Proc. 23rd Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2020, pp. 126–132.
[23] J. Vidmar, P. Maillard, T. Jones, M. Sawant, G. Gambardella, and N. Fraser, "Space DPU: Constructing a radiation-tolerant, FPGA-based platform for deep learning acceleration on space payloads," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2021, pp. 1–8.

[24] K. O'Neill, A. Severance, and D. Nandi, "Using the VectorBlox software development kit to create programmable AI/ML applications in radiation-tolerant RT PolarFire FPGAs," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2021, pp. 1–8.

[25] K. Bravhar, V. Martins, L. Santos, and D. M. Codinachs, "BRAVE NG-MEDIUM FPGA reconfiguration through SpaceWire: Example use case and performance analysis," in *Proc. NASA/ESA Conf. Adapt. Hardw. Syst. (AHS)*, Aug. 2018, pp. 135–141.

[26] V. Leon, I. Stamoulias, G. Lentaris, D. Soudris, R. Domingo, M. Verdugo, D. Gonzalez-Arjona, D. M. Codinachs, and I. Conway, "Systematic evaluation of the European NG-LARGE FPGA & EDA tools for on-board processing," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2021, pp. 1–8.

[27] G. Lentaris, V. Leon, I. Stamoulias, K. Maragos, and D. Soudris, "Computer vision acceleration on NG-MEDIUM and NG-LARGE FPGAs for rovers," in *Proc. Space FPGA Users Workshop (SEFUW)*, 2020, pp. 1–19.

[28] *ESA and Roscosmos. ExoMars 2022 Mission*. Accessed: Aug. 24, 2021. [Online]. Available: https://directory.eoportal.org/web/eoportal/satellite-missions/e/exomars-2022

[29] T. Takashima, E. Ogawa, K. Asamura, and M. Hikishima, "Design of a mission network system using SpaceWire for scientific payloads onboard the Arase spacecraft," *Earth, Planets Space*, vol. 70, no. 1, pp. 1–9, Dec. 2018.

[30] J. Both and E. Kolbe, "Jena-optronik experience summary on microsemi RTG4 designs," in *Proc. Space FPGA Users Workshop (SEFUW)*, 2018, pp. 1–20.

[31] H. Sierks, H. U. Keller, R. Jaumann, H. Michalik, T. Behnke, F. Bubenhagen, I. BRttner, U. Carsenty, U. Christensen, R. Enge, and B. Fiethe, "The dawn framing camera," *Space Sci. Rev.*, vol. 163, pp. 263–327, Dec. 2011.

[32] W. J. Markiewicz, D. V. Titov, B. Fiethe, T. Behnke, I. Szemerey, H. Perplies, M. Wedemeier, I. Sebastian, W. Boogaerts, C. Dierker, and B. Osterloh, *VMC: The Venus Monitoring Camera*, vol. 1295. Paris, France: ESA Special Publication, 2007, pp. 1–8.

[33] J. P. C. Carrascosa, B. A. del Moral, J. L. R. Mas, M. Balaguer, A. C. L. Jiménez, and J. C. del Toro Iniesta, "The RTE inversion on FPGA Aboard the solar orbiter PHI instrument," *Proc. SPIE*, vol. 9913, pp. 1438–1452, Jul. 2016.

[34] D. Gonzalez-Arjona, R. Domingo, P. Bajanaru, F. A. Stancu, A. Alexe, S. Sincan, and D. Gogu, "On-board complex image-processing based on FPGA acceleration for autonomous navigation in space," in *Proc. Eur. Workshop Board Data Process. (OBDP)*, 2019, pp. 1–23.

[35] P. Bajanaru, R. Domingo, D. Gonzalez-Arjona, F. A. Stancu, C. Onofrei, M. Marugan, R. Chamoso, and C. Mihalache, "Reconfigurable co-processor for spacecraft autonomous navigation," in *Proc. Eur. Workshop board Data Process. (OBDP)*, 2021, pp. 1–8.

[36] Xilinx. *Past, Present, Future Xilinx on Mars Rovers*. Accessed: Aug. 24, 2021. [Online]. Available: https://forums.xilinx.com/t5/Adaptable-Advantage-Blog/Past-Present-Future-Xilinx-on-Mars-Rovers/ba-p/944915

[37] NASA. *Mars 2020 Perseverance Landing Press Kit*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.jpl.nasa.gov/news/press_kits/mars_2020/landing/mission/spacecraft/perseverance_rover/

[38] NASA. *Mars 2020 PIXL*. Accessed: Aug. 24, 2021. [Online]. Available: https://mars.nasa.gov/mars2020/spacecraft/instruments/pixl/

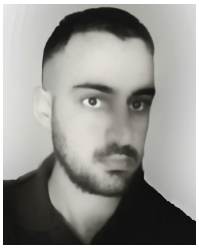
[39] F. Abazovic. *Mars Rover Perseverance Runs Xilinx*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.acanalysis.com/mars-rover-perseverance-runs-xilinx/

[40] Spacelab. *Missions and Projects*. Accessed: Aug. 24, 2021. [Online]. Available: https://spacelab.ufsc.br/en/projects/

[41] D. S. Lee. *Commercial Field-Programmable Gate Arrays for Space Processing Applications*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.osti.gov/biblio/1458107

[42] NASA. *State of the Art Small Spacecraft Technology*. Accessed: Aug. 24, 2021. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/soa2018_final_doc.pdf

[43] G. Lentaris, I. Stamoulias, D. Diamantopoulos, K. Maragos, K. Siozios, D. Soudris, M. A. Rodrigalvarez, M. Lourakis, X. Zabulis, I. Kostavelis, L. Nalpantidis, E. Boukas, and A. Gasteratos, "SPARTAN/SEXTANT/COMPASS: Advancing space rover vision via reconfigurable platforms," in *Proc. Int. Symp. Appl. Reconfigurable Comput. (ARC)*. Cham, Switzerland: Springer, 2015, pp. 475–486.



**VASILEIOS LEON** received the Diploma degree in computer engineering from the University of Patras, Greece. He is currently pursuing the Ph.D. degree with the School of Electrical and Computer Engineering, National Technical University of Athens, Greece. He is also working as a Researcher and a Developer in projects of European Space Agency with embedded computing platforms and SoCs (FPGA, VPU, and TPU). He has published more than 15 research papers in peer-reviewed conferences (including a paper in the Design Automation Conference) and journals (including articles in IEEE and ACM Transactions). His research interests include digital circuit design, hardware accelerators, digital signal processing, embedded systems, approximate computing, and computer arithmetic.

**IOANNIS STAMOULIAS** received the M.Sc. degree in microelectronics, specialized in design of integrated circuits, from the National and Kapodistrian University of Athens (NKUA), Greece, and the B.Sc. degree in computer science and telecommunications, in 2010 and 2007, respectively. He is currently a Ph.D. student with the Department of Informatics and Telecommunications of NKUA. His research interests include digital signal processing systems, embedded systems, network-on-chip, hardware/software co-design, computer vision, and so on.

**GEORGE LENTARIS** received the B.Sc. degree in physics, the M.Sc. degree in logic, algorithms, and computation, the M.Sc. degree in electronic automation, and the Ph.D. degree in computing from the National and Kapodistrian University of Athens (NKUA), Greece. He is currently a Research Associate with the National Technical University of Athens (NTUA), working on HW/SW co-design with single-, multi-, and SoC-FPGA platforms, for space applications for ESA. His research interests include parallel architectures and algorithms for DSP, digital circuit design, image processing, computer vision, video compression, and reliability of FPGA chips.

**DIMITRIOS SOUDRIS** (Member, IEEE) received the Diploma and Ph.D. degrees in electrical engineering from the University of Patras, Patras, Greece, in 1987 and 1992, respectively. Since 1995, he has been a Professor with the Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece. He is currently a Professor with the School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece. He has authored or coauthored more than 500 papers in international journals/conferences. He has coauthored/coedited seven Kluwer/Springer books. He is also the leader and a principal investigator in research projects funded by Greek Government and Industry, European Commission, ENIAC-JU, and European Space Agency. His current research interests include high-performance computing, embedded systems, reconfigurable architectures, reliability, and low-power VLSI design. He was a recipient of the award from INTEL and IBM for EU Project LPGD 25256; ASP-DAC 05 and VLSI 05 awards for EU AMDREL IST-2001-34379, and several HiPEAC awards. He has served as the general/program chair in several conferences.

**DAVID GONZALEZ-ARJONA** received the M.S. degree in computer science and the M.Phil. degree in telecommunications engineering and computer science from the Autonomous University of Madrid. He currently works with GMV Aerospace and Defence SAU, while in parallel works part-time as an Associate Professor with the Autonomous University of Madrid. He is specialized in avionics, equipment, and on-board SW for the Space Segments and Robotics Business Unit, GMV, where he has been working, since 2011. He is also the Head of the Space Equipment Section, leading as directly and managing different ESA projects related to microelectronics for space, autonomous vision-based navigation accelerators, communication boards, and avionics computers, and participating in research and development projects as ENABLE-S3 H2020 or space-based surveillance tracking on Galileo next generation. As a part-time Associate Professor, he teaches with the Department of Electronic and Telecommunication Technology, providing a state-of-the-art point of view from academic and researching perspective.

**RUBEN DOMINGO** received the degree in telecommunications engineering from the Technical University of Madrid (UPM). He currently works with GMV Aerospace and Defence SAU on the avionics & on-board SW for the Space Segments and Robotics Business Unit. He is also working in projects involving development of embedded HW and SW with focus on the latest years on the new European SRAM-based Rad-Hard FPGA from NanoXplore (BRAVE), where GMV performed a quality evaluation of both the BRAVE SW tools for NG-MEDIUM and NG-LARGE in ESA's projects QUEENS-FPGA and QUEENS2. He has also worked now on developments in BRAVE NG-LARGE device, on real-time critical SW for next ESA rover's autonomous navigation targeting the ARM R5 embedded in the NG-LARGE thanks to NXARTAN-BB ESA-funded project. He has also developed different communication and control IPs on BRAVE FPGAs, as a high-performance LVDS inter FPGA link or a CMV4000 camera sensor controller with SpaceWire interface. He worked in the development of the telemetry encoder and firmware controller for the MIURA1 micro-launcher processing avionics.

**DAVID MERODIO CODINACHS** received the M.Sc. degree in electronic engineering from the Politecnico di Torino and the M.Sc. degree in telecommunications engineering from the Universitat Politècnica de Catalunya. He has been an ASIC/FPGA Engineer with the European Space Agency (ESA), since 2005. He has been providing support on the use of FPGAs and digital ASICs to several ESA projects and missions, especially for Earth observation, telecom, and science. Prior to joining ESA 16 years ago, he has been working as an ASIC Design Consultant for Alcatel Bell (now, Nokia) and Philips (now, NXP) for micro-electronics designs in telecommunications and medical/audio products.

**ISABELLE CONWAY** received the Diplôme d'Ingénieur degree in microelectronique et automatique from Institut des Sciences de l'Ingénieur de Montpellier (ISIM), France. She is currently an Engineer in electronics and robotics by education from ISIM, and a software/system engineer by trade. She has 18 years of experience in the aeronautical industry and 15 years of experience in the space industry. Her experience covers systems engineering, SW engineering, and vendor development, with special interest in safety and security aspects and standardization. For the last 12 years, she has been working for European Space Agency, ESTEC, The Netherlands, with the Technical and Quality Department, supporting multiple missions in navigation, telecommunication, and Earth observation.