The UP2DATE Baseline Research Platforms

Álvaro Jover-Alvarez*†, Alejandro J. Calderón*§, Iván Rodriguez*†, Leonidas Kosmidis†*
Kazi Asifuzzaman†, Patrick Uven‡, Kim Grüttner‡, Tomaso Poggi§, Irune Agirre§

*Universitat Politècnica de Catalunya (UPC)
†Barcelona Supercomputing Center (BSC), Spain
‡OFFIS, Germany
§IKERLAN, Spain

***Abstract*—The UP2DATE H2020 project focuses on high-performance heterogeneous embedded platforms for critical systems. We will develop observability and controllability solutions to support online updates while ensuring safety and security for mixed-criticality tasks. In this paper, we describe the rationale behind the selection of the baseline research platforms which will be used to develop and demonstrate the project concepts, including a performance comparison to identify the most efficient one.**

I. The UP2DATE Project

The UP2DATE H2020 project [1], which is currently in its early stages addresses an extremely relevant process which is widely used in cyber-physical systems nowadays, software updates. Software updates add new functionalities and provide fixes for security vulnerabilities, however, they introduce the challenge to preserve the safety of the system. This problem becomes even more complex considering that in an attempt to reduce costs, critical systems are moving towards consolidation of multiple functionalities on fewer, more powerful, high performance platforms.

In particular, this implies that the platform can be shared by functionalities with different criticality, a concept known as mixed-criticality. In UP2DATE we will develop software update solutions which will tackle these challenges and demonstrate them in practice. This will be achieved by developing observability and controllability services.

The project consortium consists of 7 partners, 3 of which academic (IKERLAN, Barcelona Supercomputing Center and OFFIS) and 4 industrial (TTTech Auto, Ingenieurgesellschaft Fuer Auto und Verkehr GmbH, Marelli and CAF Signaling), which bring case studies from the automotive and the railway domain for the demonstration of our concepts.

The automotive industrial case studies come with certain requirements which limit the potential selection of the platforms to specific boards currently in use by each company. Therefore, for the automotive case studies the platforms have been preselected by the corresponding industrial partners.

On the other hand, for the railway case study and for the research platforms we have a higher flexibility for the platform selection. This allows us to research on more ambitious concepts with lower TRL but with a high potential impact for future updateable cyber-physical systems.

In this paper, we outline the platforms we will use in the project specifically for the research proposals. We describe the rationale behind the selection of the initial candidate platforms and the results of a benchmarking study to select the most appropriate research platforms for the project.

II. Platforms Requirements

The selection of the research platforms has been performed based on two requirement categories: a set of requirements described in the project call and in the submitted project proposal, and a second set of requirements derived from the techniques we plan to develop during the project.

A. Requirements from Call for Proposals / UP2DATE Description of Work (DoW)

The UP2DATE project summary defines a set of high-level requirements which are in line with the requirements described in the ICT-01-2019 call for proposals:

Heterogeneous platforms: The hardware platform and the developed solutions need to address not only CPUs but also at least one accelerator: GPU (Graphics Processor Unit) or FPGA (Field Programmable Gate Array / Programmable Logic). These solutions are increasingly employed in the embedded and safety critical domains since they can provide high performance with low-power, which is a high-level requirement of the ICT-01-2019 call.

High-Performance: Accelerators like GPUs or FPGAs can provide the extreme performance requirements of advanced functionalities as shown by industry decisions e.g. in the automotive sector for autonomous driving. GPUs are easier to program compared to FPGAs in the case that new developments are required.

Observability: To support this, we need the target platforms to provide access to performance counters to derive metrics and signatures, which will be used for safety and security.

Controllability: We need the target platforms to support mechanisms to modify the execution conditions e.g. changing the resources used by tasks or dealing with contention.

On-line updates: Target platforms need to offer the possibility to replace software, including while they are operating.

Safety and Security: The selected target platforms should possess a safety and security certifications or contain features which can help to obtain such certifications.

Mixed Criticality: the selected platforms should support the co-existence of both high and low criticality tasks, while preserving safety and security. A widely used solution in industry to achieve this goal is virtualisation.

Platform Characterisation/Benchmarking: Finally, the selection of the platforms should be based on the comparison of the characterisation (performance, real-time properties, power consumption) based on the execution of benchmarks.

 https://doi.org/10.23919/DATE51398.2021.9474040

TABLE I
CHARACTERISTICS OF THE UP2DATE CANDIDATE PLATFORMS.

Platform Type	**Multi-core CPU**	**GPU**	**FPGA**	**Memory**
Research and Railway Case Study				
NVIDIA Jetson TX2	4 ARM Cortex A57 and 2 NVIDIA Denver	NVIDIA Pascal	No	8GB
NVIDIA Xavier AGX	8 ARMv8 Compatible NVIDIA Carmel	NVIDIA Volta	No	32GB
Xilinx Ultrascale+ ZCU102	4 ARM Cortex A53 and 2 ARM R5	ARM Mali-400 (Graphics Only)	Yes	4GB
Renesas R-CAR H3	4 ARM Cortex A57, 4 A53, one ARM R7	PowerVR GX6650	No	4GB

B. Requirements from software layers

Next, we describe the requirements imposed by our initial analysis of the techniques we envision to use throughout the project, in order to address particular requirements.

Partitioning: Partitioning refers to the ability of providing a certain degree of isolation between software entities and it is a commonly used concept in safety critical systems for increasing both the safety and security of the system.

Space partitioning isolates the memory space of a software entity, so that in case it suffers a malfunction or security flaw, this issue will not have an impact to the rest of the system. A typical feature required for the implementation of space partitioning is the presence of a memory management unit (MMU) or a memory protection unit (MPU) in the platform.

Time partitioning ensures that each software entity has its own allocated time budget during which it can execute, so that its WCET (Worst Case Execution Time) is lower than its deadline, thus ensuring timely execution and contributing to the safety of the system. With respect to security, time partitioning ensures that the timing behaviour of a task, cannot be exploited by another one. There are two different ways to implement it. The first one is through isolation of resources so that one software task does not impact the timing of another software task, such as in the case of Single Core Equivalence [2]. Other solutions take contention into account instead of trying to avoid it completely. In particular, they inflate the WCET based on measuring contention through performance counters [3] [4] or account for the worst interference [5].

III. CANDIDATE PLATFORMS SELECTION

Based on the list of our requirements, we performed a preliminary selection of the platforms which we benchmarked.

Between the GPU or FPGAs, both solutions have their benefits for the project. For this reason, we decided to select two heterogeneous platforms, one with each accelerator, in order to increase the completeness of our solutions. However, we will prioritise one of the two platforms, which will be the primary platform which will be used for research. The second platform will be covered in a best-effort manner.

For both types of accelerators, all the embedded solutions are ARM-based. For the GPU platforms, there are two major families of products which are capable of safety certifications: the series of Jetson products of NVIDIA and the R-CAR series of Renesas, which has been selected as a requirement for one of the automotive use cases. In particular, the TX2 aims industrial automation, while the Xavier is designed to meet automotive functional safety certification up to ASIL D. Moreover, the Renesas R-CAR features also ASIL-B certification. In terms of theoretical performance, the Jetson family currently offers the highest performance not only in comparison with the R-CAR but also in the entire embedded market. For this reason, we will choose one of the Jetson platforms for the GPU-featuring research and railway platform.

Regarding the FPGA-based research platform, we select the Zynq Ultrascale+ family of products from Xilinx, which is capable of functional safety certification.

The summary of the selected candidate platforms and their relevant characteristics is shown in Table I.

TABLE II
GPU4S BENCHMARKS [6] PORTED AND USED FOR THE EVALUATION.

Benchmark	**Multi-core CPU**	**GPU**
Matrix Multiplication	OpenBLAS	cuBLAS
Fast Fourier Transform	FFTW	cuFFT
2D Matrix Convolution	OpenMP	CUDA
Finite Impulse Response Filter	OpenMP	CUDA
Local Response Normalisation	OpenMP	CUDA
Max Pooling	OpenMP	CUDA
Rectified Linear Unit (ReLU)	OpenMP	CUDA
SoftMax	OpenMP	CUDA
CIFAR10	OpenMP	CUDA

IV. BENCHMARK SELECTION AND PORTING

Since our candidate platforms are heterogeneous (different CPUs, GPU, FPGA) high performance platforms, we needed a benchmark suite supporting all or most of these elements. However, no real-time/safety critical benchmark suite supports both (multi-core) CPUs and GPU/FPGA.

As a solution, we decided to use GPU4S Bench, an open source benchmark suite developed by BSC for Space On-board Processing Systems [6] in the context of the GPU4S project [7] funded by the European Space Agency (ESA). This suite is designed for the safety critical domain, and includes many representative algorithms from several domains. In addition to the simple algorithms, the suite includes a complex application which implements neural network inference based on a network for CIFAR-10. Since it was designed for GPUs, therefore it included algorithm implementations in both CUDA and OpenCL, as well as reference CPU code for results validation.

Since there was no support for multicore processors, we ported and optimised the benchmark suite to OpenMP, in order to be able to make fair comparison of the CPUs.

The original implementation included also versions of some of the algorithms with vendor optimised libraries, for matrix multiplication and Fast Fourier Transform. We followed the same approach for the multicore CPU port we developed, in order to be able to achieve the maximum performance that each platform can provide for these algorithms. Table II shows the benchmarks which we used, as well as which library implementations our implementations are based on.

V. Research Platforms Performance Comparison

A. Platform Details

The NVIDIA platforms feature multiple power modes, each one with a different maximum power consumption (30W for the Xavier and 15W for the TX2). We use the common 15W power mode for the two boards for fairness. In the case of the Ultrascale+, Xilinx does not specify the maximum power consumption, so we use it in its default power mode.

NVIDIA Xavier: To comply with the real time needs of the project, we enabled the real time features provided in NVIDIA's kernel 4.9.140-rt93. The 15W power mode we selected uses 4 NVIDIA-designed ARM v8 compatible Carmel CPUs and a Volta GPU, both operating at low-frequency (1.2GHz and 670MHz respectively).

NVIDIA TX2: This platform is based on a hybrid architecture, consisting of 4 ARM A57 CPUs and two Denver CPUs, developed by NVIDIA, in addition to a Pascal-based GPU. The performance mode we selected employs the 4 ARM CPUs at their highest frequency (2GHz) and the GPU at a medium frequency (1.12GHz). Selecting a performance mode with equal number of cores for both TX2 and Xavier makes the comparison more straightforward. However, for single core experiments we use also the 15W performance mode with Denver cores, clocked at 2GHz, too.

Xilinx Ultrascale+ ZCU102: For the Xilinx platform we are running the benchmarks in bare metal. Our assumption is that the cores are working at their maximum frequencies of 1.5GHz for the ARM Cortex-A53 cores and 600MHz for the R5 cores as stated in the Xilinx user guide. Given that we are running in a bare metal setup, we are unable to use the multi-core capabilities of the board using OpenMP. For this reason, we perform the comparison of the Xilinx platform with the rest of the platforms only in single core mode. Then, based on the multicore performance we get in the NVIDIA platforms compared to their single core performance, we extrapolate the multicore performance for the Ultrascale. In order to perform the measurements, we access the performance counters in a low-level manner using a mixture of C and assembly. We collect the number of instructions executed for each benchmark as well as the number of cycles that it took them to execute. Then we use the nominal frequency of each processor to convert these measurements in real elapsed time.

B. Results

To evaluate each platform, we collect end-to-end execution times and we compare the calculated speedup of our algorithms for each board, using the Ultrascale as a baseline.

Single Core Performance Comparison: Figure 1 compares the single core performance of the NVIDIA platforms against the performance of the ARM-A53 CPUs of Zynq. We omit results with the real-time R5 cores in the Zynq, due to their very low obtained performance (more than an order of magnitude) compared to the other processing elements. Note that since in Zynq we are running in bare metal, we cannot use the library versions for the matrix multiplication and the FFT. For this reason, all the reported results in this figure are with the same sequential, handwritten implementation. In general we see two trends: first, NVIDIA's custom designed cores perform better than the stock ARM CPUs. Second, the NVIDIA platforms are faster than the Xilinx one.

When we compare the two NVIDIA platforms, we notice that the Carmel CPU of the Xavier has similar performance with the Denver CPU in several benchmarks. However it is faster in FFT and SoftMax and slower in LRN and matrix multiplication. We believe that the reason of this difference is the fact that in the selected power mode, the CPU in the TX2 is clocked higher than in Xavier, therefore favouring the benchmarks with higher arithmetic intensity.

Regarding the ARM CPU designs, the newer and higher-clocked A57 of the TX2, which has an out-of-order microarchitecture performs significantly better in some benchmarks, however there are several ones in which it is outperformed by the in-order A53 of the Zynq, but in a smaller extent.

Xavier's Carmel CPU is faster than the Zynq in all benchmarks except matrix multiplication. In particular Carmel is 2.5x-6x faster for all benchmarks, but almost 10x slower in matrix multiplication. However, this result is particularly strange. We suspect that the reason for this is the fact we are not sure about the operating frequency of the ARM-A53 in the Zynq, so we assume its maximum one for the conversion of cycles to real time. Possibly its actual frequency is much lower or the system is using dynamic frequency scaling, meaning that the performance of Xavier is much higher than 6x over the Zynq. Moreover, when comparing the single core performance of Carmel CPU of Xavier with the ARM-A57 of the TX2, we also see that Xavier is faster in all the benchmarks.

Multi-Core Performance: In Figure 2 we compare the Xavier multi-core performance with the TX2, using the OpenMP version of our benchmarks and libraries whenever available. The figure shows their relative normalised to the TX2. We have used the same benchmarks as before with the same input sizes, but in the cases in which the execution time was too small for these platforms (matrix multiplication and convolution 2D) we have evaluated an additional larger input set. Overall Xavier's CPU performance in the 15W mode is considerably higher than the TX2 in approximately half of the cases, while in the rest of the cases the performance is slower than the TX2 but quite close. We also observe two cases (matrix multiplication and 2D convolution) where the increase of the benchmark's input size reduces the performance benefit of the Xavier over the TX2.

GPU Performance: Figure 3 displays the relative performance of the Xavier's GPU versus the TX2 under the same power mode. The results are clearly favourable for the Xavier, given that in all of the cases, except two, the performance of the Xavier is greater.

CPU to GPU comparison: One of the stronger requirements for our candidate platforms was heterogeneity, since industry employs such solutions for the implementation of applications which require very high performance such as autonomous driving. For this reason, we also validate this hypothesis comparing the relative performance of GPU over

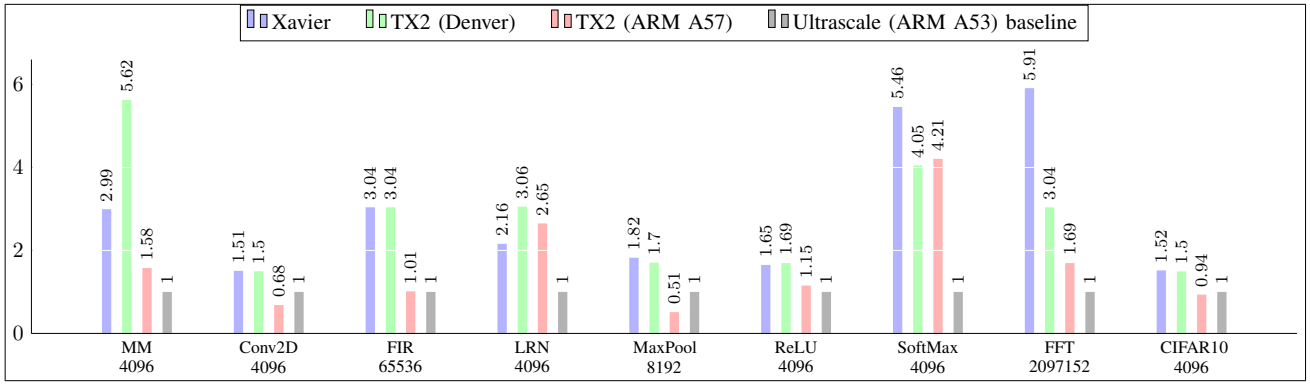

Fig. 1. Single core performance comparison between the CPUs of the three candidate platforms, relative to the performance of Zynq Ultrascale+.

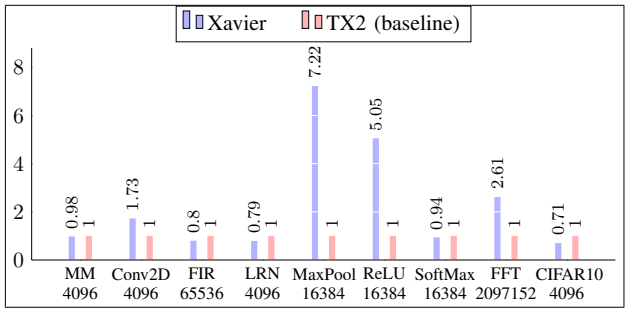

Fig. 2. Relative parallel performance of Xavier over the TX2.

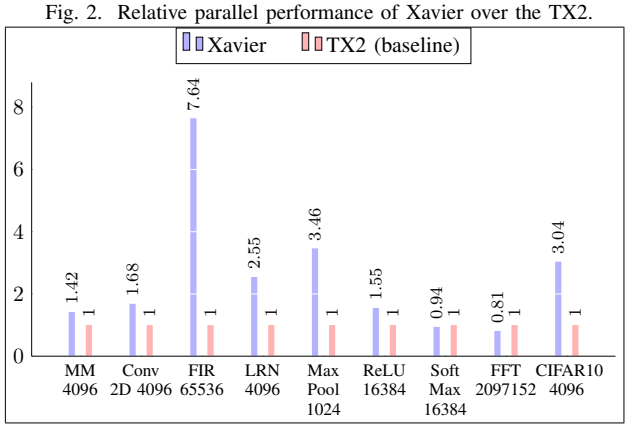

Fig. 3. Relative GPU performance of Xavier over the TX2.

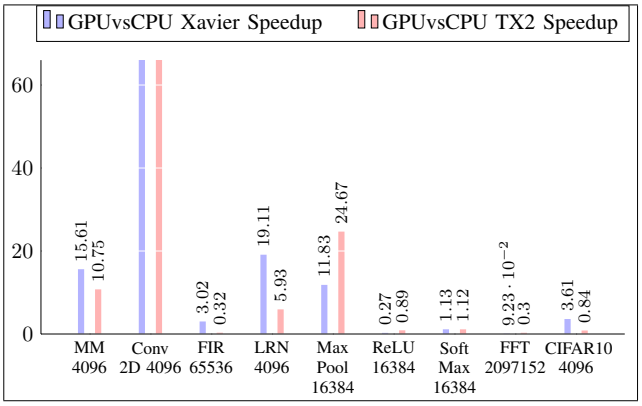

Fig. 4. Relative GPU performance over the CPU in the same SoC.

the CPU of each platform for both the Xavier and the TX2.

Figure 4 shows the results of relative performance of the GPU compared to the multicore CPU performance (obtained with the use of the 4 cores with OpenMP) of each of the considered NVIDIA platforms. Mostly the results of both platforms follow the same trend. There are benchmarks in which the GPU is many times faster than the CPU (5 orders of magnitude), especially in the case of the 2D convolution. Matrix multiplication is also more efficient in the GPU.

Real-time Performance: We have also performed latency tests on the high performance (non real-time) CPUs of the candidate platforms. We observed similar results for all platforms around 50μs, which is acceptable for real-time systems.

VI. Platform Selection and Conclusion

Based on our evaluation, the NVIDIA Xavier is the most powerful platform that we benchmarked. It is more power efficient than the TX2, and given that it supports a higher power mode (30W), more cores (8) and larger memory (32GB), it is safe to assume that it can deliver much higher overall performance, to support a larger number and more complex applications to experiment with our software update, monitoring and controllability solutions. Therefore, this will be the prioritised research platform, followed by the Xilinx Ultrascale+ as a best effort candidate.

Acknowledgments

This work is funded by the European Commission's Horizon 2020 programme under the UP2DATE project (grant agreement 871465). It is also partially supported by the Spanish Ministry of Economy and Competitiveness under grants PID2019-107255GB and FJCI-2017-34095 and HiPEAC.

References

[1] I. Agirre et al., "UP2DATE: Safe and Secure Over-the-air Software Updates on High-performance Mixed-criticality Systems," in *DSD*, 2020.
[2] R. Mancuso et al., "WCET(m) Estimation in Multi-core Systems Using Single Core Equivalence," in *ECRTS*, 2015, pp. 174–183.
[3] E. Díaz et al., "MC2: Multicore and Cache Analysis via Deterministic and Probabilistic Jitter Bounding," in *Ada-Europe*, 2017.
[4] E. Díaz et al., "Modelling Multicore Contention on the AURIX™ TC27x," in *Design Automation Conference (DAC)*, 2018, pp. 1–6.
[5] M. Fernández et al., "Assessing the suitability of the ngmp multi-core processor in the space domain," in *EMSOFT*, 2012.
[6] I. Rodriguez et al., "GPU4S Bench: Design and Implementation of an Open GPU Benchmarking Suite for Space On-board Processing," Universitat Politecnica de Catalunya, Tech. Rep. UPC-DAC-RR-CAP-2019-1, https://www.ac.upc.edu/app/research-reports/public/html/research_center_index-CAP-2019,en.html.
[7] L. Kosmidis et al., "GPU4S: Major Project Outcomes, Lessons Learnt and Way Forward," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2021.