High-Level Synthesis of an Efficient Hardware Implementation for a Smart Tactile Sensing System

María-Luisa Pinto-Salamanca[1][a], Wilson-Javier Pérez-Holguín[2][b] and José Antonio Hidalgo-López[3][c]

[1]*GridSE Research Group, Universidad Pedagógica y Tecnológica de Colombia, Colombia*

[2]*GIRA Research Group, Universidad Pedagógica y Tecnológica de Colombia, Colombia*

[3]*Departamento de Electrónica, Universidad de Málaga, Málaga, Spain*

Keywords: Hardware-Efficient Implementation, Triaxial Contact Forces, Real-Time Embedded Tactile Sensing System.

Abstract: Complex algorithms' execution can be improved by means of carefully designed digital hardware. These take advantage of parallelization techniques, heterogeneous architectures, pipelines, and reuse of functional blocks, among others, to achieve low power consumption, low area overhead, and high real-time operating speeds. However, the design process for these architectures is typically long and complex compared to equivalent software implementations. This paper presents the design process and implementation of a hardware architecture designed to reconstruct triaxial contact forces for innovative tactile sensing systems. Such algorithms were implemented in hardware on a field-programmable gate array (FPGA) platform using a high-level hardware design approach, which allows for a significantly reduced design effort and accelerates the validation process of system functionality. The hardware design flow considers the integrated circuit design cycle and is based on the evaluation of functionality and efficiency metrics. The maximum estimation error for the hardware implementation was around 14.7%, with an average response time of 58.68 ms, a power consumption of 0.871 W, and a speed of up to 65.89 MBps. The hardware design and results obtained here can be applied in different tactile sensing systems such as robotics, prosthetic hands, biosensing, human-computer interfaces, and healthcare.

1 INTRODUCTION

Computationally demanding problems such as mathematical models of complex physical phenomena, machine learning techniques, and signal processing systems can be described by algorithms implemented in software (running on a standard processor) or in dedicated hardware. The key difference between these two methods lies in the way the respective operations are calculated. In the software approach, algorithms are implemented using a programming language, and the resulting instructions are executed sequentially according to the characteristics of the processor architecture (Stallings, 2010). In the hardware approach, the algorithm is implemented using a hardware description language (HDL), in which operations are executed in parallel in a set of functional blocks that process data according to the register transfer level (RTL) principles (Plusquellic, 2017). The hardware implementation is based on the use of combinational circuits such as multiplexers, adders, multipliers, and sequential components like registers, memories, and control units.

The choice between hardware and software implementation approaches depends on design requirements, including flexibility, parallelism, operating conditions, and metrics such as performance, energy efficiency, and power consumption (Schaumont, 2013). Software implementations are more flexible, less complex, and reduce design costs. Hardware implementations, on their part, improve performance, energy efficiency, and power density, albeit with a significant increase in design complexity and development time.

Having an efficient hardware implementation is essential for applications that require speed, real-time computing, complex computation, and large volumes of data, such as Machine Learning (Mohammadi

[a] https://orcid.org/0000-0002-2089-0683
[b] https://orcid.org/0000-0001-5238-4470
[c] https://orcid.org/0000-0001-9505-9141

Pinto-Salamanca, M.-L., Pérez-Holguín, W.-J. and Hidalgo-López, J. A.
High-Level Synthesis of an Efficient Hardware Implementation for a Smart Tactile Sensing System.
DOI: 10.5220/0013839800003982

In *Proceedings of the 22nd International Conference on Informatics in Control, Automation and Robotics (ICINCO 2025) - Volume 1*, pages 524-531
ISBN: 978-989-758-770-2; ISSN: 2184-2809

et al., 2025), signal processing (Xia et al., 2023), digital design (Lifa et al., 2015), and robotics (Schenck et al., 2017), among others.

In particular, smart tactile sensing systems are one of the most computationally demanding applications today. These allow the emulation of the functioning of the sense of touch through the use of sensors, electronic interface circuits, and tactile decoding systems (Dahiya et al., 2010). Tactile sensing systems process contacts from information captured by multiple tactile sensors, and the execution of tactile event reconstruction algorithms running on high-performance hardware or software platforms (Ibrahim et al., 2018b).

Tactile emulation is fundamental in object manipulation and grasping tasks, as well as, replicating the exploratory action of contact, for applications such as human-robot interaction (Han et al., 2024), and prosthetic hands (Ham et al., 2023). Detecting the distribution, intensity, and temporal evolution of forces in real time is critical for both tactile property processing and sliding control loops (Jang et al., 2024).

There are several ways to perform contact force estimation depending on the type of tactile sensor employed and its transduction technology, the force estimation model, and the software or hardware strategy adopted for its implementation. In all cases, tactile emulation imposes that the response must be independent of the contact phenomenon (i.e., generalizable), independent of the contact area (i.e., scalable), and with a predictable response time and minimal resource consumption (i.e., efficient) (Wasko et al., 2019). The transduction technology and the force reconstruction model constrain the generalization and scalability attributes, while software and/or hardware implementation of the model defines the efficiency feature, see Figure 1.

Although there is a large amount of research on tactile sensing systems reported in the literature, most contact force reconstruction implementations are performed on PCs or workstations. In contrast, hardware-integrated tactile sensing systems operating in real time are less common. Among these, the most remarkable works are those describing parallel hardware implementations on Field-Programmable Gate Arrays (FPGAs) (Hosseinabadi et al., 2021). Other authors focus their efforts on electronic interfaces for sensor data acquisition, communication, and accelerated execution, using high-performing analog-to-digital converters (ADCs) (Kerner et al., 2022), graphics processing units (GPU) (Zhao et al., 2025), and single-board computers (or System on Chips) (Ma et al., 2019).

This work presents the design and implementation of efficient hardware architectures to reconstruct triaxial contact forces in the tactile data decoding stage of smart tactile sensing systems. The developed architectures are implemented on an FPGA platform using a high-level hardware design approach, significantly reducing the required design effort and accelerating the system functional validation process. The hardware design flow is based on the typical integrated circuit design cycle and is guided by some metrics evaluation criteria.

This paper is organized as follows. Section 2 summarizes the hardware alternatives for the hardware implementation of efficient architectures and the use of high-level synthesis tools for the entire design process. Section 3 describes the proposed methodology to assess the hardware design of two contact force reconstruction algorithms. Section 4 reports the experimental results and discusses our main findings. Finally, Section 5 concludes the paper and draws future works.

2 EFFICIENT HARDWARE IMPLEMENTATIONS

Some platforms commonly used for the hardware implementation of complex and/or large algorithms include FPGAS, System on Chip (SoCs), FPGA-SOCS, Graphic Processing Units (GPUs), and Application Specific Integrated Circuits (ASICs), among others. Each of these has different advantages and/or disadvantages that can enhance or reduce system performance depending on the characteristics of the application. This makes a correct choice of the platform a crucial technical aspect in the first design stages.

Computer architecture capabilities can be significantly increased through parallelism or heterogeneity strategies, which allow obtaining a low-power architecture for high-speed and real-time applications (Gardezi et al., 2019). Alternatively, multi-core architectures have become a popular solution to the growing demand for computing power in modern applications (Racordon and Buchs, 2016). Other heterogeneous platforms may be composed of combinations of CPUs with GPUs, ASICs, or FPGAs. These are typically geared toward massive data processing supported by the use of hardware accelerators (HW accele) with high energy efficiency and reconfigurable options to improve latency, energy consumption, or hardware flexibility (Zhu et al., 2019).

Regarding efficiency, it can be assumed that the implementation of computationally intensive algorithms is considered ”efficient” when, in addition to meeting the expected functionality of the system, optimal resource management decisions and compliance

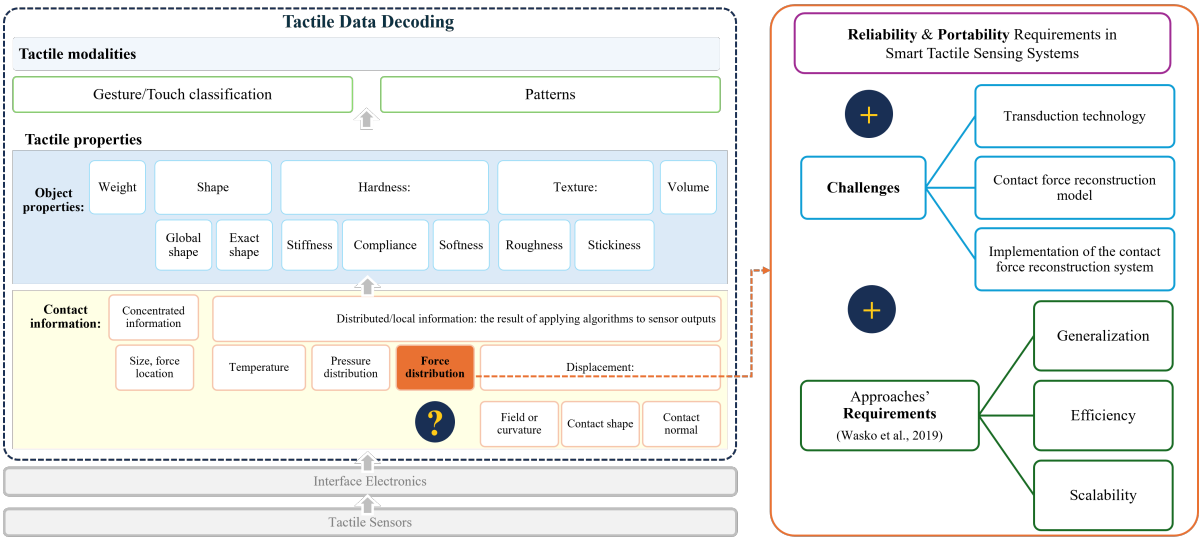

Figure 1: Hardware design challenges in the contact force reconstruction problem.

with constraints are also considered during the design and verification processes. This can be measured through evaluation metrics, such as the use of hardware resources, energy consumption, performance, speed, processing time, memory, and accuracy. The number of metrics considered in each case depends on the complexity of the algorithm implemented.

2.1 Efficient Smart Tactile Sensing Systems

Hardware designers of tactile-sensing systems face challenges such as real-time response, reliability, shrinking area sizes, and low-power consumption. All of these elements constitute a broad field of research usually known as "real-time integrated tactile sensing systems."

(Magno et al., 2017) and (Magno et al., 2018) address efficiency from the system energy consumption point of view, considering wearable and prosthetic applications powered by batteries. In such works, a low-power embedded system based on a parallel ultralow-power processor PULP is used to demonstrate the energy-efficient implementation of tactile data decoding circuits. Those approaches focus on maximizing energy efficiency, improving computational performance by means of two strategies: i) parallel architectures for near-threshold operation, based on multi-core clusters, and ii) low-power fixed-function hardware accelerators coupled with programmable and flexible processors.

In (Alameh et al., 2019), an efficient tactile processing system is proposed for object recognition by using a learning solution to classify three tactile modalities. That work evaluates convolutional neural networks CNN accelerators as an efficient alternative of embedded intelligence based on system-on-module low-power consumption and high performance. (Ibrahim et al., 2018a) demonstrate the implementation of machine learning techniques integrated with approximate computing methods as a solution to tactile data decoding with low-latency, real-time operation, and ultra-low power consumption constraints in high computational complexity applications.

In (Fleer et al., 2020), a machine learning model is presented for the discrimination of unknown object shapes with a rigid tactile sensor array. In that work, the efficiency is seen from the software context as the algorithm's ability to simultaneously execute tasks, such as feature extraction and control strategy calculation, while continuously acquiring data online. For that, the authors formalize the target task of object identification in a reinforcement learning framework. In that sense, efficient smart tactile sensing is also an issue of optimizing multiple variables that can be faced with hardware implementations on embedded electronic systems.

2.2 High-Level Approach to Hardware Design Implementation

Hardware design based on high-level system tools is becoming increasingly popular, as it simplifies and speeds up the development process of complex systems. High-level synthesis (HLS) is an automated design process for digital systems that focuses on mapping behavioral or algorithmic specifications (described in C/C++) onto register-transfer-level (RTL)

structures. This strategy accelerates the verification process for early-stage designs, enabling hardware designers to enhance functional features while improving optimization targets (Skalicky et al., 2013).

The HLS design cycle is performed in several stages. First, the behavioral description of a design is analyzed and architecturally constrained using an HLS tool. Then, the scheduling and trans-compile procedure maps (from the *transaction level model* or TLM to RTL) of the design are carried out in a hardware description language (HDL). Next, a functional validation is carried out in HLS based on two simulation stages: *i) untimed* that focus on validating the functions to be synthesized through functional testbenches, and *ii) RTL cycle-functional testbenches*, and *RTL cycle-accurate* that supports functional comparison between untimed and RTL cycle-accurate results. After that, the translation and verification stages allow the correlation between TLM and RTL structures, supporting the identification and observation of the substructures in a design, which can contribute to the analysis and evaluation of the reliability of digital designs.

3 MATERIALS AND METHODS

3.1 Contact Force Reconstruction Algorithm for Tactile Sensing Systems

The algorithms implemented in this work are the Triaxial Forces Reconstruction Algorithm (TFRA) (Pinto-Salamanca et al., 2023) and the Sparse Triaxial Force Reconstruction Algorithm (SpTFRA) (Pinto-Salamanca et al., 2024). These are based on analytical models and allow the reconstruction of triaxial contact forces using normal stress tactile sensor arrays.

The TFRA and SpTFRA algorithms are constrained to the estimation of triaxial forces for simple, static, and continuous contact conditions on non-deformable plane surfaces. Some positive features of these algorithms include that they allow input data to be independent of the transduction technology used and that they can employ low-cost and widely available commercial tactile sensors, such as piezoresistive, resistive, or capacitive sensors. The latter are highly valued in processing contact forces in robotic or biomedical contexts.

In addition to a software implementation of the mentioned algorithms, they can also be efficiently implemented in hardware, so that the requirements of real-time integrated tactile sensing systems can be met. Thus, during the hardware design, functional verification, and system validation stages, we employ a comprehensive approach that considers both algorithm functionality and hardware efficiency. This approach seeks a hardware implementation of the TFRA and SpTFRA algorithms that meet the criteria defined in (Wasko et al., 2019), which include having a generalizable and scalable implementation that achieves the lowest error estimation, minimizes power consumption, utilizes hardware resources efficiently, and meets memory requirements, while maintaining the highest throughput and the shortest response time.

3.2 Integrated Circuit Design Cycle

Hardware design comprises four steps: i) software implementation features, ii) hardware design, iii) design verification, and iv) system validation.

3.2.1 Software Implementation Features

Software implementations of the TFRA and SpTFRA algorithms were carried out in Matlab® MathWorks™ 2021. These algorithms were validated on two tactile sensor arrays covering areas of 40×20 mm^2 (hereafter sensor 1), and 40×40 mm^2 (hereafter sensor 2), limited to a 100-taxel tactile sensor array for normal stress data inputs up to 50,000 N/m^2 with resultant forces close to 6 N.

The system specification stage allowed for the identification of algebraic requirements, data ranges, and computational complexity. The functional characterization of the TFRA and Sp TFRA algorithms needs to perform some arithmetic operations, such as generalized multiplication of dense matrices by vectors, the square root, and the inverse tangent operation. In the design process, the operations that can be executed using sequential blocks were also identified, along with the memory sizing for the constant matrices and variable vectors in the model. The input and memory load data were pre-encoded using a 32-bit floating-point format.

3.2.2 Hardware Design

The hardware design stage includes a review of design techniques for efficient hardware implementation, high-level design of the systems to be implemented, and the low-level design of such systems in an HDL. The entire hardware design process was conducted using AMD Vitis™ HLS 2022 and Vivado™ Design Suite 2022. The TFRA and SpTFRA hardware implementations were evaluated on a Zynq UltraScale+ MPSoC ZCU102 FPGA development board. The design environment includes high-

level design tools and a design cycle that begins with the description of each functional block using C++. Furthermore, the design flow enables high-level simulation, functional verification with simplified file management, and preliminary synthesis reports before exporting the hardware accelerators.

All arithmetic operations were described using the high-level synthesis math library of AMD®(Advanced Micro Devices Inc.,). The high-level simulation results were validated at the functional level by running test benches in C++ and comparing them with the results of the software implementation in Matlab®.

3.2.3 Design Verification

Design process include a verification at gate level simulation, prototyping on FPGA platform including logic synthesis, place, and route elements. Once the functional behavior of the SpTFRA was verified, the options of filters used were analyzed from the perspective of hardware resource consumption, synthesizing only those of smaller size. In the system testing task, metrics and resource consumption were evaluated at a pre-synthesis level including a comparison for the use of digital signal processors (DSP), block random access memory (BRAM), look-up tables (LUT), and Flip-Flops (FF) elements required by each hardware implementation. Such analysis enables the selection of the best features proposed for the SpTFRA, utilizing sparse matrix approximations with the Modified Compressed Sparse Row format (MCSR) as proposed by (Hosseinabady and Nunez-Yanez, 2020).

This approach, centered on optimizing hardware resource usage and verifying time propagation delay for each case, enables an efficient hardware implementation of the algorithms. The system's response time was verified using Integrated Logic Analyzer Accelerators from AMD® (Advanced Micro Devices,) and the available experimental setup.

3.2.4 System Validation

System validation was conducted through functional evaluation, assessing the force estimation error of each hardware implementation, and verifying the real-time response of smaller implementations in hardware resources and energy consumption, as best throughput. The design methodology followed herein compliance with hardware functionality and efficiency criteria, which were the main guides to the final implemented solutions.

The presented hardware design approach leverages the generalized matrix–vector multiplication to optimize hardware resource consumption by replacing dense matrices with approximate sparse matrices.

The last significant contribution of this work was the analysis and evaluation of vulnerability to transient faults in sparse matrix hardware accelerators when implementing force estimation in safety-critical tactile sensing applications and other fields that rely on large matrices.

4 EXPERIMENTAL RESULTS

Figure 2 shows the evolution and numerous software and hardware implementations analyzed for each algorithm as the design process progresses. During the system specification, 20 software implementations were analyzed for each sensor, of which 2 corresponded to the TFRA evaluation and the remaining 38 were evaluations of different threshold filters provided in the SpTFRA. For these cases, each software alternative involved a memorization process of constant matrices in the system memory. Therefore, for the hardware design stage, only the ten implementations with the lowest force estimation error calculated by the software implementations in each case were described.

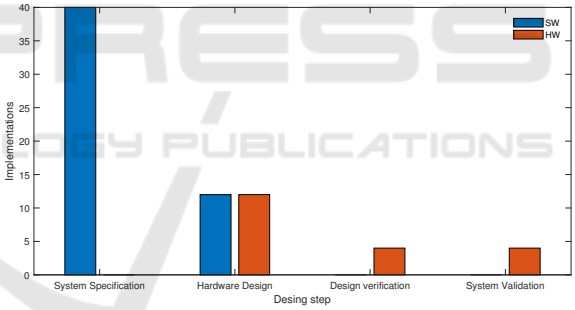

Figure 2: Hardware design flow proposed for the hardware implementation.

The high-level synthesis and simulation reporting tools available in HLS allowed us to select two hardware implementations for each sensor with the best functionality and efficiency. These implementations were then taken to the design verification and system validation levels. The process followed with HLS verified the reduction in the design effort since changes in the algorithms at the software level could be considered with a rapid evaluation of their implications at the hardware synthesis level.

Figure 3 summarizes the behavior of the preliminary synthesis reports obtained by the Vitis™ HLS 2022 tools in the high-level Synthesis (C Synthesys) before exporting the hardware accelerators. At this level, The synthesis report for the TFRA (HW1, HW2

applied in two sensor cases) showed excessive use of DSP modules for the Zynq Ultrascale ZCU102 FPGA. This means that the synthesis of this algorithm couldn't be performed in hardware and that the TFRA implementation was only done at the functional verification level in Xilinx's Vivado. In contrast to the TFRA, the SpTFRA algorithm was effectively implemented and synthesized in hardware, following the same design flow for TFRA, thanks to the reduction in hardware consumption generated by using sparse matrices and data encoding strategies that reduced the number of operations in sparse matrix-vector multiplications.

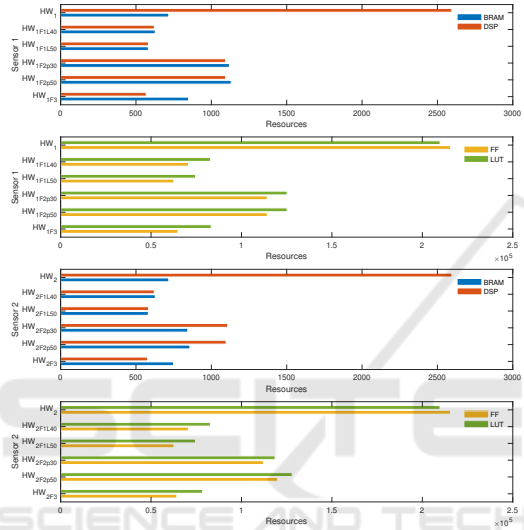

Figure 3: Comparison of hardware consumption for the analyzed implementations

Figure 4 shows a comparison between the high-level hardware simulation and software implementations of the SpTFRA for calculating some g_{pq} internal coefficients of the algorithm in one of the contact cases for Sensor 1. The similarity of the responses indicates that the hardware implementation accurately computes those coefficients. Some differences can be explained by the approximations made by the software vs. hardware math library.

Table 1 summarizes the efficiency evaluation parameters for the TFRA and SpTFRA algorithms. The values considered for the system evaluation were: the maximum relative error in the estimation of the tangential resultant forces (e_{FX}, e_{FY}), the normal resultant forces (e_{FZ}), the friction coefficient (e_{μ}), and the orientation angle of the tangential forces (e_{ϕ}). These errors were calculated from the reference values generated by a Finite Element Analysis (FEA) technique using Comsol Multiphysics® 6.0 software. The efficiency metrics considered are also presented in Table 1, which includes latency, power, throughput, and execution time. Regarding power consumption, for the case of FPGA-based touch sensing systems, authors as (Mendoza-Peñaloza and Muñoz, 2023) report power consumptions of around 2W for all cases, values that are higher than those obtained in this work, which are under 1W. Power consumption results for TFRA implementations could not be obtained because these implementations proved physically unimplementable due to exceeding the capabilities of the hardware platform employed. The SpTFRA hardware implementation stands out for achieving latency metrics much lower than those of TFRA.

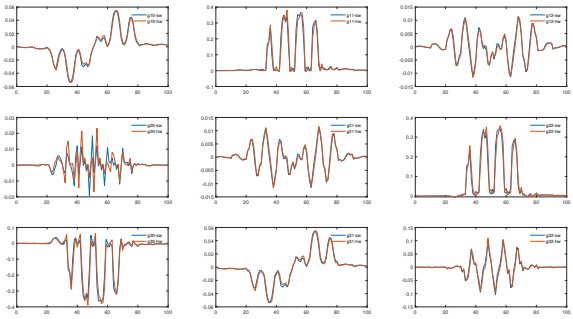

Figure 4: High-level hardware simulation and software implementations of the SpTFRA for calculating g_{pq} coefficients.

The response time was determined using a system clock frequency of 125 MHz. Force estimation for the SpTFRA implementations was performed in an average time of 58.68 ms, which is in the same range as other works reporting real-time operation for force estimation, such as (Xu et al., 2024) (33 ms), and (Alameh et al., 2020) (300 ms). However, all these response times were obtained using software implementations running on PC workstations. Some authors report response times and resource consumption for an 8×8 sensor matrix size (Seminara et al., 2015), while others report resource consumption but do not perform force reconstruction (Mendoza-Peñaloza and Muñoz, 2023). All of this makes it difficult to thoroughly compare these works and the results obtained in this work.

5 CONCLUSIONS

This work presents the design and implementation of two efficient hardware architectures for reconstructing triaxial contact forces in tactile data decoding. Developed architectures are implemented on an FPGA platform using a high-level hardware design approach, which enables to get a significant reduction in the design effort, speeds up the system validation

Table 1: Efficiency performance parameters of the TFRA and SpTFRA algorithms and their hardware implementations.

Parameters	Sensor and Algorithms					
	TFRA	SpTFRA		TFRA	SpTFRA	
SW Ref.	SW_1	SW_{1F1L40}	SW_{1F3}	SW_2	SW_{2F2p50}	SW_{2F3}
e_{FX}	5.94	7.48	7.52	7.84	6.11	6.11
e_{FY}	3.44	4.87	4.23	5.44	5.9	5.89
e_{FZ}	10.93	13.63	13.66	2.69	3.67	3.66
e_{μ}	9.41	12.17	14.31	4.46	12.97	14.92
e_{ϕ}	1.64	1.64	1.64	3.16	3.16	3.16
HW Ref.	HW_1	HW_{1F1L40}	HW_{1F3}	HW_2	HW_{2F2p50}	HW_{2F3}
e_{FX}	5.95	7.45	7.7	6.12	5.51	6.12
e_{FY}	3.45	4.79	4.17	5.9	5.92	5.91
e_{FZ}	11.87	10.83	12.57	3.66	3.37	3.67
e_{μ}	9.42	12.15	14.67	8.97	9.01	8.99
e_{Φ}	1.93	1.93	1.93	2.2	2.2	2.2
Power [W]	3992.9**	0.871	0.894	3992.9**	0.953	0.983
Latency (CLK @ 125 MHz)[ms]	12	5.4	5.9	12	6.4	6.7
Throughput [MBps]		65.89	59.96		62.72	55.32
Execution time [ms]		51.27	51.49		58.52	58.68

** Physically unimplementable.

process, reduces data storage requirements, the number of processing elements, and the energy consumption of the whole system.

This approach is generalizable, which means that it can be replicated for algorithms with similar restrictions on the use of resources, energy consumption, performance, latency, and response time. Similarly, the design method is applicable to other fields of tactile detection, applications, and components for analytical models and data-based hardware applications.

Integrated circuit design methodology allowed us to find an efficient hardware implementation for contact force reconstruction. The process was guided by a rigorous review of the challenge transduction technology, the complexity of the contact force reconstruction model, and the implementation of the contact force reconstruction system, as well as by an evaluation of the algorithm's functionality and hardware efficiency.

Future works could include exploring alternatives for data access, addressing, and reuse for TFRA and SpTFRA implementations that take advantage of the on-chip memory availability of FPGAs and other embedded systems to overcome the bottleneck caused by poor memory access times.

ACKNOWLEDGEMENTS

This research has been partially supported by the projects UPTC VIE SGI 3940.

REFERENCES

Advanced Micro Devices, I. Integrated Logic Analyzer (ILA).

Advanced Micro Devices Inc. HLS Math Library, Vitis High-Level Synthesis User Guide (UG1399), Reader AMD Technical Information Portal.

Alameh, M., Abbass, Y., Ibrahim, A., and Valle, M. (2020). Smart tactile sensing systems based on embedded cnn implementations. *Micromachines*, 11.

Alameh, M., Ibrahim, A., Valle, M., and Moser, G. (2019). Dcnn for tactile sensory data classification based on transfer learning. In *2019 15th Conference on Ph.D Research in Microelectronics and Electronics (PRIME)*, pages 237–240.

Dahiya, R. S., Metta, G., Valle, M., and Sandini, G. (2010). Tactile sensing—from humans to humanoids. *IEEE Transactions on Robotics*, 26:1–20.

Fleer, S., Moringen, A., Klatzky, R. L., and Ritter, H. (2020). Learning efficient haptic shape exploration with a rigid tactile sensor array. *PLoS ONE*, 15.

Gardezi, S. E. I., Aziz, F., Javed, S., Younis, C. J., Alam, M., and Massoud, Y. (2019). Design and vlsi implementation of csd based da architecture for 5/3 dwt. In *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 548–552.

Ham, J., Huh, T. M., Kim, J., Kim, J.-O., Park, S., Cutkosky, M. R., and Bao, Z. (2023). Porous Dielectric Elastomer Based Flexible Multiaxial Tactile Sensor for Dexterous Robotic or Prosthetic Hands. *Advanced Materials Technologies*, 8(3):2200903.

Han, C., Cao, Z., Hu, Y., Zhang, Z., Li, C., Wang, Z. L., and Wu, Z. (2024). Flexible Tactile Sensors for 3D Force Detection. *Nano Letters*, 24(17):5277–5283.

Hosseinabadi, A. H. H., Black, D. G., and Salcudean, S. E. (2021). Ultra low-noise fpga-based six-axis optical

force–torque sensor: Hardware and software. *IEEE Transactions on Industrial Electronics*, 68:10207–10217.

Hosseinabady, M. and Nunez-Yanez, J. L. (2020). A Streaming Dataflow Engine for Sparse Matrix-Vector Multiplication Using High-Level Synthesis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(6):1272–1285.

Ibrahim, A., Osta, M., Alameh, M., Saleh, M., Chible, H., and Valle, M. (2018a). Approximate computing methods for embedded machine learning. In *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 845–848.

Ibrahim, A., Pinna, L., and Valle, M. (2018b). Experimental characterization of dedicated front-end electronics for piezoelectric tactile sensing arrays. *INTEGRATION-THE VLSI JOURNAL*, 63:266–272.

Jang, H., Bae, J., and Haninger, K. (2024). Soft finger grasp force and contact state estimation from tactile sensors.

Kerner, S., Krugh, M., and Mears, L. (2022). Wearable shear and normal force sensing glove development for real-time feedback on assembly line processes. *JOURNAL OF MANUFACTURING SYSTEMS*, 64:668–675.

Lifa, A., Eles, P., and Peng, Z. (2015). On-the-fly energy minimization for multi-mode real-time systems on heterogeneous platforms. Institute of Electrical and Electronics Engineers Inc. Cited By :1Export Date: 10 March 2020.

Ma, D., Donlon, E., Dong, S., and Rodriguez, A. (2019). Dense tactile force estimation using gelslim and inverse fem. In *2019 International Conference on Robotics and Automation, ICRA 2019*, volume 2019-May, pages 5418–5424. Institute of Electrical and Electronics Engineers Inc.

Magno, M., Ibrahim, A., Pullini, A., Valle, M., and Benini, L. (2017). Energy efficient system for tactile data decoding using an ultra-low power parallel platform. In *2017 New Generation of CAS (NGCAS)*, pages 17–20.

Magno, M., Ibrahim, A., Pullini, A., Valle, M., and Benini, L. (2018). An energy efficient e-skin embedded system for real-time tactile data decoding. *JOURNAL OF LOW POWER ELECTRONICS*, 14:101–109.

Mendoza-Peñaloza, J. and Muñoz, D. M. (2023). Hardware implementation of a sliding detection algorithm for robotic hands using force sensors. In *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, pages 1–6.

Mohammadi, A., Kneale-Roby, H., Sadrafshari, S., Bienek, M., Betts, J., and Shokrani, A. (2025). Hardware implementation of convolutional neural network for high-precision machining at the sensor edge. In *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5.

Pinto-Salamanca, M. L., Castellanos-Ramos, J., Pérez-Holguín, W. J., and Hidalgo-López, J. A. (2023). An estimation of triaxial forces from normal stress tactile sensor arrays. *Mechatronics*, 96:103070.

Pinto-Salamanca, M.-L., Pérez-Holguín, W.-J., and Hidalgo-López, J. A. (2024). Hardware Implementation for Triaxial Contact-Force Estimation from Stress Tactile Sensor Arrays: An Efficient Design Approach. *Sensors*, 24(23).

Plusquellic, J. (2017). The nature of hw/sw ii. hw/sw codesign w/ fpgas.

Racordon, D. and Buchs, D. (2016). Verifying multicore schedulability with data decision diagrams. Cited By :1Export Date: 10 March 2020Correspondence Address: Racordon, D.; Centre Universitaire d'Informatique, University of GenevaSwitzerland; email: dimitri.racordon@unige.ch.

Schaumont, P. R. (2013). *A practical introduction to hardware/software codesign*. Springer US.

Schenck, W., Horst, M., Tiedemann, T., Gaulik, S., and Möller, R. (2017). Comparing parallel hardware architectures for visually guided robot navigation. *Concurrency Computation*, 29.

Seminara, L., Capurro, M., and Valle, M. (2015). Tactile data processing method for the reconstruction of contact force distributions. *MECHATRONICS*, 27:28–37.

Skalicky, S. et al. (2013). High level synthesis: Where are we? A case study on matrix multiplication. In *2013 Int. Conf. on Reconfigurable Computing and FPGAs (ReConFig)*, pages 1–7.

Stallings, W. (2010). *Computer Organization and Architecture: Designing for Performance*. Prentice Hall, 8th edition.

Wasko, W., Albini, A., Maiolino, P., Mastrogiovanni, F., and Cannata, G. (2019). Contact Modelling and Tactile Data Processing for Robot Skins. *SENSORS*, 19(4).

Xia, Y., Meng, Y., Xiang, S., Wang, J., and Yang, C. (2023). An efficient hardware implementation of dilated convolution using a novel channel-equivalent decomposition method. In *2023 IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA)*, pages 172–173.

Xu, D., Hong, W., Hu, B., Zhang, T., Chen, D., Yan, Z., Yao, X., Zhang, X., Zhao, Y., Sun, T., Zhang, C., Pan, M., Ruan, X., Yan, R., Wang, J., and Guo, X. (2024). River valley-inspired, high-sensitivity, and rapid-response capacitive three-dimensional force tactile sensor based on u-shaped groove structure. *Smart Materials and Structures*, 33:35006.

Zhao, C., Liu, J., and Ma, D. (2025). ifem2.0: Dense 3-d contact force field reconstruction and assessment for vision-based tactile sensors. *IEEE Transactions on Robotics*, 41:289–305.

Zhu, Z., Zhang, J., Zhao, J., Cao, J., Zhao, D., Jia, G., and Meng, Q. (2019). A hardware and software task-scheduling framework based on cpu+fpga heterogeneous architecture in edge computing. *IEEE Access*, 7:148975–148988.