Board 91: Work in Progress: An Interdisciplinary Subject on Hardware Accelerated Computing

Dr. Glenn J Bradford, University of Melbourne

Glenn J. Bradford is a wireless engineering professional with experience in industry and education. From 2020 to 2023 he was a Teaching Fellow in the Department of Electrical and Electronic Engineering at the University of Melbourne, Australia, where he worked to create innovative curriculum incorporating practical, hands-on experiences to better drive student learning. He worked previously as a wireless systems engineer at both Intel Corp. and Motorola Solutions, Inc. Glenn received his Ph.D. in electrical engineering from the University of Notre Dame in 2014.

Prof. Gavin Buskes, The University of Melbourne

Gavin is a Professor and Deputy Head (Academic) in the Department of Electrical and Electrical Engineering at the University of Melbourne, Australia. He teaches a wide range of engineering subjects and has research interests in optimal control, idea generation, prior knowledge and developing professional skills. He also holds the role of Assistant Dean (Teaching and Learning) in the Faculty of Engineering and Information Technology.

Dr. Paul N Beuchat, The University of Melbourne

Paul is currently a Teaching Fellow at the Department of Electrical and Electronic Engineering at the University of Melbourne. He completed his doctoral degree at ETH Zurich, Switzerland, in 2019 under the supervision of Prof. John Lygeros at the Automatic Control Laboratory. The topic of his dissertation is theoretical guarantees and practical algorithms for Approximate Dynamic Programming. He received the B.Eng. degree in mechanical engineering and B.Sc. in physics from the University of Melbourne, Australia, in 2008, and the M.Sc. degree in robotics, systems and control from ETH Zurich in 2014. Paul's automation research interests are control and optimization of large-scale and robotic systems with applications in the areas of building control and coordinated robotics. Paul's engineering education research and teaching focuses on devising new and adapting existing project-based learning pedagogies for educating the next generation of engineers.

WIP: An Interdisciplinary Subject on Hardware Accelerated Computing

Abstract

In this paper we report on the initial design and delivery of a hardware accelerated computing subject targeted at an interdisciplinary cohort of engineering and computing students. Within the subject, students explore different approaches to accelerating computationally intensive algorithms through customized hardware, with a particular emphasis placed on the use of FPGAs and high-level synthesis (HLS) tools. The subject aims necessitate covering aspects from a diverse range of topics, including fundamentals of digital design, computer architecture, parallel programming, and systems thinking. Although such concepts naturally intersect within the discipline of computer engineering, structural considerations within our master's programs and disparate prior knowledge within our cohort entail students inherently experience the subject as interdisciplinary in nature. This presents numerous challenges in subject design but offers an opportunity for developing interdisciplinary competencies and an appreciation for other disciplinary ways of thinking. Based on instructor observations while teaching, we reflect on the successes and shortcomings in the subject's design that impact interdisciplinary knowledge development. We then conclude with proposed revisions to address identified shortcomings.

Introduction

Today's engineering graduate will have the opportunity to work on a set of unique and meaningful problems with aspects spanning multiple disciplines. For example, meeting many of the current global development goals [1], such as providing sustainable energy, clean water and sanitation, or high-quality education, will require a seamless integration of knowledge and methods from numerous technical and non-technical disciplines. Intuitively, an engineering graduate that has developed interdisciplinary competencies will be well-suited for solving such challenges, where interdisciplinarity may be considered "as attempts to address real-world cases and problems by integrating heterogeneous knowledge bases and knowledge making practices" [2]. These heterogeneous knowledge bases and practices may include both different engineering and science disciplines as well as non-technical disciplines that illuminate the economic, social, and cultural dimensions of a given problem. Such a well-rounded perspective has long been high on the list of desirable attributes sought in future engineering graduates [3]. For example, ABET's accreditation requirement that students be able to solve complex engineering problems includes solving problems with "many components or sub-problems, involving multiple disciplines, or having significant consequences in a range of contexts" [4].

In this work-in-progress paper, we report on the initial design of a new subject within our coursework master's program that teaches the techniques of hardware acceleration [5][6] to an interdisciplinary cohort of engineering (electrical and mechatronics) and computing (software engineering and information technology) students. Much of the conceptual material sits squarely within the discipline of computer engineering, but recent trends in computer architecture [5][6]

and limited exposure to the topic within both our engineering and computing curricula creates an opportunity to develop interdisciplinary competencies. Ideally, through formal teaching and learning activities and associated peer interactions, students will be able to apply methods and approaches from their peers' discipline, appreciate other disciplinary ways of thinking, and practice the synthesis of knowledge and methods from the other discipline with their own domain expertise. These three capabilities will serve as our definition of interdisciplinary engineering competencies within this paper.

We begin with a brief overview of relevant literature on interdisciplinary engineering education and explain the suitability of hardware acceleration for developing such competencies. We next detail the subject's design and, based on instructor observations and student feedback, reflect on the successes and limitations encountered in the subject's first offering. Among the shortcomings identified in reflection are the balance of disciplinary knowledge exercised, providing sufficient scaffolding to the different cohorts within the subject, and reliably assessing whether interdisciplinary knowledge and ability have been gained. Such challenges have previously been identified in the literature, and so we discuss our experience in light of these reported findings. Finally, we propose possible future subject modifications along with qualitative and quantitative metrics to be collected in support of a rigorous evaluation of its design.

Background

Interdisciplinary Engineering Education

A common taxonomy found in the literature distinguishes between a *multidisciplinary* setting, in which two or more disciplines are present but remain largely distinct in the outcome of any interaction, and an *interdisciplinary* setting, in which a significant *integration* occurs synthesizing knowledge and perspectives from multiple disciplines into something new [7]. Our interest lies primarily in this latter setting, as the challenges faced by future engineering graduates are unlikely to be solved by simple divide-and-conquer approaches. There is also discussion in the literature as to whether interdisciplinarity is a distinct *competency* that can be developed within a student or whether it is simply a *description* of the desired knowledge or solution produced, but it is generally now understood to be "both a process and outcome" [8].

Despite a range of publications exploring interdisciplinary engineering education, it is still an active area of research with numerous open questions and a lack of clear guidance for subject designers. Van den Beemt et al. [2] extensively document major themes that have emerged in recent publications on the topic, examining the motivations for interdisciplinary engineering education, its teaching aspects, and its necessary support structures. We now highlight some of the key themes they identified. A common justification for interdisciplinary learning is to equip engineers to solve complex real-world problems in a socially aware manner. To this end, many interdisciplinary learning experiences are designed around a single disciplinary cohort and introduce knowledge and methods external to that discipline, while others bring together students from multiple disciplines in a common setting. Systems approaches and project-based learning (PjBL) pedagogies are key enablers in this context, but care must be taken that a vagueness of

vision for the purpose of interdisciplinary learning does not result in unclear learning goals. As might be expected, pedagogical considerations common to PjBL are highly relevant for interdisciplinary subject design, especially team formation aspects, providing an appropriate level of scaffolding, and balancing disciplinary contributions within the project.

A key observation made in [2] is that greater insight into the definition and assessment of interdisciplinary learning outcomes is needed by subject designers for rigorous constructive alignment [9]. Some useful guidance for subject design, however, can still be drawn from results in the literature. Richter and Paretti [7] propose a number of interdisciplinary learning outcomes based on surveying the literature and their own case study, which include the ability of students to:

- "Identify contributions that new areas of knowledge can make to their own disciplinary expertise;"
- "Identify ways in which their disciplinary expertise can contribute to the solution of interdisciplinary problems;"
- "Identify the value and contributions of other areas of expertise to a particular interdisciplinary challenge;" and
- "Synthesize both concepts and approaches from multiple domains to develop an integrated solution."

In terms of assessment, Mansilla et al. [10] developed a rubric for evaluating interdisciplinary writing, with criteria that include:

- *Purposefulness* – multidisciplinary nature of the problem is given clear framing.
- *Disciplinary grounding* – disciplinary knowledge and methods are used correctly.
- *Integration* – perspectives from multiple disciplines are present and combined in a balanced manner.
- *Critical awareness* – an understanding of the limitations of the work is demonstrated.

Although specifically designed for the evaluation of interdisciplinary writing, these criteria are relevant and easily generalizable to assessments of other formats.

Hardware Accelerated Computing

Modern computing systems are elaborately complex, having evolved over decades through the combined effort of computer engineers, computer scientists, electrical engineers, software engineers, and information technologists. This is in addition to any domain-specific expertise needed to realize applications of interest, e.g., in the modeling of genomics, fluid mechanics, etc. Table 1 highlights major levels of abstraction in computer system design, where each level is the domain of specialized practitioners with extensive expertise. At the very least, this is a multidisciplinary setting in which a systems approach is necessary to manage complexity and allow practitioners to collaborate with their peers at adjacent levels.

Table 1: Levels of abstraction in computer system design.

Abstraction Level	Example Practitioners
User Applications	Computer Scientists & Software Engineers
Information Systems Management	Information Technologists
High-level Languages	Computer Scientists & Software Engineers
Operating Systems	Computer Scientists & Software Engineers
Instruction Set Architectures	Computer Engineers
Microarchitecture	Computer Engineers
Digital Logic	Electrical and Computer Engineers
Circuit Design	Electrical Engineers
Semiconductor Fabrication	Physicists & Electrical Engineers

Much of the exponential gain in computing performance has been driven by a raw increase in processing power afforded by advances in semiconductor fabrication, which have in turn been leveraged by computer architects, first through the use of instruction level parallelism and subsequently through the addition of parallel processing cores [5]. With the breakdown of Moore's Law, however, new approaches are needed to continue increasing performance, with one of the most promising approaches being the spread of *domain specific accelerators* (DSAs) [5][6]. DSAs are hardware cores tailored to perform computations for a specific *domain* of applications, with performance gains over general-purpose processors achieved through the use of specialized arithmetic units, reduced instruction overhead, and an expanded use of parallelism and memory locality [6]. The successful design of a DSA requires a thorough understanding of algorithms within the domain, with *codesign* of algorithms and hardware being a common practice. This synthesis of knowledge from different domains is interdisciplinary in nature and "will require vertically integrated design teams that understand applications, domain-specific languages, …, computer architecture and organization, and underlying technology" [5]. DSA design thus promises to be an excellent topic by which to expose students to interdisciplinary engineering practices.

Recently, a number of innovative subjects on hardware acceleration [11]-[13] have appeared at both the graduate and undergraduate levels, often with a strong emphasis on designing accelerators for machine learning applications. At UCSD, an undergraduate subject on embedded systems [11] teaches students accelerator design through the lens of parallel programming and has generated an open-source textbook [14] of illustrative projects. High-level synthesis (HLS) for FPGAs [15] is a key enabling tool deployed in most subjects to lower the of barrier entry to hardware design and increase the speed of design space exploration. At Georgia Tech, a subject [12] is taught that covers similar material at the graduate level with many of the lecture examples and laboratories focusing on the acceleration of neural networks. A subject at UC Berkeley [13] focuses exclusively on hardware design for machine learning but, again, relies on HLS as the tool to facilitate exploration.

Subject Design

Cohort Prior Knowledge

In designing any new subject, it is important to identify relevant student characteristics and prior knowledge, however this is even more critical for interdisciplinary cohorts in which these qualities vary greatly. Both our engineering and computing students are enrolled in separate coursework master's programs lasting two to three years. Students in all programs are a combination of pathway students, from our undergraduate bachelor's programs, and lateral entry students, from other institutions, with a significant proportion of students having an international background. These factors all increase the variation in student prior knowledge to be managed.

Table 2 lists the prerequisite knowledge we assume of each cohort along with complementary knowledge some students may have gained in elective subjects. For engineering students, we assume a fundamental understanding of digital logic design and basic programming abilities with some possible exposure to microprocessor architectures. For computing students, we assume familiarity with the modeling of computations and strong programming abilities, with some possible exposure to parallel programming and cloud computing concepts.

Table 2: Assumed prior knowledge in subject cohorts.

	Engineering	**Computing**
Prerequisite Knowledge	Digital Logic Embedded Systems C Programming	Computational Models Software Modeling and Design
Complementary Knowledge	Microprocessor Design	Parallel Programming Cloud Computing

Learning Outcomes and Syllabus

Subject intended learning outcomes (SILOs) for the initial offering addressed disciplinary learning outcomes, primarily in the domain of computer engineering but also with complementary outcomes in the electrical and software engineering domains. On successful completion of the subject, a student is expected to be able to:

1. Explain hardware accelerator architectures and computational models.
2. Implement digital logic functions in a variety of industrial tools, such as hardware description languages, high-level synthesis tools, and OpenCL.
3. Analyze advanced computational algorithms for amenability to hardware acceleration, mapping to heterogeneous computing systems to exploit parallel computation.

4. Design hardware to accelerate computationally intensive algorithms using a systems-based approach with consideration given to trade-offs in speed, energy efficiency, and area.
5. Articulate the importance of hardware acceleration for computational systems as a continuing trend in industry.

Interdisciplinary competencies are not explicitly mentioned in the current SILOs, but there is an implicit need for interdisciplinarity to meet SILOs 3-4, i.e., creating DSAs requires combining domain algorithm knowledge with an understanding of computer architecture concepts, all while considering performance constraints imposed by digital logic.

Figure 1 outlines the main conceptual topics covered in lecture over the 12-week long semester. To establish a common knowledge baseline among the cohorts, the semester began with a review of relevant content from electrical and software engineering. From the engineering side, this included a review of digital logic design, FPGA architectures, and the use of a hardware description language (HDL). From the computing side, this included models of computation and basic parallel programming concepts. The majority of the semester was then spent covering hardware accelerator design as expressed through parallel programming concepts applied via HLS. HLS allows a more rapid exploration of hardware design spaces than is possible with traditional HDLs and alleviates the need for computing students to master an HDL, a notoriously time-consuming affair. Introductory material still covers the basics of the Verilog HDL so that key concepts and limitations of hardware design are not abstracted away from students seeing the topic for the first time. The final weeks of semester introduce convolutional neural networks (CNNs), which are the basis of the subject's design project, and supplementary knowledge on graphic processing units (GPUs) as a point of comparison.

Week	1	2	3	4	5	6	7	8	9	10	11	12
	Review - computational models - digital design - Verilog HDL - FPGA architectures				Accelerator Design - high-level synthesis - hardware parallelism - memory locality - arithmetic specialization - codesign					Supplemental - CNN basics - CNN optimizations - GPUs		

Figure 1: Syllabus of conceptual topics covered in lecture.

Assessment

PjBL strategies are well suited for exercising interdisciplinary competencies, as selecting an appropriate project can setup interdisciplinary interactions. In line with a PjBL pedagogy, half of the points awarded in the subject are for completing a significant design project, including workshop-based assessments introducing the design tools, a set of intermediate project milestones, and a final project report. Individual written exams still account for the remaining half of subject points as it is important to ascertain whether an individual student has developed

competencies outside of their specific discipline. Table 3 summarizes the assessment breakdown in the subject.

Table 3: Subject assessment and associated weighting.

Assessment	Weight
Guided Workshops	10%
Mid-semester Exam	10%
Project Milestones	15%
Project Report	25%
Final Exam	40%

CNN Design Project

The assigned project for semester was to design a hardware accelerator for the inference step of a convolutional neural network trained to perform image super-resolution (SRCNN) [16]. This task is similar to the second laboratory in Georgia Tech's subject [12] but requires implementing a complete network rather than a single layer, which ensures sufficient project scope based on the allocated assessment weighting. Accelerators were designed using AMD's Vitis HLS tool [15] and deployed on AMD's Kria KV260 AI Vision Starter Kit [17], a development board built around a Zynq Ultrascale+ MPSoC with a quad core ARM processor system (PS) and substantial programmable logic (PL) fabric.

Students worked on the project in teams of two that were self-selected in composition. The guided workshops in the first half of semester served to familiarize students with the tools and techniques employed to complete the project. These initial workshops also gave students the chance to identify a partner for the project. Full project details were then released in week eight of semester, and teams were assigned three milestones as checkpoints for instructors to give formative and summative feedback: the submission of a C++ (golden) reference implementation, a 15-minute project update presentation, and a final written report documenting all design and performance results. Lectures covering the basics of CNNs and potential optimization approaches were timed to coincide with when teams were ramping up their investigations for the project.

The majority of computational load for a CNN is in performing multidimensional multiply accumulate (MAC) operations at each convolutional layer. Despite the relative simplicity of the algorithm, the project afforded teams ample optimization opportunities along multiple dimensions. At a macroarchitecture level, teams had to make intelligent choices about memory management to minimize global memory access and pipeline reading, computation, and writing latencies. These macroarchitectural considerations were made more complex by the need to apply successive convolutional layers within the CNN. At the microarchitecture level, teams had to make design choices about how to parallelize the MAC operations among multiple processing

engines (PEs). Those teams that converted the algorithm from its original floating-point implementation to fixed-point arithmetic could reduce PE footprint, possibly trading off fidelity to the original model against increased parallelism from the instantiation of more PEs. Stronger teams within the cohort also explored alternative architectures beyond a straightforward tiled-based implementation of the network.

Discussion

In total, 25 students enrolled in the subject with the breakdown between cohorts being two thirds engineering students and one third computing students. Student feedback, both formal and informal, was generally positive with computing students expressing a strong appreciation for obtaining greater exposure to the hardware underlying their typical software designs. Feedback from engineering students was generally more mixed, with a common theme being that too much time was spent on the review of digital design.

Balance of Disciplinary Knowledge

A major instructor concern entering semester was whether students in one cohort would be at an advantage based on their prior knowledge, however there was no discernible disparity between the final grade distributions of the two cohorts. Additionally, the vast majority of project teams, whether mono- or interdisciplinary, produced a reasonable outcome on the SRCNN project.

Although no cohort grade disparity existed, this does not mean the balance struck between disciplinary content was optimal. Upon instructor reflection, and in alignment with student feedback, it was determined that the start of semester review was heavily weighted toward prerequisite material from electrical engineering, particularly the basics of digital logic design. This level of review was deemed necessary to ensure computing students had the conceptual knowledge to make sense of later discussions on accelerator design and performance tradeoffs, but it is clear many engineering students did not find value in the extent of review. The challenge of balancing the learning needs of two different cohorts is a common theme arising in interdisciplinary engineering education [2]. Two major adjustments are planned for the second offering to address this imbalance. The first adjustment is to include a more extensive review of topics from computing, particularly the use of computational models. The second adjustment is to shift much of the prerequisite review to offline learning modules that students can complete independently. This will allow students to work through prerequisite material at their own pace, possibly spending less time on topics with which they are already familiar. This format will maintain sufficient lecture time to cover new material.

Although not reported in student feedback, another instructor observation was that the project could better utilize expertise from both disciplinary cohorts. In its current form, the SRCNN project is the implementation of a machine learning algorithm for a computer vision application, two fields closely associated with software engineering. This means the project domain aligns well with the computing cohort's background; however, in practice the solutions produced by the project teams were primarily hardware-based in nature, methodologies for which more closely

align with the digital design experience of the engineering cohort. In the next subject offering, it would be desirable that the project also necessitate a significant application of methods and techniques from software development. One way this could be achieved is by selecting a different target algorithm for which a significant fraction of the computational complexity is naturally implemented in software rather than only parallel hardware. Alternatively, the SRCNN algorithm could be retained but reframed as one part of a larger application involving the integration of the accelerator with other software-based subsystems.

Interdisciplinary Learning Outcomes

The current subject design relies on an implicit interdisciplinary experience created by bringing together two distinct student cohorts and addressing a topic sitting at the intersection of those two disciplines. To best achieve the development of interdisciplinary competencies it is important to avoid vagueness of purpose [2], and so the next subject iteration should revise learning outcomes to include an explicit focus on interdisciplinary outcomes. This should include learning outcomes grounded in both the individual cohort disciplines as well as outcomes with an integrative aspect [7][8]. Existing SILO 2 already covers disciplinary methods from electrical engineering and can be modified to include language on how such techniques can "contribute to the solution of interdisciplinary problems" [7]. An explicit learning outcome covering techniques from the computing side, most likely in relation to parallel computational models, should be added with similar verbiage about its interdisciplinary relevance to accelerator design. Finally, SILOs 3-4 can be modified to make explicit the dependence of hardware accelerator design on the synthesis of knowledge and techniques from both the engineering and computing domains.

With a modification to learning outcomes, a corresponding adjustment to assessment is warranted to maintain proper constructive alignment. Revisions to the design project discussed in the previous section will help ensure both disciplinary outcomes (related to digital design and software design) and interdisciplinary outcomes (related to the design of hardware accelerators through the synthesis of disciplinary knowledge) are thoroughly assessed. Additionally, project teams in the revised subject will shift from being self-selected to instructor-assigned to increase the number of close interdisciplinary interactions occurring between the cohorts. Naturally, the efficacy of this change relies on a balance in the enrollment numbers from each cohort.

Subject Evaluation

Finally, evaluation of this initial subject iteration has largely been informal, based on student feedback and instructor observations. It is important that the next offering collect metrics relevant to establishing the efficacy of the subject's design in meeting its interdisciplinary objectives. The large percentage of subject points awarded for written exams (totaling 50% for the mid-semester test and final exam) affords the chance to measure individual achievement relative to the stated disciplinary and interdisciplinary learning outcomes. Specific exam questions can be written targeting each discipline as well as the synthesis of disciplinary knowledge required for hardware accelerator design. Student performance on questions covering their own field of study relative to those that cover the other discipline would then be of interest,

as would any asymmetries in statistics between the cohorts. Furthermore, the final report assignment can be modified to require a reflection on the design approach taken by each team. These statements would then be useful qualitative data for determining the extent to which teams are taking interdisciplinary design approaches and the value they place on contributions from the other discipline [7].

Conclusion

In this paper we have detailed our design of a subject on hardware accelerate computing taught to an interdisciplinary cohort of engineering and computing students at the master's level. Hardware acceleration is a natural topic for use in exploring interdisciplinary competencies due to an inherent need for synthesizing diverse knowledge and methodologies. Although the first subject offering was relatively successful based on student feedback and instructor observations, a number of important adjustments are planned for the next iteration. These include establishing a better balance between disciplinary learning outcomes, making explicit the interdisciplinary outcomes expected of students, and incorporating more formal subject design evaluations for future analysis.

References

[1] United Nations Sustainable Development Goals. https://sdgs.un.org/goals. Accessed 2024-01-21.

[2] A. Van den Beemt, M. MacLeod, J. Van der Veen, A. Van de Ven, S. Van Baalen, R. Klaassen, and M. Boon, "Interdisciplinary engineering education: A review of vision, teaching, and support," *Journal of Engineering Education*, *109*(3), 508-555, 2020.

[3] E. Crawley, J. Malmqvist, S. Ostlund, D. Brodeur, and K. Edstrom, *Rethinking Engineering Education: The CDIO Approach*. Springer, 2014.

[4] ABET Criteria for Accrediting Engineering Programs, 2022-2023. https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accreditingengineering-programs-2022-2023/. Accessed 2024-01-22.

[5] J. L. Hennessy and D. A. Patterson, "A new golden age for computer architecture," *Communications of the ACM*, *62*(2), 48-60, 2019.

[6] W. J. Dally, Y. Turakhia, and S. Han, "Domain-specific hardware accelerators," *Communications of the ACM*, *63*(7), pp.48-57, 2020.

[7] D. M. Richter and M. C. Paretti, "Identifying barriers to and outcomes of interdisciplinarity in the engineering classroom," *European Journal of Engineering Education*, *34*(1), pp.29-45, 2009.

[8] L. R. Lattuca, D. B. Knight, H. K. Ro, and B. J. Novoselich, "Supporting the development of engineers' interdisciplinary competence," *Journal of Engineering Education*, *106*(1), pp.71-97, 2017.

[9] J. Biggs, "Enhancing teaching through constructive alignment," *Higher education*, *32*(3), pp.347-364, 1996.

[10] V. B. Mansilla, E. D. Duraisingh, C. R. Wolfe, and C. Haynes, “Targeted assessment rubric: an empirically grounded rubric for interdisciplinary writing,” *The Journal of Higher Education*, *80*(3), pp.334-353, 2009.

[11] UCSD CSE237C Testing and Validation of Embedded Systems. http://kastner.ucsd.edu/ryan/cse237c/. Accessed 2024-01-22.

[12] Georgia Tech ECE-8893 Parallel Programming for FPGAs. https://sharclab.ece.gatech.edu/teaching/2022-spring-fpga/. Accessed 2024-01-22.

[13] UC Berkeley EE290-2 Hardware for Machine Learning. https://inst.eecs.berkeley.edu/~ee290-2/sp20/. Accessed 2024-01-22.

[14] R. Kastner, J. Matai, and S. Neuendorffer, “Parallel Programming for FPGAs,” *ArXiv eprints*, 1805.03648, May 2018.

[15] AMD Vitis HLS. https://www.xilinx.com/products/design-tools/vitis/vitis-hls.html. Accessed 2024-01-22.

[16] C. Dong, C. C. Loy, K. He and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295-307, Feb. 2016.

[17] AMD Kria KV260 AI Vision Starter Kit. https://www.xilinx.com/products/som/kria/kv260-vision-starter-kit.html. Accessed 2024-01-22.