

Embedded FPGA Developments in 130nm and 28nm CMOS for Machine Learning in Particle Detector Readout

J. Gonski,¹ A. Gupta,¹ H. Jia,^{1,2} H. Kim,¹ L. Rota,¹ L. Ruckman,¹ A. Dragone,¹ and R. Herbst¹

¹*SLAC National Accelerator Laboratory, 2575 Sand Hill Road, M/S 96, Menlo Park, CA 94025, USA*

²*Stanford University, 450 Jane Stanford Way, Stanford, CA 94305, USA*

E-mail: jgonski@slac.stanford.edu

ABSTRACT:

Embedded field programmable gate array (eFPGA) technology allows the implementation of reconfigurable logic within the design of an application-specific integrated circuit (ASIC). This approach offers the low power and efficiency of an ASIC along with the ease of FPGA configuration, particularly beneficial for the use case of machine learning in the data pipeline of next-generation collider experiments. An open-source framework called "FABulous" was used to design eFPGAs using 130 nm and 28 nm CMOS technology nodes, which were subsequently fabricated and verified through testing. The capability of an eFPGA to act as a front-end readout chip was tested using simulation of high energy particles passing through a silicon pixel sensor. A machine learning-based classifier, designed for reduction of sensor data at the source, was synthesized and configured onto the eFPGA. A successful proof-of-concept was demonstrated through reproduction of the expected algorithm result on the eFPGA with perfect accuracy. Further development of the eFPGA technology and its application to collider detector readout is discussed.

KEYWORDS: Reconfigurable Computing; Open Source; FPGA; ASIC; Machine Learning

Contents

1	Background	1
2	Developments in 130nm CMOS	3
2.1	eFPGA Customization	3
2.2	ASIC Digital Architecture	3
2.3	Fabrication	4
2.4	Testing Results	4
2.4.1	Simple Counter Test	4
2.4.2	ASIC Power Draw	5
3	Motivation for Transitioning from 130nm to 28nm CMOS Technology	6
4	Developments in 28nm CMOS	6
4.1	eFPGA Customization	6
4.2	ASIC Digital Architecture	7
4.3	Fabrication	8
4.4	Testing Results	8
4.4.1	Simple Counter Test	8
4.4.2	ASIC Power Draw	8
4.4.3	AXI Stream Loopback in the eFPGA	9
5	Application for Machine Learning-based At-Source Processing	10
6	Summary	13

1 Background

Silicon microelectronics technology is a key piece of the modern computational paradigm, offering high performance and low power options for data processing tasks. A diversification of microchip designs offer the ability to customize computing solutions to a particular problem. Common options such as application-specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs) offer trade-offs between performance, power consumption, and flexibility to suit a wide variety of applications. While an FPGA is fully customizable, it draws more power than an ASIC; however, the ASIC requires considerable expertise to design, and is fixed to a certain task once fabricated. Microelectronics remain a rapidly developing field, with advances promising to deliver even denser logic and faster processing.

In parallel, artificial intelligence and machine learning (AI/ML) have become essential tools in data science. In light of increasing dataset sizes and expanding computational capacity, ML-based

methodology can execute common scientific tasks such as signal-noise classification, regression of key quantities, fast generation of simulation, or anomaly detection, with good efficiency and performance. Leveraging ML in these scenarios requires a high degree of reconfigurability in computing architecture, such that weights and biases of a model, as well as the model itself, can be updated throughout training and over time as the task evolves.

Many data processing tasks have a requirement to perform sophisticated AI inference within resource-constrained systems, such as with low power or latency, motivating the incorporation of ML into microelectronics. This is enabled by “fast ML”, wherein high-level synthesis of software algorithms enables the implementation of ML models in hardware. In this work the *embedded FPGA* (eFPGA) technology is studied as an option to dynamically reconfigure logic to enable ML within an ASIC. eFPGAs comprise a digital block of FPGA fabric that can be embedded into an ASIC design. In this way they can offer the best of both worlds between FPGA and ASIC processing, with lower power demand than an FPGA but with the ease of firmware-based reconfiguration. eFPGAs also mitigate issues of commercial FPGA obsolescence by providing a hardware structure that can be continually updated with newer firmware.

In high energy collider experiments like the Large Hadron Collider (LHC) at the CERN laboratory in Geneva, Switzerland, the ability to execute reconfigurable and fast logic is indispensable. The LHC collides bunches of protons at a frequency of 40 MHz, creating thousands of particles that pass through detectors surrounding the beam interaction point. These conditions create a requirement to handle a total data rate of hundreds of terabytes of data per second, with latencies as low as $\mathcal{O}(\text{ns})$. Furthermore, the high radiation doses and stringent spatial constraints associated to the collision environment motivate the development of custom ASICs and electronics to read out the detector channels at high speed with good fidelity.

Fast ML is being explored in the collider context and across the sciences [1]. A common application for fast ML at colliders is with the trigger system, which comprises hardware- and software-based algorithms that decide which collision events should be written to disk based on an assessment of potentially interesting physics activity [2, 3]. ML in the trigger systems of LHC experiments is being studied to perform fast featurization [4] or anomaly detection [5] on incoming data. Before the trigger, front-end on-detector readout electronics take low-level information collected by detector elements, perform simple operations such as amplification or digitization, and transmit this data off-detector for further processing. The information collected by on-detector readout electronics ultimately feeds the trigger algorithm and the eventual data acquisition, making this information an essential deliverable of the detector.

Existing work to implement ML in the front-end focuses on dedicated ASIC designs for a particular architecture or model [6, 7], which, while functional, is limiting in scope due to the rigidity of the hardware. eFPGAs can fill a crucial technology gap for high energy physics (HEP) experiments by providing an ASIC that can allow not only the reconfiguration of ML model weights, but the entire ML architecture itself onto the ASIC. As the transmission of data off the detector is often less efficient than data processing, the performance of sophisticated data compression and processing as close to the front-end as possible is a key task for modern collider experiments. Such capability will become crucial for the more challenging environments of future colliders [8], where experiments will have to cope with data rates on the exascale with even higher radiation doses and limited space for material budget or cooling services. As a result, community driven exercises on

detector R&D for HEP list increased intelligence on-detector as a key priority for the success of future detector upgrades and designs [9, 10].

This work describes the design and fabrication of eFPGAs using the 130nm and 28nm CMOS technology nodes. The open-source “FABulous” framework is used [11] to generate the eFPGA fabric. An application of the 28nm eFPGA for at-source ML-based classification is performed, using data from simulated silicon pixel sensors in a high energy collider detector. The unique features of the eFPGA, namely its high degree of flexibility, relatively low power and footprint, and publicly accessible design framework make it an ideal tool for known data acquisition challenges in collider physics.

2 Developments in 130nm CMOS

2.1 eFPGA Customization

The ability to customize the tiles of the eFPGA is provided by the FABulous framework [11]. Because this was our very first FABulous eFPGA to design, extreme conservatism was practiced, and the reference design provided by FABulous, known as “fabric_TSMC_example”, was utilized, which has been proven suitable for tapeout. A .csv file is used by FABulous framework to customize the eFPGA tile configuration, which is shown in Figure 1 for the 130nm eFPGA. The eFPGA tiles utilized in this design included:

- W_IO: A 2-bit general-purpose input/output (GPIO) interface with tri-mode support
- RegFile: A simple dual-port LUTRAM, consisting of 32 entries of 4 bits
- DSP_top & DSP_bot pair: Digital signal processor (DSP) primitive, providing 8x8 multipliers with 20-bit accumulators
- LUT4AB: A logic cell comprising a Look-Up Table (LUT) with 4 inputs and a Flip-Flop (FF)
- CPU_IO: An interface of 8-bits per tile from CPU to eFPGA, and 12-bits per tile from eFPGA to CPU
- NULL: An empty tile, used for terminating at eFPGA corners
- N_term_single2: Used for terminating the column on the northern side
- s_term_single2: Used for terminating the column on the southern side

The total resources for the 130nm eFPGA consisted of 384 logic cells, 128 registers, and 4 DSP slices.

2.2 ASIC Digital Architecture

A block diagram of the 130nm ASIC’s digital architecture is shown in Figure 2. For register access from an external FPGA to this ASIC to be performed, a SLAC Ultimate Gateway Operational Interface (SUGOI) digital block is utilized. SUGOI is a simple packet-based control protocol that allows memory-mapped resources within ASICs to be read and modified using 8B10B serial

	A	B	C	D	E	F	G	H	I	J
1	FabricBegin									
2	NULL	N_term_single2	N_term_single2	N_term_single	N_term_single	N_term_single	N_term_single	N_term_single	N_term_single	NULL
3	W_IO	RegFile	DSP_top	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
4	W_IO	RegFile	DSP_bot	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
5	W_IO	RegFile	DSP_top	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
6	W_IO	RegFile	DSP_bot	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
7	W_IO	RegFile	DSP_top	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
8	W_IO	RegFile	DSP_bot	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
9	W_IO	RegFile	DSP_top	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
10	W_IO	RegFile	DSP_bot	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	LUT4AB	CPU_IO
11	NULL	S_term_single2	S_term_single2	S_term_single	S_term_single	S_term_single	S_term_single	S_term_single	S_term_single	NULL
12	FabricEnd									

Figure 1. Screenshot of the 130nm eFPGA tile configuration file.

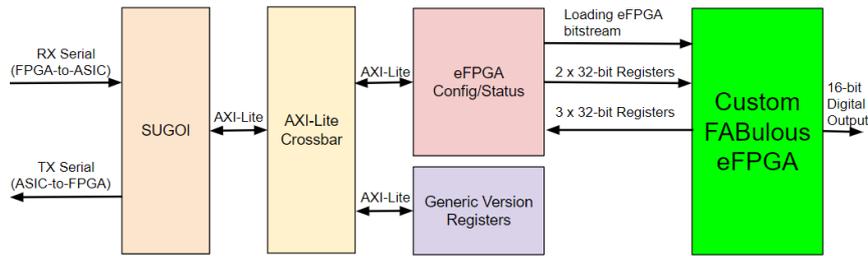


Figure 2. Block diagram of the 130nm CMOS ASIC design.

encoding [12]. Communication by the SUGOI module is carried out via an AXI-Lite [13] crossbar to two separate AXI-Lite endpoints: an eFPGA configuration/status module and a generic version registers module. Diagnostic information about the ASIC, such as the git hash of the code when the ASIC was taped out and the revision number of the ASIC, is provided by the generic version registers module. The eFPGA status/configuration module is employed for loading the bitstream into the eFPGA itself, for reading from multiple 32-bit buses of the eFPGA, and for driving multiple 32-bit buses to the eFPGA. A 16-bit digital output bus via the W_IO tiles for monitoring/debugging on a digital logic analyzer is featured by the eFPGA.

2.3 Fabrication

The submission of this ASIC design to the Taiwan Semiconductor Manufacturing Company (TSMC) 130nm Multi Project Wafer (MPW) was completed in May 2022, and the design was received in August 2022. The design of the custom Printed Circuit Board (PCB) carrier adhered to the FPGA Mezzanine Card (FMC) form factor [14]. Firmware and software were developed to control and monitor the ASIC through an AMD KCU105 development board [15]. A photograph showing the 130nm CMOS ASI wire-bonded to a custom PCB carrier can be seen in Figure 3, as well as a photograph displaying the ASIC on the FMC card alongside the KCU105.

2.4 Testing Results

2.4.1 Simple Counter Test

The basic functionality of the eFPGA was evaluated through the compilation and loading of the bitstream containing simple 16-bit counter firmware into the eFPGA. A photograph showing the

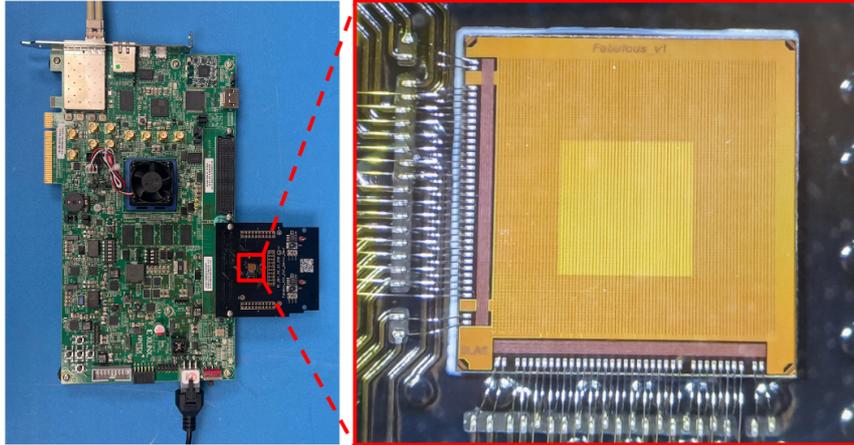


Figure 3. (Left) Photograph of the KCU105 development board with the custom FMC ASIC carrier with ASIC wire bonded to it. (Right) Zoomed in photograph of the 130nm CMOS ASIC (5mm x 5mm).

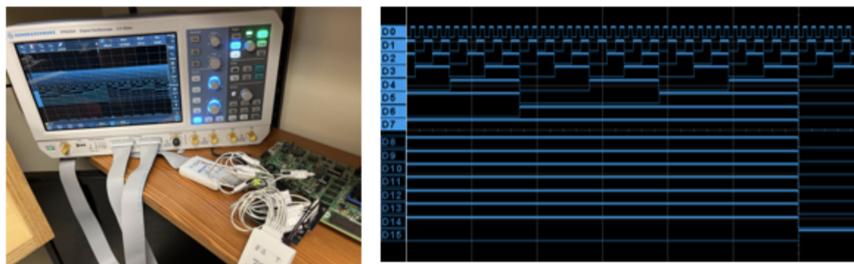


Figure 4. Photograph of the 130nm ASIC connected to a logic analyzer (left), and logic analyzer display of the eFPGA loaded with a simple 16-bit counter bitstream (right).

setup for this test can be seen in Figure 4. The bus of the eFPGA counter was interfaced with the GPIO tiles, which subsequently interfaced with a 16-pin header. This header was then connected to a digital logic analyzer, facilitating the observation of the eFPGA 16-bit counter firmware's behavior. The behavior of the firmware was observed to be as anticipated, thus demonstrating the successful loading of the bitstream.

2.4.2 ASIC Power Draw

To measure the power draw, the same firmware from the 16-bit counter test was used, and the clock frequency of the ASIC was varied. This clock generated by the KCU105's FPGA is sourced by the ASIC's SUGIO, AXI-Lite endpoints, and the eFPGA itself. At clock frequencies ranging from 10 to 125 MHz, both the ASIC's core rail (+1.2VDC) and I/O rail (+1.2VDC) current draws were measured to calculate the total ASIC power drawing. A plot of these measurements is shown in Figure 5.

When the clock frequency exceeded 74 MHz, a lock on the SUGIO link was still achieved by the ASIC, whereas the FPGA failed to achieve such a lock. The problem was traced back to the ASIC's output driver, which was determined to be excessively slow. The slew rate of this output driver was measured to be a rising edge of 38 ns and a falling edge of 32 ns. Conversely, the rising

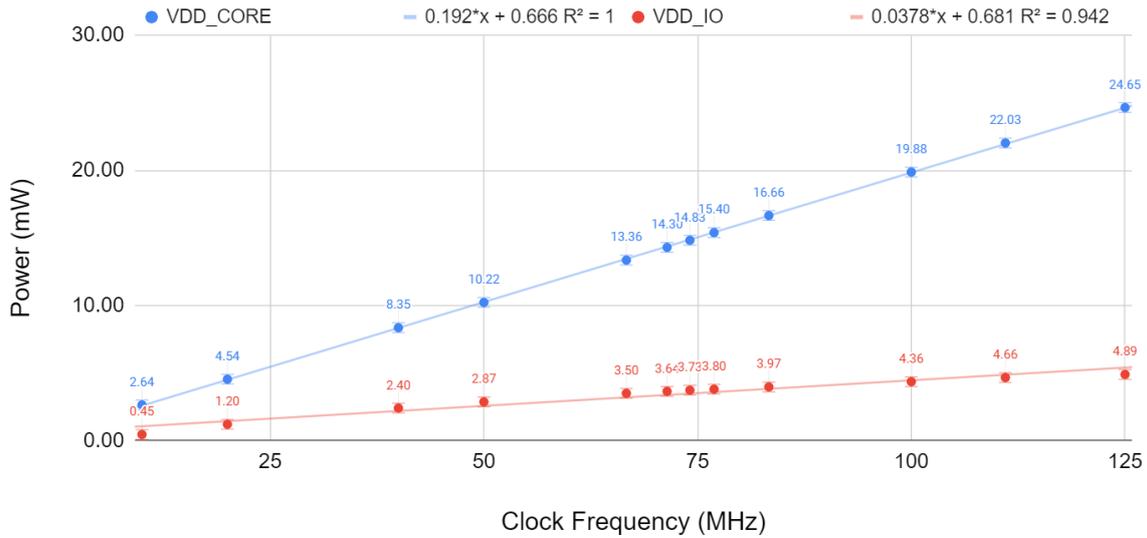


Figure 5. 130nm ASIC power as a function of clock frequency (blue: core voltage supply, red: I/O supply).

and falling edges for the FPGA were found to be on the order of a few nanoseconds. Beyond a clock frequency of 74 MHz, it was possible to only set the ASIC registers via the SUGOI interface, with the functionality to read them back being unavailable. Measurements were ceased after 125 MHz, as this was the timing constraint employed in the place and route software for the ASIC’s digital logic.

3 Motivation for Transitioning from 130nm to 28nm CMOS Technology

The motivation for the transition from 130nm to 28nm CMOS technology was driven by the pursuit of increased logic density, enhanced radiation hardness [16–19], and reduced power consumption. This shift was necessitated by the need to accommodate evolving eFPGA-based data processing requirements. Through the utilization of a smaller CMOS technology node, significant improvements in the performance and adaptability of eFPGA architectures were made possible. The optimization of the physical characteristics of the eFPGA, to take advantage of the benefits offered by the 28nm technology, was the focus of the design choices. Specifically, the decreased feature size of the 28nm process allowed for the integration of more complex logic circuits at higher densities, resulting in a factor of 21 improvement in area efficiency and a factor of 2.8 reduction in core power consumption at 100 MHz. Combined with the need to keep pace with foundry production capabilities, this development also ensures the eFPGA can keep pace with industry standards and be viable for large-scale production towards future collider detector needs.

4 Developments in 28nm CMOS

4.1 eFPGA Customization

The 28nm eFPGA design is similar to the 130nm eFPGA configuration described in Section 2.1. The primary distinction between the 28nm eFPGA and the 130nm eFPGA configuration lies in

	A	B	C	D	E	F	G	H	I	J
1	FabricBegin									
2	NULL	N_term_single	N_term_single	N_term_single	N_term_single	N_term_DSP	N_term_single	N_term_single	N_term_single	NULL
3	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_top	LUT4AB	LUT4AB	LUT4AB	EAST_IO
4	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_bot	LUT4AB	LUT4AB	LUT4AB	EAST_IO
5	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_top	LUT4AB	LUT4AB	LUT4AB	EAST_IO
6	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_bot	LUT4AB	LUT4AB	LUT4AB	EAST_IO
7	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_top	LUT4AB	LUT4AB	LUT4AB	EAST_IO
8	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_bot	LUT4AB	LUT4AB	LUT4AB	EAST_IO
9	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_top	LUT4AB	LUT4AB	LUT4AB	EAST_IO
10	WEST_IO	LUT4AB	LUT4AB	LUT4AB	LUT4AB	DSP_bot	LUT4AB	LUT4AB	LUT4AB	EAST_IO
11	NULL	S_term_single	S_term_single	S_term_single	S_term_single	S_term_DSP	S_term_single	S_term_single	S_term_single	NULL
12	FabricEnd									

Figure 6. Screenshot of the 28nm eFPGA tile configuration file.

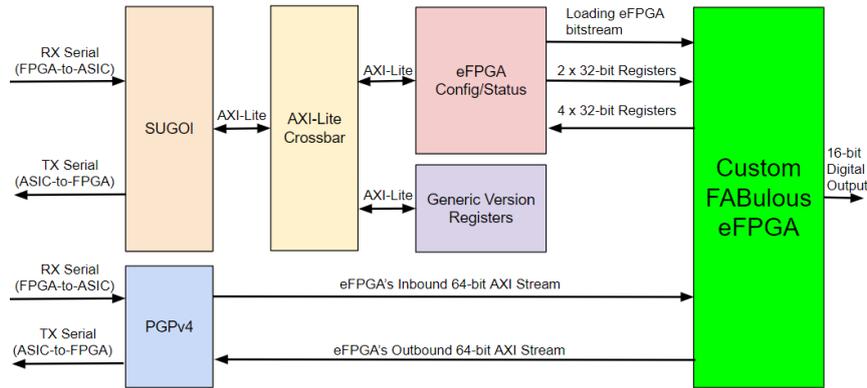


Figure 7. Block diagram of the 28nm CMOS ASIC design.

the removal of “RegFile” tiles, which have been substituted by LUT4AB tiles. Additionally, the replacement of W_IO tiles with WEST_IO tiles, and CPU_IO tiles with EAST_IO tiles, is made. “User defined” tiles, namely WEST_IO and EAST_IO tiles, have been developed to enhance the interconnectivity between the eFPGA and the remaining digital logic within the ASIC. All the other tiles are defined in Section 2.1. The .csv file is used by FABulous framework to customize the 28nm eFPGA tile configuration is shown in Figure 6. In total, the 28nm eFPGA comprises 448 logic cells and 4 DSP slices.

4.2 ASIC Digital Architecture

A block diagram of the 28nm ASIC’s digital architecture is shown in Figure 7. The digital architecture is very similar to the 130nm digital architecture described in Section 2.2, with two major differences. The first is the number of 32-bit buses from the eFPGA to the eFPGA configuration/status module, which was increased from three to four. The second is the addition of AXI streams [20] to/from the eFPGA. The AXI streams to/from the eFPGA are connected to a Pretty Good Protocol Version 4 (PGPv4) module. PGPv4 is a low-latency, serial 64B66B-based serial protocol for high-speed data transfer over point-to-point link between FPGA/FPGA or FPGA/ASIC communication[21].

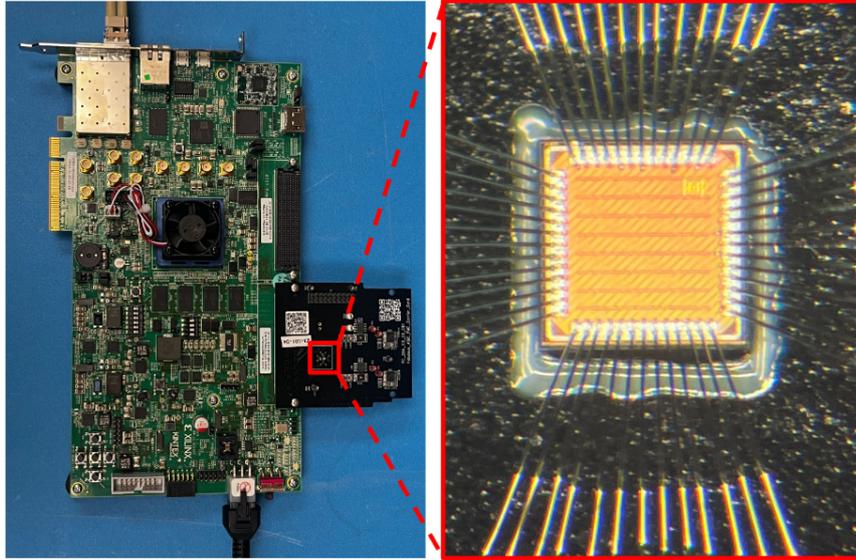


Figure 8. (Left) Photograph of the KCU105 development board with the custom FMC ASIC carrier with ASIC wire bonded to it. (Right) Zoomed in photograph of the 28nm CMOS ASIC (1mm x 1mm).

4.3 Fabrication

The submission of this ASIC design to the TSMC 28nm MPW was completed in July 2023, and the design was received in January 2024. Although the custom PCB carrier for the 28nm ASIC differs from that of the 130nm ASIC (as mentioned in Section 2.3), the majority of the firmware and software was adapted from the previous project with minor modifications to incorporate streaming PGPv4 support, utilizing the same KCU105 development board. A photograph showing the 28nm CMOS ASIC wire-bonded to a custom PCB carrier, and the ASIC on the FMC card alongside the KCU105, can be seen in Figure 8.

4.4 Testing Results

4.4.1 Simple Counter Test

Similar to Section 2.4.1, the basic functionality of the eFPGA was evaluated through the compilation and loading of the bitstream containing very simple 16-bit counter firmware into the eFPGA. A photograph showing the setup for this test can be seen in Figure 9. In the absence of GPIO tiles, 16 bits from the WEST_IO tiles were taped off and driving off chip, which subsequently interfaced with a 16-pin header. This header was then connected to a digital logic analyzer, facilitating the observation of the eFPGA 16-bit counter firmware's behavior. As with the 130nm eFPGA, the behavior of the firmware was confirmed, indicating successful loading of the bitstream.

4.4.2 ASIC Power Draw

As with the 130nm test, the 16-bit counter firmware was used while the clock frequency of the ASIC was varied, and the ASIC's core rail (+0.9VDC) and I/O rail (+1.8VDC) current draws were measured to calculate the total ASIC power drawing. The resulting measurements are shown in

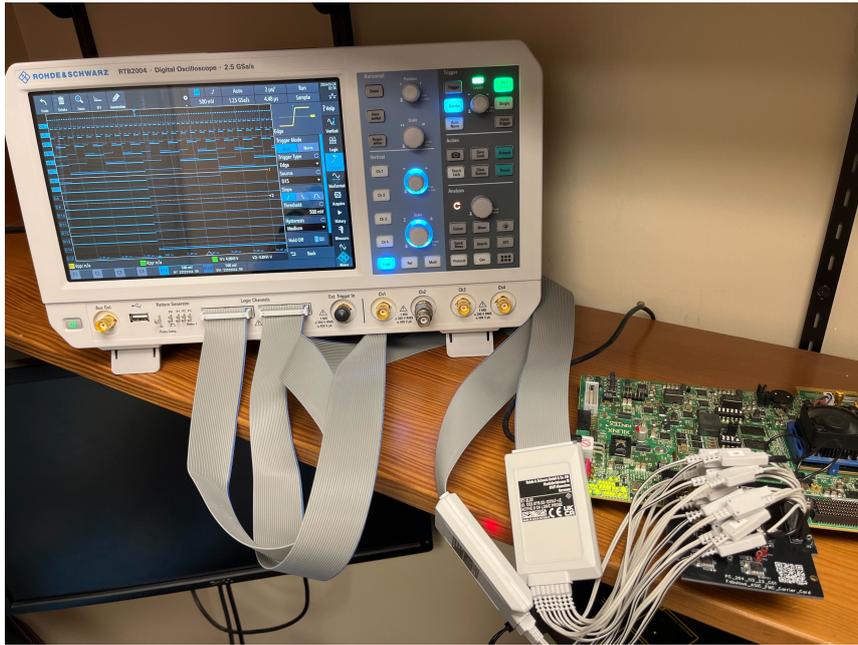


Figure 9. Photograph of the 28nm ASIC connected to a logic analyzer with the eFPGA loaded with a simple 16-bit counter bitstream.

Figure 10, demonstrating that the 28nm ASIC’s core voltage rail power consumption at a 125 MHz clock is approximately one third that of the 130nm ASIC design.

In contrast to the 130nm ASIC (refer to Section 2.4.2), issues with the CMOS output driver slew rate of the 28nm ASIC were not observed and appeared to be comparable to that of the KCU105’s FPGA. A stable SUGOI link lock was achieved on both the ASIC and FPGA sides from 10 MHz to 250 MHz. Although the timing constraints used in the place and route software for the ASIC’s digital logic were set for 200 MHz (5ns clock period), no unusual behavior was noted. Measurements were discontinued beyond 250 MHz due to the FPGA’s inability to achieve timing closure on a combinatorial chain in the PGPv4 protocol, which was related to calculating a 32-bit Cyclic Redundancy Check (CRC) value.

4.4.3 AXI Stream Loopback in the eFPGA

To test the eFPGA’s AXI stream interface, the firmware was developed to enable loopback of the inbound stream into the outbound stream through a single register stage with back pressure handshaking being implemented. For the purposes of this testing, PseudoRandom Binary Sequence (PRBS) frames generated by the software were transmitted to the KCU105’s FPGA, which then forwarded them to the ASIC. With the loopback bitstream having been loaded into the eFPGA, the PRBS frames were sent back (also known as "loopback") to the KCU105’s FPGA, which, in turn, forwarded them back to the software. Upon receiving the frames, the software verified the absence of bit errors in the frames.

This test was conducted using the same clock frequencies steps from Section 4.4.2, ranging from 10 MHz to 250 MHz. At each clock frequency step, the test ran for approximately 10 minutes, during which the software monitored for bit errors in the received frames compared to the frames

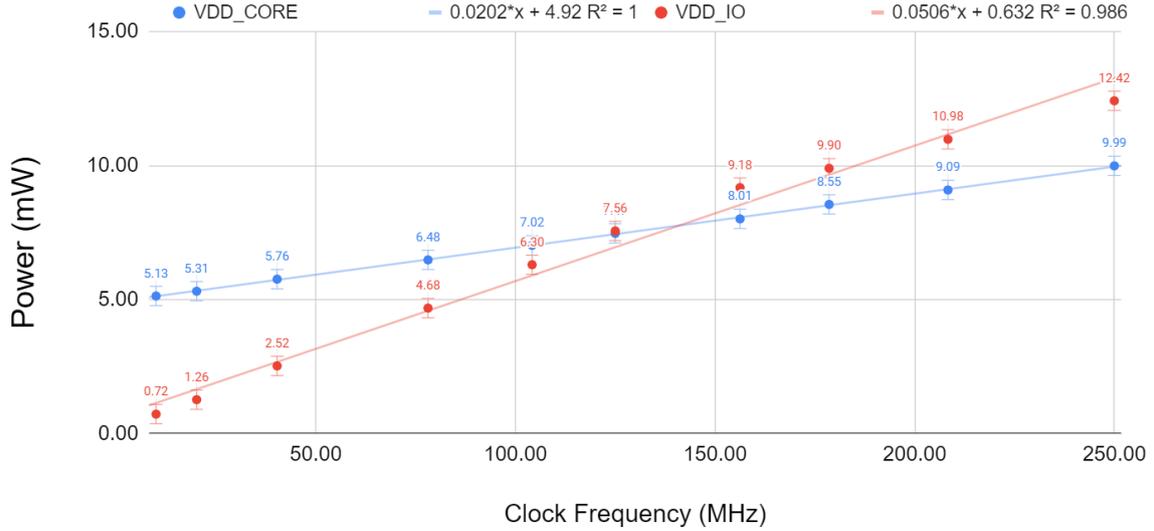


Figure 10. 28nm ASIC power as a function of clock frequency (blue: core voltage supply, red: I/O supply)

it had sent. Throughout these 10-minute intervals, frames were continuously sent by the software, with the software transmission rate being constrained by the ASIC’s PGP link rate, corresponding to the ASIC’s clock frequency (e.g., a 250MHz clock resulted in a 250 Mbps link rate). Throughout all the steps of clock frequency tested, no bit errors were detected.

5 Application for Machine Learning-based At-Source Processing

In light of the successful eFPGA design and fabrication, scientific applications of the technology can be considered. A variety of HEP readout tasks can benefit from reconfigurable logic within an ASIC. Silicon pixel detectors detect the passage of charged particles and are commonly the first layer of high-energy collider detectors after the beam pipe, making them subject to the highest particle fluxes and radiation doses of any subsystem. The new all-silicon ATLAS Inner Tracker (ITk), to be installed in 2026, comprises 1.4 billion pixels with sizes down to $250 \mu\text{m}^2$, resulting in an overall data rate of $O(10)$ Tb/s [22]. However, the large majority of this data rate goes to the recording of “pileup” particles, which arise not from the hard-scatter proton collision, but from soft adjacent collisions that are less likely to contain physics processes of interest. Future detector designs could benefit from the reduction of this rate through filtering or featurization to mitigate the computational power required for trigger decisions and increase signal efficiency. Further benefits of at-source ML-based processing will come to bear for detectors at future 10 TeV parton-center-of-mass “discovery” machines, such as the Future Circular Collider (hh), where data rates are expected to near 1 exabyte per second with unparalleled radiation doses [23].

The “smart pixel” collaboration provides a simulated dataset of particles propagated through a futuristic pixel sensor, which can be utilized to study ML-based readout [24]. This dataset comprises 500,000 fitted tracks originating from high-energy pions collected by the Compact Muon Solenoid (CMS) experiment and propagated through a futuristic pixel detector. This detector features sensors composed of a 21×13 pixel array with a $50 \times 12.5 \mu\text{m}$ pitch, located at a radius of 30 mm from

the beamline within a magnetic field of 3.8 T. Each track is represented as a sequence of eight deposited charge arrays in the (x, y) pixel dimensions, at time intervals of 200 ps. Figure 11 illustrates a diagram of a pixel sensor from this dataset, and an example signal from a charged particle track passing through it. Information about deposited charge in the pixel over time can be used to distinguish high-momentum particles from the hard-scatter proton interaction from low-momentum particles arising from pileup. High-momentum particles are less curved in the magnetic field, therefore traversing fewer pixels compared to pileup particles. It is important to note that the x -profile (sum over pixel columns) runs parallel to the magnetic field and is thus not sensitive to particle momentum. In contrast, the y -profile (sum over pixel rows) is sensitive to the track's incident angle, and thereby its momentum.

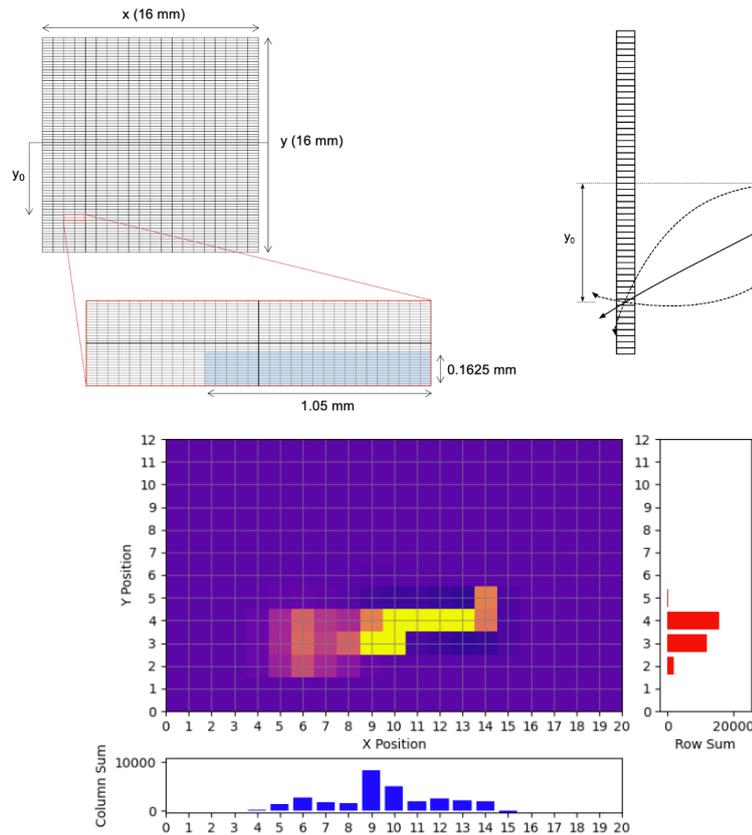


Figure 11. A diagram of the smart pixel sensor (top) [24], and an example smart pixel track represented as deposited charge across the 2D pixel grid (bottom).

Previous work has shown the feasibility of using ML to classify high-momentum tracks from pileup based on the pattern of charge distribution across the sensor over time, enabling real-time data rate reduction by rejecting pileup tracks at the sensor level. This pileup classification model has also been demonstrated to be feasibly integrated into an ASIC design [25]. Another approach for on-chip ML involves performing *regression* on low-level data to determine particle energy and angular information at the front-end, thus reducing the volume of information that needs to be transmitted [26]. Both approaches can significantly reduce the amount of data required to be

transferred off-detector, leading to reductions in cabling and computational power requirements in the trigger system. However, these methods depend on dedicated ASIC designs, which cannot be reconfigured to any other algorithmic operation, suggesting a potential for further advancement with an eFPGA-based approach.

The 28nm eFPGA was used as a proof-of-concept for reconfigurable logic in pixel readout, via the application of small ML-based methods for pileup classification using the smart pixel dataset. The specific task was to output a probability that a particle passing through the has a transverse momentum $p_T < 2 \text{ GeV}$, indicating it is likely to be pileup and thus should not be retained for further offline analysis.

An initial attempt was to design a simple Neural Network (NN) with two or three fully connected layers. Despite utilizing a few nodes per layer, this shallow NN required over 6,000 LUTs, significantly exceeding the capacity of the 28nm eFPGA ASIC. As an alternative, a Boosted Decision Tree (BDT) model was considered, known for its fast training, robust performance, and resource efficiency. Since a BDT primarily relies on comparison operations and thresholds, which can be directly embedded into the LUT logic and FFs, it does not require any block RAM or DSP resources. The BDT models each track as a 1D array of 14 values, 13 of which are the y -profile for each pixel summed over time, and the distance of the pixel from the interaction point y_0 , and outputs a probability that the track has $p_T < 2 \text{ GeV}$. As described in Section 4, the current 28nm eFPGA has a very small logical capacity, with only 448 LUTs. Given these stringent resource constraints, this model consists of a single tree with a depth of 5 and uses gradient boosting with the SCIKIT-LEARN package [27], as shown in Figure 12. These limitations significantly constrain the performance of the model. Before quantization, a background rejection of 4.35% is achieved for a signal efficiency of 97.53%. After synthesis with quantization using `ap_fixed<28,19>`, performance of the model is showed in Table 1.

Signal Efficiency	Background rejection
96.4%	5.8%
97.8%	3.9%
99.6%	1.1%

Table 1. Performance of the synthesized BDT model under different thresholds.

While the performance of this model is limited by the small logical capacity of the hardware, its ability to successfully perform the classification task and fit the resource limitations enabled a full configuration test on the 28nm eFPGA. High-level synthesis of the software algorithm was performed using CONIFER [28], an open-source framework for generating firmware configuration files from software-based BDT algorithms. A full hardware synthesis of the pileup classification BDT was achieved, including threshold values quantization and pruning to accommodate the BDT within stringent resource constraints. This process involved synthesis from C to Verilog firmware and simulation of the hardware response. A single tree was synthesized into a single decision function module in RTL, featuring only 9 threshold parameters and 7 inputs. The entire module utilized 294 LUTs, fitting within the confines of the current 28nm eFPGA design. The compiled bitstream was then loaded into the eFPGA, and a test of the BDT output was performed using the full smart pixel dataset of 500,000 events. A 100% accuracy rate in results was obtained compared

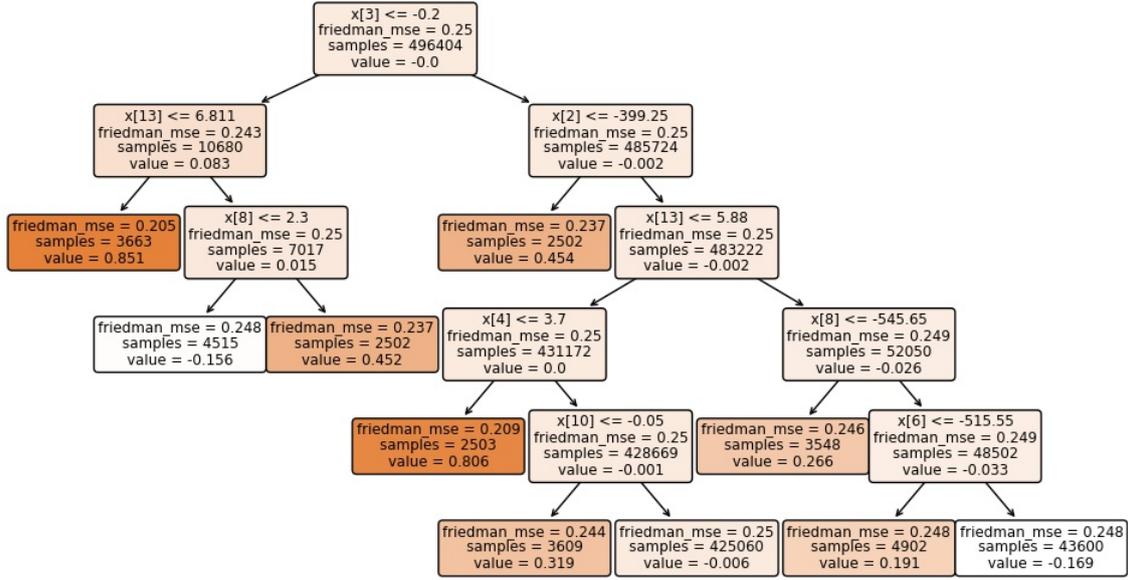


Figure 12. A diagram of the single tree BDT model used for proof-of-concept synthesis to the 28 nm eFPGA.

to the "golden" result of the synthesized model, achieving an operational runtime of less than 25 ns in simulation.

These results provide a proof-of-concept for the use of ML on eFPGAs. However, further development is needed to enable eFPGA technology for readout in collider detectors. The small logical capacities of the ASICs described here are not feasible for the data processing needed for HEP particle detectors, which need to read out $O(10^5)$ pixels with very good signal efficiency and better background rejection than the pileup BDT provides. A next-generation eFPGA with a larger logical capacity could enable the study of higher-performance models and a variety of algorithms to fully exploit the configurability of the ASIC. Further studies into power consumption will also be needed to ensure the eFPGA can be sufficiently low-power for the spatially constrained inner detector environment. Additionally, any readout ASIC in a collider inner system will need to be insensitive to radiation-induced issues such as single-event effects. The implementation of triple modular redundancy (TMR) in FABulous could open up the broad usage of eFPGAs in collider readout scenarios.

6 Summary

This work describes the development of eFPGA technology using the FABulous open-source design framework. Two eFPGAs were designed and fabricated using 130 nm and 28 nm CMOS technologies respectively, and were tested for programmability, power consumption, and performance. The reconfigurability of the eFPGA makes it an excellent candidate for readout of silicon pixel detectors in high energy collider experiments, which can benefit from ML-based filtering or feature extraction to reduce data rates. A proof-of-concept BDT is designed that can classify and reject pile-up tracks ($p_T < 2$ GeV) using a simulated dataset of high-energy pions passing through a generic

pixel sensor. Configuring this BDT model on the eFPGA achieved 100% accuracy compared to the golden results. Future work will focus on development of the FABulous framework, co-design of high-performance sensor processing algorithms, a larger eFPGA design with higher logic density for more complex processing, and dedicated radiation tolerance measures in order to accommodate realistic future collider readout scenarios.

References

- [1] A. McCarn Deiana et al, "Applications and Techniques for Fast Machine Learning in Science", [Front. Big Data 5, 787421](#) (12 April 2022)
- [2] ATLAS Collaboration, "The ATLAS Trigger System for LHC Run 3 and Trigger performance in 2022", [10.48550/arXiv.2401.06630](#) (12 Jan 2024)
- [3] CMS Collaboration, "Development of the CMS detector for the CERN LHC Run 3", [10.48550/arXiv.2309.05466](#) (11 Sept 2023)
- [4] CMS Collaboration, "The Phase-2 Upgrade of the CMS Level-1 Trigger", [CMS-TDR-021](#) (7 Apr 2020)
- [5] E. Govorkova et al, "Autoencoders on FPGAs for real-time, unsupervised new physics detection at 40 MHz at the Large Hadron Collider", [Nature Machine Intelligence 4, 154–161](#) (23 Feb 2022)
- [6] G. Di Guglielmo et al, "A reconfigurable neural network ASIC for detector front-end data compression at the HL-LHC", [IEEE Trans. Nucl. Sci. 68, 2179](#) (7 June 2021)
- [7] S. Kulkarni et al, "On-Sensor Data Filtering using Neuromorphic Computing for High Energy Physics Experiments", [10.48550/arXiv.2307.11242](#) (20 July 2023)
- [8] H. Murayama et al, "Exploring the Quantum Universe: Pathways to Innovation and Discovery in Particle Physics", <https://www.usparticlephysics.org/2023-p5-report/>
- [9] B. Fleming et al, "Basic Research Needs for High Energy Physics Detector Research & Development", [10.2172/1659761](#) (14 Dec 2019)
- [10] A. Apresyan et al, "Detector R&D needs for the next generation $e + e -$ collider", [10.48550/arXiv.2306.13567](#) (26 June 2023)
- [11] Bea Healy, Jing Yu, Nguyen Dao, King Lok Chung, Dirk Koch "FABulous: an Open-Everything Framework for Embedded FPGAs", [Article 15, Workshop on Open-Source EDA Technology \(WOSET\)](#) (2021)
- [12] L. Ruckman, L. Rota, A. Gupta, "SLAC Ultimate Gateway Operational Interface (SUGOI) Protocol for Fiber Optic ASIC/FPGA Communication", [2022 IEEE Nuclear Science Symposium and Medical Imaging Conference \(NSS/MIC\)](#) (9 Nov 2022)
- [13] AMBA AXI and ACE Protocol Specification, ARM ARM IHI 0022E (ID033013)
- [14] ANSI/VITA 57.1-2008. American National Standard for FPGA Mezzanine Card.
- [15] AMD Kintex UltraScale FPGA KCU105 Evaluation Kit, <https://www.xilinx.com/products/boards-and-kits/kcu105.html>
- [16] G. Borghello et al, "Total ionizing dose effects on ring-oscillators and SRAMs in a commercial 28 nm CMOS technology", [JINST 18 C02003](#) (2 Feb 2023)

- [17] S. Bonaldo et al, "Influence of halo implantations on the total ionizing dose response of 28-nm pMOSFETs irradiated to ultrahigh doses", [IEEE Transactions on Nuclear Science 66.1:82-90](#) (Jan 2019)
- [18] Chun-Min Zhang et al, "Characterization and modeling of Gigarad-TID-induced drain leakage current of 28-nm bulk MOSFETs", [IEEE Transactions on Nuclear Science 66.1:38-47](#) (26 Oct 2018)
- [19] Chun-Min Zhang et al, "Characterization of gigarad total ionizing dose and annealing effects on 28-nm bulk MOSFETs", [IEEE Transactions on Nuclear Science 64.10:2639-2647](#) (5 Sept 2017)
- [20] AMBA 4 AXI4-Stream Protocol Specification, ARM IHI 0051A (ID030610)
- [21] B. Reese, L. Ruckman, R. Herbst, "PGP4: A Pretty Good Protocol Version 4 for 10 Gbit FPGA-to-FPGA Communication", [2022 IEEE Nuclear Science Symposium and Medical Imaging Conference \(NSS/MIC\)](#) (7 Nov 2022)
- [22] ATLAS Collaboration, "Technical Design Report for the ATLAS Inner Tracker Pixel Detector", [CERN-LHCC-2017-021](#) (23 Sept 2017)
- [23] M. Benedikt et al, "FCC-hh: The Hadron Collider: Future Circular Collider Conceptual Design Report Volume 3. Future Circular Collider", [The European Physical Journal Special Topics Article, Vol 228, p. 755-1107](#) (5 July 2019)
- [24] Morris Swartz and Jennet Dickinson, "Smart pixel dataset (Version 3)", <https://doi.org/10.5281/zenodo.10783560>
- [25] J. Yoo et al, "Smart pixel sensors: towards on-sensor filtering of pixel clusters with deep learning", [10.48550/arXiv.2310.02474](#) (3 Oct 2023)
- [26] J. Dickinson et al, "Smartpixels: Towards on-sensor inference of charged particle track parameters and uncertainties", [10.48550/arXiv.2312.11676](#) (18 Dec 2023)
- [27] F. Pedregosa et al, "Scikit-learn: Machine Learning in Python", [JMLR 12, pp. 2825-2830](#) (2011)
- [28] S. Summers et al, "Fast inference of Boosted Decision Trees in FPGAs for particle physics", [JINST 15 P05026](#) (29 May 2020)