FPGA-Accelerated Time Series Mining on Low-Power IoT Devices

Seongyoung Kang
Department of Computer Science
Kookmin University
ksy9164@kookmin.ac.kr

Jinyeong Moon
Department of Electrical and Computer Engineering
Florida State University
j.moon@fsu.edu

Sang-Woo Jun
Department of Computer Science
University of California, Irvine
swjun@ics.uci.edu

***Abstract*—We present a case for FPGA-accelerated edge processing for low-power Internet-of-Things (IoT) devices, using time series similarity search as a driving application. As the data collection capabilities of low-power IoT device increase, the primary constraint on their capacity is becoming the resource requirements of wirelessly transferring collected data to a central repository. This work presents a solution to this limitation by augmenting the IoT device with a inexpensive, power-efficient FPGA accelerator, which can perform fairly complex edge mining operations and drastically reduce the wireless data transfer requirements. This approach reduces the total power consumption of the device despite the added FPGA component, while also reducing the computation requirements at the central server. We use the Dynamic Time Warping (DTW) algorithm as an example workload. Using a low-cost Lattice iCE40 UltraPlus FPGA, we demonstrate that the FPGA-augmented mining algorithm can both support significantly higher data collection rate while improving the computation power efficiency of the entire deployment by an order of magnitude.**

***Index Terms*—IoT, FPGA, edge mining, time series**

I. Introduction

The Internet-of-Things (IoT) paradigm augments our physical world with a wide array of sensors, computation, and networking, and is shaping up to become one of the most influential technologies of the modern world. IoT is impacting areas spanning from agriculture [3] and healthcare [14], to manufacturing [11], by enabling extremely pervasive data collection to support deeper analytics and insight.

One of the key components of IoT is the sensor node, which is typically a small, low-power device with an array of sensors, wireless communication, and a modest amount of computation. As these nodes are often deployed at scale in places without significant networking or power infrastructure, such as farms and wilderness, many sensor nodes are designed to be very resource-efficient, in order to autonomously operate for significant amounts of time. For example, they may operate for years on a small battery [9], and communicate via slow, low-power communication networks [7], [13].

While we are seeing continued growth in both data collection capabilities of IoT devices as well as analytics capacity on the collected data, the performance and power efficiency of wireless communications have not scaled as much due to physical constraints. As a result, the data collection capacity of low-power IoT devices is often restricted both by the supported data rate and power consumption of its wireless communication module [2], [7], [9]. Network technologies for IoT devices provide a wide spectrum of choices, spanning from fast and power hungry to slow and power efficient.

One prominent approach to solving the communications issue is *edge mining*, where the IoT nodes themselves perform some computation in order to reduce the amount of data to be transmitted. Many real-world applications have shown that data mining at the edge can reduce the data transmission requirement by over 95% [6]. However, moving computation to the edge also means the IoT nodes must have significant computation capabilities, which in turn increases the cost and power requirements of each node.

In this work, we present a case for edge mining at the edge using Field-Programmable Gate Arrays (FPGA), which can potentially improve the performance and power efficiency of computation by an order of magnitude. We demonstrate this idea on the application of time series similarity search, using the Dynamic Time Warping (DTW) algorithm. We implement the DTW algorithm on a very low-cost and low-power Lattice iCE40 UltraPlus FPGA connected to a microcontroller unit via SPI, and demonstrate efficient filtering at the edge can improve the data collection rate per node by more than 2×, while actually lowering its power budget by 15%. Furthermore, by offloading the computation originally done at a central server to power-efficient FPGAs, this approach also improves the computation power efficiency of the total deployment by an order of magnitude.

While using reconfigurable hardware accelerators such as FPGAs to achieve high performance at the edge is not a new idea [4], [18], we provide an instance of a complex edge mining implementation, as well as in-depth evaluation on performance and power efficiency incorporating wireless communication overhead.

The rest of the paper is organized as follows: We first provide some background and related works in Section II. We then present the design and implementation details of our device in Section III, and provide in-depth evaluation of our design in Section IV. We conclude with future work in Section V.

2160-052X/20/$31.00 ©2020 IEEE
DOI 10.1109/ASAP49362.2020.00015

II. Background And Related Work

A. Dynamic Time Warping

Dynamic Time Warping (DTW) is a distance metric between two time series. Unlike naive euclidean distance, DTW tries to *warp* the time of the two time series to find the closest possible match. As a result, DTW is very effective at finding similar time series with time shifts. Figure 1 (Left) shows a simple example of DTW between two similar time series. Even though the shape of the signal is very similar, euclidean distance would have said they are very different, because of the time shift. Indeed, DTW has been shown to be one of the most effective tools for time series data mining [5], for example to detect patterns in the time series input. There has been some efforts in accelerating DTW with accelerators as well, including server-class FPGAs [15] and GPUs [17].

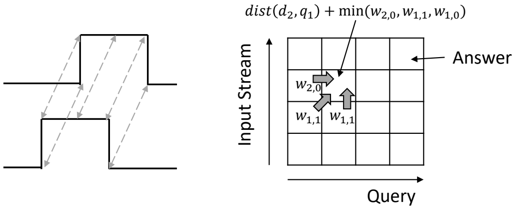

Fig. 1. DTW measures distance between time series better than euclidean distance (Left). Dynamic programming algorithm for DTW (Right)

Figure 1 (Right) describes the dynamic programming algorithm for calculating the DTW distance. The algorithm effectively fills in the cells of a $len(A) \times len(B)$ matrix, based on the distance between the A and B elements at the current location, as well as the minimum of the three temporally previous cells. Since the rows and columns are written once and scanned once before being never accessed again, there are more resource-efficient ways of calculating DTW without keeping the whole matrix in memory.

B. Communication Technologies For IoT

There is a wide spectrum of communication technologies used for machine-to-machine communication in the IoT setting [7], [10]. The various technologies typically differ by the trade-off between bandwidth and cost/power consumption. For example, LTE-M and NB-IoT are cellular communications technologies on top of LTE or GSM under licensed frequency bands. Among the IoT communications technologies, they are relatively fast, reaching hundreds of kbps. However, they are also relatively expensive and power-hungry [9]. On the other hand, Low-Power Wide-Area Network (LPWAN) technologies like LoRaWAN are slow, power-efficient methods which organize around a base station over dozens of kilometers [2]. For example, there is an order of magnitude difference between LoRaWAN and NB-IoT in terms of bandwidth (NB-IoT is faster) and power consumption (LoRaWAN is lower). The power efficiency of LPWAN is desirable, but it sometimes cannot support modern high-data rate IoT deployments.

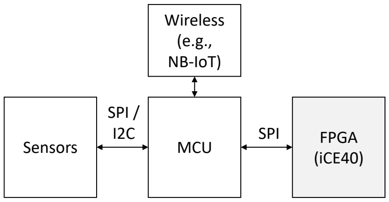

Fig. 2. Overall architecture of an FPGA-accelerated IoT device

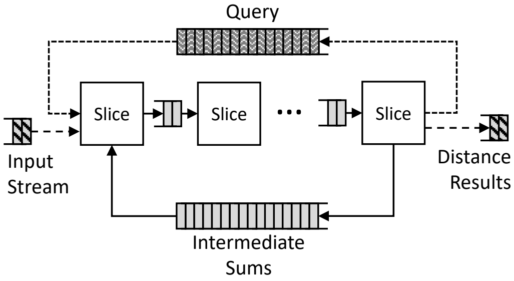

Fig. 3. Architecture of the Dynamic Time Warping accelerator

III. Design And Implementation

A. Overall Design

Figure 2 shows the high-level design of our system. The base system consists of a Microcontroller unit (MCU) such as an Arduino, connected to an array of sensors as well as a wireless communication module such as NB-IoT. We augment this design with a low-cost and power-efficient FPGA such as the Lattice iCE40 FPGA. The MCU communicates with the FPGA via a standard Serial Peripheral Interface (SPI) link, which provides relatively high performance for an embedded peripheral communications link.

B. Dynamic Time Warping Accelerator

In the FPGA, we implement a Dynamic Time Warping accelerator to offload time series mining. Figure 3 dows the DTW accelerator architecture in detail. The accelerator consists of multiple *slices*, each of which computes one intermediate sum per cycle. Each slice is responsible for one element in the input stream, and for a query of length q, it will compute q intermediate sums before moving on to the next element in the input stream. The query data is stored in a block RAM FIFO, and is streamed through the whole chain of slices d/N times, where d is the length of the current input time series, and N is the number of slices. This structure allows the accelerator to work as a filter to the input stream, so the input stream is scanned only once.

The slices are organized into a ring, and each slice forwards each computed intermediate sum to the next slice in the ring. For this ring to make progress, the total number of FIFO slots in the ring must be as large as d. Since there are less slices than the length of the input time series, there is a single d-size block RAM FIFO to hold intermediate sums, between the first and last slices. This structure is memory-efficient, as it only

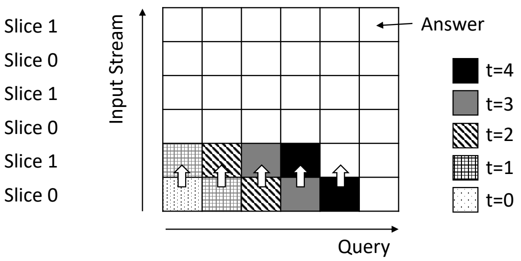

Fig. 4. Dynamic Time Warping computation process with two slices

needs to allocate one d-size queue for the intermediate sums, instead of having to allocate the entire $d \times q$ matrix.

Figure 4 describes the operation of our DTW accelerator, using an example with two slices. Each slice is responsible for one input stream element at a time. Each intermediate sum computed per cycle is sent to the next slice, which will use it at the next cycle. The intermediate sums so far calculated by slice 1 are stored in the block RAM FIFO (e.g., t=0 to t=3 in Figure 4), for slice 0 to load after finishing the current row. This structure keeps all slices busy, consistently computing N elements per cycle.

C. Implementation Details

We have constructed our system using an Arduino Due [1] microcontroller board, and UPDuino [8] FPGA board equipped with a Lattice iCE40 UP5K FPGA. The UPDuino FPGA board is an extremely low-cost device (less than $20 as of 2020), and the Arduino Due is equipped with a capable ARM Cortex-M0 CPU, on top of conveniently providing 3.3V GPIO pins to the UPDuino FPGA. FPGA Development was done with open source Bluespec [12] and Icestorm [16].

Our accelerator design is parameterized, so that we can give data types, maximum query size, as well as slice count as parameters and simply generate the accelerator. We evaluate two configurations: 8-bit inputs with maximum query size of 2048, and 16-bit inputs with maximum query size of 1024. Query sizes were determined by the available block RAM capacity on the UP5K FPGA.

IV. Evaluation

A. FPGA Resource Utilization

The following table describes the FPGA resource utilization for the DTW accelerator. Both designs were clocked at 12 MHz as it was the best clock achievable by dividing the 48 MHz input clock.

Design	Slices	LUTS	BRAM	Clock
8 bit	12	94%	93%	21.52 MHz
16 bit	6	99%	86%	13.49 MHz

B. Performance

Figure 5 presents the performance of the DTW accelerator, in terms of processed time series inputs per second. The 8-bit design was tested on random time series of length 2048, and the 16-bit one with length 1024. In Figure 5, *FPGA* corresponds to the measured performance with the 12 MHz clock, whereas *FPGA-FastClock* corresponds to a hypothetical system clocked at the maximum achievable clock according to the synthesize toolchain. *MCU* shows the relatively slow performance of DTW implemented on the Arduino, which is unable to keep up with high-speed sensors emitting thousands of samples per second, especially while managing the sensors as well. For example, the 8 bit MCU implementation took longer than 1 second to perform DTW on two 2048-length series, despite running on a relatively powerful Cortex CPU.

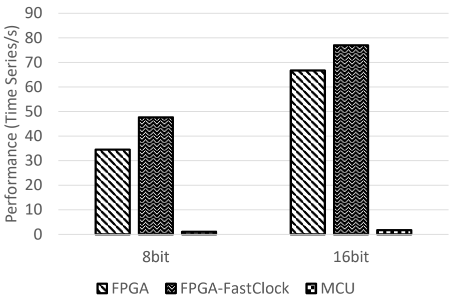

Fig. 5. DTW accelerator performance compared to microcontroller

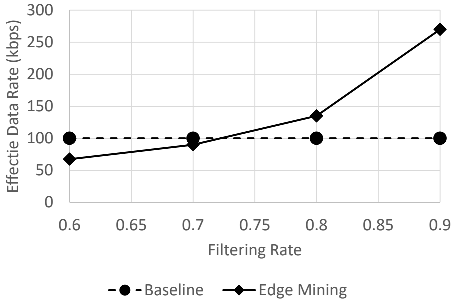

Fig. 6. Effective data rate on a low-power LoRaWAN compared to NB-IoT with different effectiveness of edge mining

Wire-speed time series mining via the FPGA allows more efficient use of low-power wireless communication technologies such as LoRaWAN as an alternative to faster technologies such as NB-IoT. While the peak data rate of LoRaWAN is much slower than NB-IoT, Figure 6 shows that the effective bandwidth of the system increases as the filtering rate improves. Considering previous research on real-world applications have regularly demonstrated more than 95% filtering ratios, the accelerated FPGA approach is expected to achieve more than 2× effective bandwidth compared to the baseline.

C. Power Consumption

The power consumption of the accelerator was measured end-to-end using a USB power monitor plugged into the FPGA board. This way, we measured the power consumption of the entire off-the-shelf unit, instead of estimating the power of only the FPGA chip. During steady-state execution, the FPGA accelerator consumed around 380 mW of power.

Figure 7 shows the power efficiency between the FPGA accelerator running at 12 MHz, FPGA accelerator with fastest

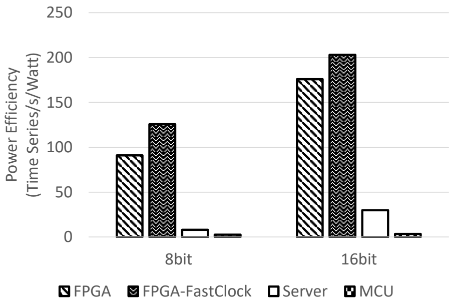

Fig. 7. Power efficiency of DTW computation on different platforms

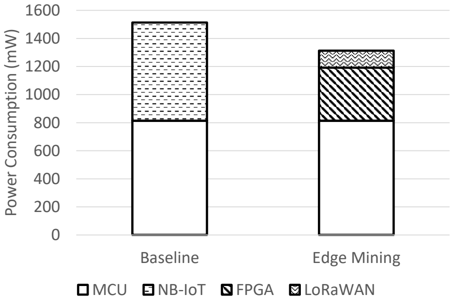

Fig. 8. Total power consumption of an IoT node with and without FPGA-accelerated edge mining

achievable clock, an x86 server, and the ARM Cortex-M0 MCU. The x86 server code was written using C++, and compiled with -O3 -funroll-loops -mavx, and -mavx2. Also, instead of measuring the total power consumption of the server machine, we only measured the difference between the idle state and steady state under load, in order to take the economy of scale into account. Despite the favorable conditions for the x86 software, the FPGA power efficiency is multiple orders of magnitude better than both software solutions.

Figure 8 compares the power consumption of an IoT node. The baseline node uses a high-bandwidth NB-IoT network and no filtering, while *Edge Mining* performs edge filtering using the FPGA accelerator, and transmits data over the slower LoRaWAN network. The power consumption numbers for NB-IoT and LoRaWAN during transmission was taken from existing work [2], [9]. Power consumption of the MCU was measured while receiving data from the host over UART and transmitting it to the FPGA. It can be seen that despite the computation offloading and the subsequently higher effective bandwidth, the edge mining node still consumes 15% less power compared to the baseline.

V. Conclusion

In this work, we have presented an FPGA-accelerated edge mining system using a low-cost, low-power FPGA acceleration, and have shown that it can improve the data collection rate by over 2×, while reducing the node power consumption by 15% and improving the computation power efficiency by multiple orders of magnitude. We plan to continue exploring this direction of exploration, attempting to improve the power efficiency and effective network bandwidth of an IoT node, by offloading computation-intensive filtering operations to an FPGA edge accelerator. We are currently exploring IoT applications with a very strict power budget, for example machine diagnostics powered by power harvesters providing less than 500 mW for the whole system.

We believe FPGA offloading of edge mining can help IoT devices reach both goals of performance and power efficiency, in order to support the ever-increasing data processing requirements of the future.

References

[1] Arduino. Arduino due. https://store.arduino.cc/usa/due.

[2] Lluís Casals, Bernat Mir, Rafael Vidal, and Carles Gomez. Modeling the energy performance of lorawan. *Sensors*, 17(10):2364, 2017.

[3] Danco Davcev, Kosta Mitreski, Stefan Trajkovic, Viktor Nikolovski, and Nikola Koteli. Iot agriculture system based on lorawan. In *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 1–4. IEEE, 2018.

[4] Antonio De La Piedra, An Braeken, and Abdellah Touhafi. Sensor systems based on fpgas and their applications: A survey. *Sensors*, 12(9):12235–12264, 2012.

[5] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[6] Elena I Gaura, James Brusey, Michael Allen, Ross Wilkins, Dan Goldsmith, and Ramona Rednic. Edge mining the internet of things. *IEEE Sensors Journal*, 13(10):3816–3825, 2013.

[7] Sotirios K Goudos, Panagiotis I Dallas, Stella Chatziefthymiou, and Sofoklis Kyriazakos. A survey of iot key enabling and future technologies: 5g, mobile iot, sematic web and applications. *Wireless Personal Communications*, 97(2):1645–1675, 2017.

[8] Gnarly Grey. Upduino_v2_0. https://github.com/gtjennings1/UPDuino_v2_0.

[9] Mads Lauridsen, Rasmus Krigslund, Marek Rohr, and Germán Madueno. An empirical nb-iot power consumption model for battery lifetime estimation. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2018.

[10] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. Overview of cellular lpwan technologies for iot deployment: Sigfox, lorawan, and nb-iot. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 197–202. IEEE, 2018.

[11] D Mourtzis, E Vlachou, and NJPC Milas. Industrial big data as a result of iot adoption in manufacturing. *Procedia cirp*, 55:290–295, 2016.

[12] Rishiyur Nikhil. Bluespec system verilog: efficient, correct rtl from high level specifications. In *Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2004. MEMOCODE'04.*, pages 69–70. IEEE, 2004.

[13] Nicolas Sornin, Miguel Luis, Thomas Eirich, Thorsten Kramp, and Olivier Hersent. Lorawan specification. *LoRa alliance*, 2015.

[14] Arijit Ukil, Soma Bandyoapdhyay, Chetanya Puri, and Arpan Pal. Iot healthcare analytics: The importance of anomaly detection. In *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, pages 994–997. IEEE, 2016.

[15] Zilong Wang, Sitao Huang, Lanjun Wang, Hao Li, Yu Wang, and Huazhong Yang. Accelerating subsequence similarity search based on dynamic time warping distance with fpga. In *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, pages 53–62, 2013.

[16] Clifford Wolf and Mathias Lasser. Project icestorm. http://www.clifford.at/icestorm/.

[17] Yaodong Zhang, Kiarash Adl, and James Glass. Fast spoken query detection using lower-bound dynamic time warping on graphical processing units. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5173–5176. IEEE, 2012.

[18] Chao Hu Zhiyong, Liu Yingzi Pan, Zhenxing Zeng, and Max Q-H Meng. A novel fpga-based wireless vision sensor node. In *2009 IEEE International Conference on Automation and Logistics*, pages 841–846. IEEE, 2009.