Accelerating Solar Spectra Analysis:A High-Performance FPGA Framework with Parameterized Filters

Verjina Torosian Khouygani[1], Shahnam Mirzaei[2], Ph.D., *Member, IEEE*
Christian Beck[3] Ph.D., Debi Prasad Choudhary[4] Ph.D.
California State University, Northridge CA 91330, USA, National Science Observatory, Boulder, CO 80303, USA
[1]verjina.torosiankhouygani.925@my.csun.edu,[2]smirzaei@csun.edu, [3]cbeck@nso.edu,
[4]debiprasad.choudhary@csun.edu

Abstract— Astronomical objects like the Sun are studied through remote sensing, allowing detailed analysis of atmospheric properties from high-resolution spectra. A typical 1-hour solar observation results in a 4D data cube with 700 million spectra, requiring individual analysis. Processing this data on a standard quad-core desktop takes about 250 days with current inversion codes. In this paper, we present a hardware-accelerated framework for solar spectral analysis, featuring a parameterized matched filter IP core. Our system processes a 3D data cube of 700,000 spectra, matching it to an archive of 240,000 spectra in under 3 minutes. Operating at 90 MHz, it processes each spectrum in 124 µs, achieving a 241x speedup over software-based analysis and 10x faster than a GTX 1050 Ti GPU. Implemented on a Zynq UltraScale+ MPSoC, our framework uses the SSD method for spectral matching and can be scaled for higher accuracy and throughput. The IP core is highly configurable for various parameters, such as accuracy, wavelength points, performance, and speedup. This work represents the first hardware-accelerated system for solar spectral matching, truly accelerating the software algorithm on hardware, providing a significant improvement in performance over conventional approaches.

Keywords—ZYNQ, SoC, Match Filter, FPGA, Solar Spectra Matching, High Speed Processing, Hardware Acceleration, Inversion, Data cube

I. Introduction

Astronomical objects such as stars or the Sun can only be studied by remote sensing as in situ measurements are rarely possible [1]. The primary tool is the detailed analysis of the electromagnetic radiation that these objects emit. Stars are usually spatially unresolved and thus only provide disc-integrated radiation, while the Sun can be studied at high spatial resolution down to a few ten kilometers on its surface thanks to its relative proximity [2, 3]. For stars observations in broad wavelength ranges of several ten nm suffice to determine characteristic properties such as the average surface temperature. However, the largest amount of information in the electromagnetic spectrum is contained in the shape of absorption or emission lines from different chemical elements that form in the solar or stellar atmospheres and that contain height-dependent information on the atmospheric properties [4]. Spectral imaging is a technology that integrates conventional imaging and spectroscopy to get both spatial and spectral information from an object. Although this technology was originally developed for remote sensing, it has been extended to the biomedical engineering field as a powerful analytical tool for biological and biomedical research [5].

This framework can be scaled up using faster and higher density memories and FPGA devices to enable researchers to design and build a hardware accelerated computer system to analyze solar spectral data (or any other spectral data) in considerably less time and with less upfront effort than it would take using existing software tools alone. The following is the contribution of our work:

a. Develop an FPGA framework to accelerate solar spectral data analysis.
b. Implement efficient data transfer from memory and processing system to the spectral matching accelerator.
c. Design a parameterized filter core for configurable trade-offs in accuracy, performance, and speedup.

This paper presents a configurable matched filter IP core that finds the best-fit spectral data cube from an archive using the sum of squared differences (SSD) method. The IP core, designed for solar physics inversion codes, allows parameter tuning to meet user requirements. Implemented on the Xilinx Zynq UltraScale+ MPSoC, it combines ARM-based processing with programmable logic, leveraging both memory space and hardware parallelism. This is the first hardware implementation of direct solar spectra matching on reconfigurable hardware.

The paper is organized as follows: Section II covers background and related work. Section III explains spectral data extraction from the data cube. Section IV details the hardware accelerator framework on the Zynq UltraScale+ MPSoC. Section V presents experimental results and analyzes the hardware implementation. Section VI concludes with key remarks and future work.

II. Background And Related Work

The inversion process separately analyses each and every observed spectrum I(x, y, λ), where x, y indicate the spatial location and λ indicates the wavelength (see Figure 1). From the comparison to the pre-calculated spectral archive, a single best matching spectrum is found that minimizes the sum given by Equation (II-1):

$$\epsilon = \sum_{i=0}^{n-1} (I_{OBS}(x, y, \lambda_i) - I_{ARCHIVE}(x, y, \lambda_i))^2 \qquad (II\text{-}1)$$

Where $I_{OBS}$is the intensity of the observed spectral data and $I_{ARCHIVE}$ is the intensity of the archive spectral data. Each archive spectrum is uniquely coupled to one specific temperature stratification in the optical depth scale T(log τ) or in geometrical height T(z). By finding the best matching spectrum for each observed spectrum in a single spectral scan, one thus obtains a temperature cube T(x, y, z) in the solar atmosphere at one moment of time. Running the inversion over multiple spectral scans acquired at different times t_i then provides a four dimensional temperature cube T(x, y, z, t) that traces the evolution of the solar atmosphere in space and time. The information on the thermal properties of the solar atmosphere can then be used to determine the two-dimensional (2D) or three-dimensional (3D) structure of solar structures such as magnetic elements [6], hot chromospheric loops [7] or flow channels [8]. An analysis of time series of thermal cubes can be used to trace wave propagation [9] or energy deposit in the upper solar atmosphere [10, 11]. For the solar case, high spectral resolution is possible due to abundant light. Capturing the Sun's rapid atmospheric changes also requires a cadence of a few seconds. Combined with high spatial resolution, this results in four-dimensional data cubes: spatial (x, y), spectral (λ), and temporal (t), as shown in Figure 1.

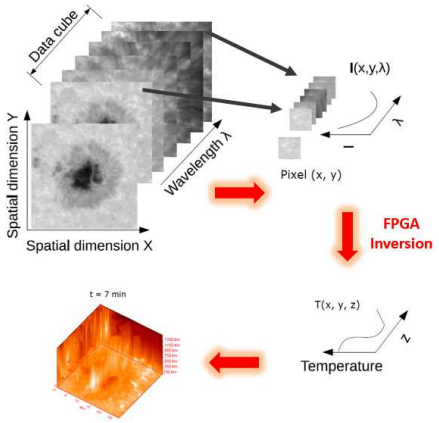

Fig. 1. Inversion process for a spectral scan. The data cube I(x, y, λ) (top left) is used to extract an individual spectrum I(x, y, λ) (top right). The FPGA finds the best-matching spectrum from the archive, yielding the temperature stratification T(x, y, z) (bottom right). After processing all pixels, a temperature cube T(x, y, z) is generated (bottom left). The 3D rendering shows maximum temperature along each ray, with sunspots appearing as dark patterns and bright regions indicating higher temperatures.

For modern-day imaging spectrometers such as the Interferometric BIdimensional Spectrometer (IBIS) [12] with a 1k × 1k chip at a 7Hz frame rate and a 5 s cadence for a spectral scan, a typical data cube from a 1-hr observation contains a total of 1000 × 1000 × 720 = 720,000,000 spectra. To understand the structure and the temporal evolution of the solar atmosphere in detail, each and every spectrum must be examined. However, converting observed spectral intensities to atmospheric properties is complex due to contributions from different height layers in the atmosphere. Typically, the "inverse" problem is solved by adjusting a model atmosphere to match the observed spectrum using an iterative gradient method. The model atmosphere is iteratively modified by a gradient method that minimizes the least-squares difference between the observed and the synthetic spectrum [13]. This process, requiring multiple radiative transfer integrations, makes inversion codes slow, often taking several seconds per spectrum. As a result, analyzing data cubes with millions of spectra becomes computationally unfeasible. One solution is to use a pre-calculated archive of spectra for various atmospheric stratifications, avoiding repeated radiative transfer calculations [14, 15]. The inversion process then involves finding the closest match in the archive. While this approach has limitations, such as a fixed set of solutions, it significantly speeds up analysis, reducing processing time to milliseconds per spectrum on a standard desktop [15]. However, even with 30 ms per spectrum, analyzing a data cube with 720 million spectra would still take about 250 days of computation for 1 hour of observations.

To the best of our knowledge, the only hardware-based solution to solar spectral matching/inversion problem is implemented by Carrascosa et al. [16. 17]. An FPGA based system was developed to implement radiative transfer equation (RTE) inversion. Authors have implemented a multiple instruction multiple data (MIMD) processor system on Virtex 5 FPGA with the utilization of 21745 LUTs, 28399 FFs, 212 BRAMs, and 136 DSP blocks running at 200 MHz. It is claimed that the system can achieve the speed up of 41 compared to the software-based solution running on standard personal computers (PCs). However, this may not be compared to our work as the framework proposed by this paper implements a true acceleration of the inversion software algorithms. There has not been any custom hardware developed for the inversion process in [16].

III. Understanding Solar Spectral Data

The observed data cube is the solar chromospheric spectral line of Ca II IR at 854 nm, with 27 wavelength points and a 1000×1000 pixel field of view [2]. Two data sets are prepared: the first is a 3D data cube rearranged into a 2D array of 27×1,000,000; the second is the archive data, containing 239,400 spectra of 27 wavelength points in a 2D array of 27×239,400. Both arrays are converted to binary format using MATLAB and transferred to an SD memory card. Figure 2 illustrates a conceptual data cube with 2D images at wavelengths 1, 8, and 15. For the pixel at coordinates (400, 300), the observed spectral data is shown in the bottom right of Figure 2, where the horizontal axis represents wavelength and the vertical axis represents the normalized intensity at that pixel.

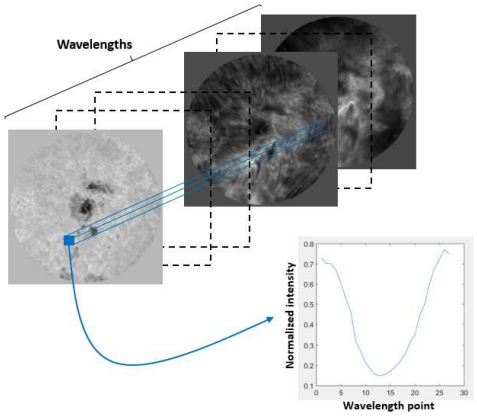

Fig. 2. Observed solar spectra at coordinates (400, 300) with 27 different wavelength points for image frame sizes of 1000×1000 pixels

IV. Solar Spectra Analysis Framework

A. System Architecture Overview

The components in the accelerator framework include: a software application that runs on an ARM cortex R5F processor, off-chip memory to store pixel and archive data, system interconnect, and the programmable logic (PL). The PL mainly consists of a parameterized matched filter IP core, a block memory (BRAM) controller and a BRAM storage unit. Figure 3 shows the top-level block diagram of the system architecture.

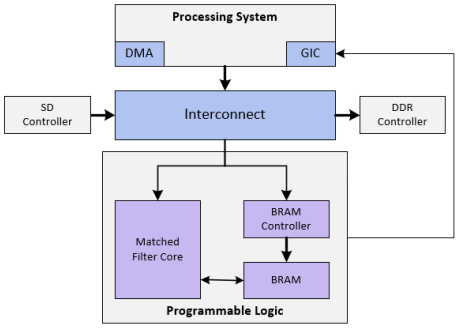

Fig. 3. Solar spectra accelerator system architecture

Solar spectral data are stored on an off-chip SD card and read via the FAT system. The data are transferred to DDR memory and loaded into PL BRAMs using DMA, which allows large data transfers without processor involvement. Interrupts are managed by the ARM generic interrupt controller (GIC), mapping all interrupts to the processing system (PS)'s IRQ input. The processing system provides the PL with requested observed or archive data. Since the data size exceeds BRAM capacity, it is divided into smaller chunks, transferred from DRAM to PL BRAM via DMA. The chunk size is configurable within the IP core. Figure 4 shows the software application functional diagram.

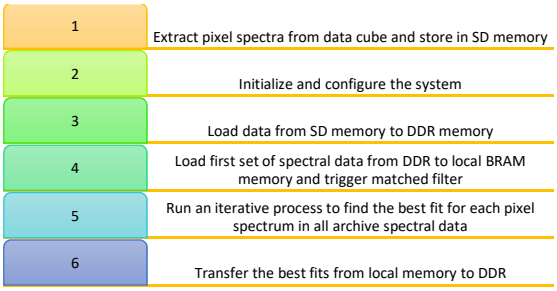

Fig. 4. Processor system functional diagram

After initialization, the first set of pixel and archive spectra are loaded into BRAMs. The PS triggers the matched filter IP core to find the best-fit archive spectrum for each pixel. An interrupt loads the next set of archive spectra, repeating until all spectra are compared. The PS reads the results, saves them to DDR, and sends the next set of pixel spectra. This continues until all pixels are matched, and the final interrupt transfers results to DDR and stops the PL. The bottom of Figure 3 shows the PL setup, where an AXI smart connector links three slave IPs in the PL to the PS via a high-performance AXI interface. Section B discusses the matched filter IP core architecture.

B. Configurable Matched Filter IP Core Architecture

The core of our framework is a configurable accelerated matched filter IP core. This parameterized module allows configuration of spectrum data width, wavelength points, BRAM usage, and depth. For a given data cube and archive, a module with 25 BRAMs per group is instantiated, as detailed in Sections IV.B.1 and IV.B.2. A decoder module ensures pixel and archive data are written sequentially into the appropriate BRAMs. Data is transferred via the high-performance AXI interface with 32-bit width, and pixel and archive data streams are read as outlined in Figure 5.

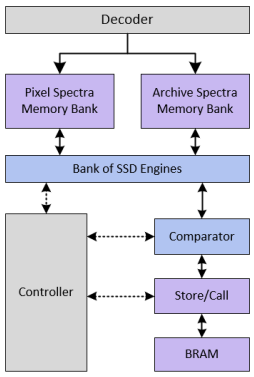

Fig. 5. Parameterized matched filter IP core block diagram

B.1 Matched Filter IP Core Memory Banks

Figure 6 shows the addressing of BRAMs for pixel and archive spectra. These dual-port BRAMs have port A as write-only and port B as read-only. For the PS, they form a memory space of 50 BRAMs connected in a cascaded arrangement. The SSD module and BRAM controller access these BRAMs through a decoder. Dual-port BRAMs eliminate the need for extra decoders and MUXs, enabling simultaneous read and write operations.

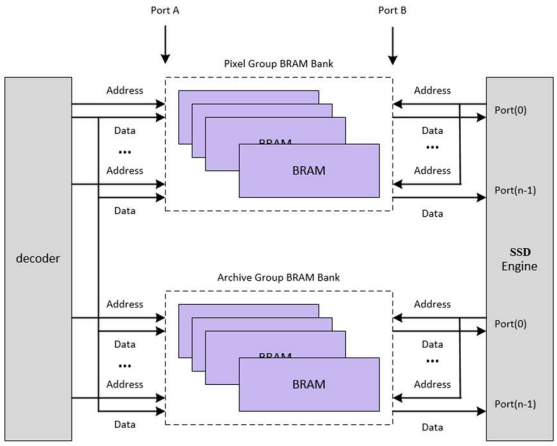

Fig. 6. Parameterized matched filter IP core memory bank block diagram. The memory bank in this work is configured for N = 25

B.2 Bank of SSD Engines

The SSD method has been used as a measure of similarity between two spectra according to equation (*IV-1*).

$$SSD = \sum_{i=0}^{n-1}(a_i - b_i)^2 \qquad (IV\text{-}1)$$

Where n is the number of wavelength data points and a_i and b_i are intensity values at corresponding points of observed and archive spectral data respectively. For each pair of observed and archive spectra data, the SSD is calculated and the minimum value will be selected as the best fit within each set of data that is loaded into the BRAMs. On the other hand, since each group has a shared address bus on their port B, any address provided with the SSD module, reads the data from the same location of all BRAMs in its related group. Assuming the number of wavelength data points is represented by W_p, in every $W_p+2 = 29$ cycles, the SSD module reads 25 pixel spectra (pixel spectrum group) and 25 archive spectra (archive spectrum group) from these two groups of BRAMs. Then it calculates the sum of squared differences between values of each spectrum from the pixel spectrum group and of all spectra from the archive spectrum group. The result is a two-dimensional 25×25 array of SSDs where each row contains the results for one pixel as shown in Figure 7. By utilizing this method, 625 SSD operations are done in $W_p + 2 = 29$ cycles. Reading values of each spectrum in a serial manner for the SSD calculation (for total time of $W_p + 2$ cycles) allows the parameterization of the number of wavelength point values in the spectrum so the design can apply to any spectral data cube. The SSD module runs in parallel with comparator and Store/Call modules. While the SSD calculates the sum of squared differences between all 25-pixel spectra and all 25 archive spectra (total of 625 SSDs), comparator processes the result of the SSD module done in the previous round (each round takes $W_p + 2$ cycles). For each pixel spectrum, the comparator compares SSDs between the pixel spectrum and all 25 archive spectra, as well as with the minimum SSD from the previous round. To do this, the Store/Call module retrieves the best-fit spectrum indices and SSDs from dedicated BRAMs in advance (Figure 5). The comparator stores its results in these BRAMs after processing each group of pixels. Designed to minimize resource usage, the comparator performs comparisons sequentially within its Wp + 2 cycles. Control signals between the matched filter IP core and the PL are transferred via the interrupt IP with an embedded AXI-Lite interface. For our design, the number of SSD engines was chosen to be 625 to fully utilize FPGA resources. Each SSD engine needs 4 DSP blocks to multiply two 27-bit numbers. The total number of DSP blocks available on XCZU9EG is 2520. Also, the number of BRAMs is related to the number of SSDs based on the Equation (IV-2) according to Figure 7. This gives 25 BRAMs for every group. The size of pixel spectra and archive data are still much larger than the total sizes of BRAMs. For this reason, data are divided into sets. The detail of this is presented in Section V.B.

$$Number\ of\ SSD\ operations = Number\ of\ BRAMs^2 \quad (IV\text{-}2)$$

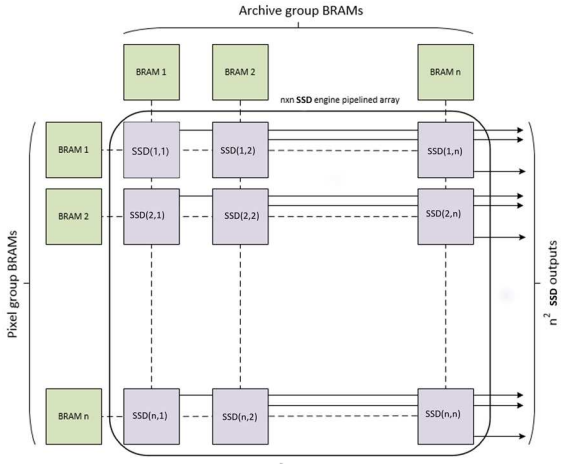

Fig. 7. Parameterized N×N SSD engine pipelined array embedded in matched filter IP core. The SSD bank in this work is configured for N = 25 for total of 625 SSD engines.

V. Experimental Results And Analysis

We implemented our hardware accelerated framework for solar spectral matching on Zynq UltraScale+ XCZU9EG running at 90 MHz. In the following we will discuss the implementation details.

A. FPGA Implementation Results

Table 1 shows the area utilization of the complete hardware accelerated system on Zynq UltraScale+XCZU9EG equivalent to the system architecture shown in Figure 3. Our implemented hardware accelerated framework can process a three-dimensional data cube of 700,000 spectra to find the best fit to the observation from an archive of 240,000 pre-calculated spectra. We have measured the number of FPGA resources for two different parameters: wavelength data point data width (8, 14, 20, and 27), and the number of BRAMs corresponding to the number of SSD operations (10, 15, 20, and 25). Results of these experiments are illustrated in Figure 8 when implemented on a Zynq MPSOC for these parameters.

Table 1: Solar spectral matching FPGA resource utilization on XCZU9EG

Subsystem	LUT	FF	BRAM (18 Kb)	DSP
Interrupt Controller	55	170	0	0
Filter IP Core	137562	118629	728	2507
BRAM Controller	296	288	32	0
AXI Interconnect	7864	8850	0	0
Total	145787	127971	760	2507
Available	274080	548160	912	2520
Utilization Ratio	49.97%	21.59%	79.82%	99.48%

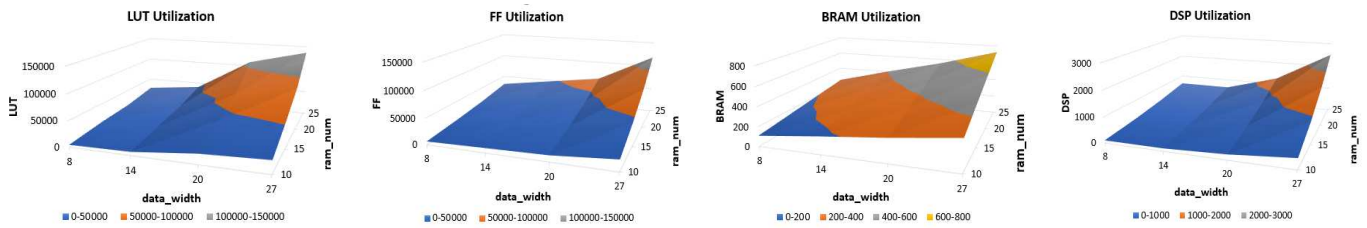

Fig. 8. Device utilization of parameterized filter IP core for different parameters: LUTs (Left), FFs (Middle Left), BRAMs (Middle Right), DSP blocks (Right)

B. Effect of matched Filter IP Core Parameters on Implementation Results

The matched filter IP core is a customizable, parameterized module for various applications. It can be adjusted for the number of wavelength points, data width, memory depth, and SSD operations. More wavelength points improve accuracy but increase latency, while wider data increases resolution and accuracy. Increasing SSD operations boosts speed, and greater memory depth processes more spectra per round, reducing processing time. More wavelength points improve inversion accuracy but increase latency. Increasing data width enhances resolution and accuracy. The number of SSD operations has the greatest impact on speed, while greater memory depth processes more spectra per round, reducing profile processing time and improving performance. Memory depth is limited by FPGA resources, so careful selection of this parameter optimizes system performance. Increasing wavelength points has minimal impact on FPGA area usage, as shown in Table 2, which compares performance with 27 to 128 wavelength points, 27-bit data width, and 10 BRAMs per group.

Table 2: Effect of wavelength points on FPGA area utilization

Points	LUT	FF	BRAM (36Kb)	DSP
27	24770	20917	323	404
64	24764	21396	324	404
128	24796	21594	325	404

Assume N_p is the number of pixel spectra, N_l is the number of archive spectra, W_p is is the number of wavelength points in one spectrum, B_p and B_l are the number of spectra in BRAMs (total size of BRAMs divided by the number of wavelength points W_p) for one set of pixel spectra and one set of archive spectra respectively, T_{clk} is the PL clock cycle time, and finally K is the number of BRAM instances for one set of pixel or archive spectra in the PL, then the number of pixel spectrum sets is $pset = N_p/B_p$, the number of archive spectrum sets is $lset = N_l/B_l$, the number of groups (group refers to spectra located at the same addresses in BRAMs) in one set of pixel spectra is $cntp = B_p/K$ and the number of groups in one archive spectra set is $cntl = B_l/K$. Since for each set of pixel the whole archive should be transferred to the PL, the estimated time spent on data transfer is $(B_p + N_l) \times W_p \times T_{clk}$ and the time for all pixel sets is given by Equation (*V-1*):

$$Total\ process\ time\ in\ PS = pset \times (B_p + N_l) \times W_p \times T_{clk} = (N_p + N_p/B_p \times N_l) \times W_p \times T_{clk} \qquad (V\text{-}1)$$

The time that the PL takes to process one set of pixel spectra with one set of archive spectra is $cntl \times cntp \times (W_p + 2) \times t$. So the estimated time spent in the PL for the whole process can be written using Equation (*V-2*):

$$Total\ process\ time\ in\ PL = cntl \times cntp \times (W_p + 2) \times T_{clk} \times lset \times pset = (W_p + 2) \times T_{clk} \times N_l/K \times N_p/K \qquad (V\text{-}2)$$

In these two equations the factors that can be changed are K and B_p. Since k^2 indicates the number of squared differences calculated in $(W_P + 2)$ cycles, it is strictly limited by the number of available resources.

C. Performance and Spectral Data Analysis

The ultimate goal in this work is to speed up the solar spectral matching process for a data cube. As explained before, the SSD was used as a measure of similarity between two spectra. In this design, 625 SSDs are being calculated every 29 cycles (for total of 1000,000 × 239,400 SSDs). Figure 9 shows the comparison of six random pixel spectra against their corresponding best fits found by our accelerator.

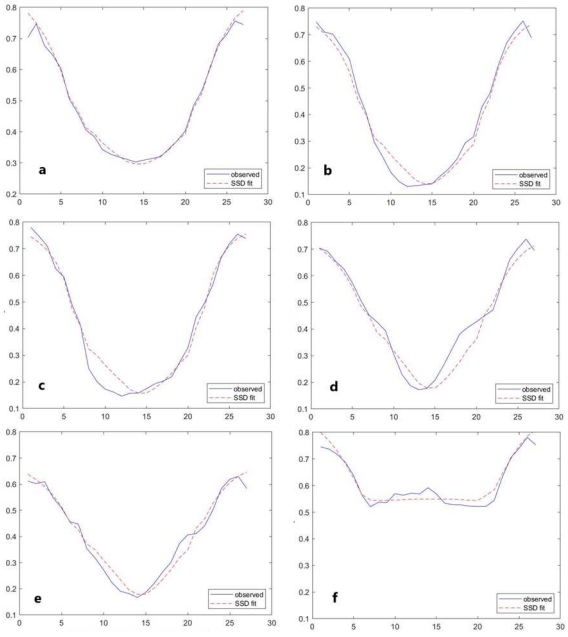

Fig. 9. Random samples of matched pair spectral data, horizontal axis: wavelength points, vertical axis: normalized intensity, (a) through (f): Blue curves (solid lines): spectra of random pixel data, Red curves (dashed lines): best fit spectra from archive

Our proposed hardware accelerated framework is able to process a three-dimensional data cube of 700,000 spectra to find the best fit to the Sun's spectral data from an archive of 240,000 pre-calculated spectra in less than 3 min. Our hardware accelerated system has an average processing time of 0.124 ms per spectrum with no timing constraints at 90 MHz as opposed to 30 ms running on a standard quad core desktop machine. This gives the speed up of almost 241.

Compute Unified Device Architecture (CUDA) [18] is a parallel computing framework and programming model developed by NVIDIA [18] for general computing on graphical processing units (GPUs). We developed CUDA code for solar spectra matching and ran it on a GTX 1050 Ti GPU with 768 CUDA cores [20], achieving 5 ms per profile. Assuming each CUDA core is equivalent to one processing unit [21], it can be compared to a DSP block on the FPGA accelerator. To run the code on other GPUs, details like shared memory, multiprocessor count, and CUDA cores per multiprocessor are needed, which is beyond this work. Comparing the GPU to our hardware accelerator, each CUDA core roughly matches one DSP block on the FPGA. This gives a speedup of 10 for our hardware accelerator, as the GPU's execution time per profile (5 ms) is roughly equivalent to 1.25 ms on the FPGA, assuming each SSD uses 4 DSP blocks.

VI. Conclusion And Future Work

The paper presents a hardware-accelerated framework for solar spectral matching using a scalable, parameterized matched filter IP core. This system efficiently matches a 3D data cube of 700,000 spectra against an archive of 240,000 precomputed spectra in under 3 minutes. With an average processing time of 124 microseconds per spectrum at 90 MHz, the framework is approximately 241 times faster than a software-based solution on a quad-core CPU and about 10 times faster than a GTX 1050 Ti GPU implementation. It is adaptable for larger datasets, higher accuracy, and general spectral analysis.

Our work can serve as a foundation for larger data sets for scientific applications. In more realistic scientific applications, multiple spectral scans can be processed to increase the system throughput. This will be beneficial where input spectra would no longer be the 700,000 spectra from a single spectral scan, but hundreds of millions from a time series of observations. These scans can be processed sequentially, randomly, combined in batches, or in parallel across multiple FPGA frameworks. An interesting application of this work is FPGA-based inversion on satellites with low data bandwidth, allowing spectral analysis onboard and transferring only the results. The temperature stratification can be retrieved using the best-fit number from the archive, without needing the temperature archive onboard. Space-qualified FPGAs can handle harsh environments, as demonstrated by the Solar Orbiter mission [22, 23], which performs onboard inversion due to the low data rate at its distance from Earth. Our framework offers scalable computing capacity at a lower cost compared to full-scale cores, thanks to the reprogrammable and configurable capabilities of FPGAs.

References

[1] Fox, N. J., Velli, M. C., Bale, S. D., et al., "The Solar Probe Plus Mission: Humanity's First Visit to Our Star", 2016, Space Sci. Rev., 204, 7

[2] Elmore, D.F., Rimmele, T., Casini, R., Hegwer, S., Kuhn, J., Lin, H., McMullin, J.P., Reardon, K., Schmidt, W., Tritschler, A. and Wöger, F., 2014, July. The Daniel K. Inouye Solar Telescope first light instruments and critical science plan. In Ground-based and Airborne Instrumentation for Astronomy V (Vol. 9147, pp. 47-53). SPIE.

[3] Scharmer, G.B., Löfdahl, M.G., Sliepen, G. and de la Cruz Rodriguez, J., 2019. Is the sky the limit?-Performance of the revamped Swedish 1-m Solar Telescope and its blue-and red-beam reimaging systems. Astronomy & Astrophysics, 626, p.A55.

[4] Rutten, R. J. 2003, Radiative Transfer in Stellar Atmospheres, Lecture Notes Utrecht University

[5] Q. Li, X. He, Y. Wang, H. Liu, D. Xu, F. Guo, 2013, Journal of Biomedical Optics 18(10), 100901, "Review of spectral imaging technology in biomedical engineering: achievements and challenges

[6] Beck, C., Rezaei, R. and Puschmann, K.G., 2013. Can spicules be detected at disc centre in broad-band Ca ii H filter imaging data?. Astronomy & Astrophysics, 556, p.A127.

[7] Beck, C., Choudhary, D. P., & Rezaei, "A three-dimensional view of the thermal structure in a super-penumbral canopy", R. 2014, The Astrophysical Journal, 788, 183

[8] Choudhary, D. P. & Beck, C., "Thermodynamic Properties of the Inverse Evershed Flow at Its Downflow Points", 2018, The Astrophysical Journal, 859, 139

[9] Beck, C., Rezaei, R. and Puschmann, K.G., 2013. The energy of waves in the photosphere and lower chromosphere-IV. Inversion results of Ca II H spectra. Astronomy & Astrophysics, 553, p.A73.

[10] Rezaei, R. and Beck, C., 2015. Multiwavelength spectropolarimetric observations of an Ellerman bomb. Astronomy & Astrophysics, 582, p.A104.

[11] Grant, S.D., Jess, D.B., Zaqarashvili, T.V., Beck, C., Socas-Navarro, H., Aschwanden, M.J., Keys, P.H., Christian, D.J., Houston, S.J. and Hewitt, R.L., 2018. Alfvén wave dissipation in the solar chromosphere. Nature Physics, 14(5), pp.480-483.

[12] Gary, D.E., 2023. New insights from imaging spectroscopy of solar radio emission. Annual Review of Astronomy and Astrophysics, 61(1), pp.427-472.

[13] del Toro Iniesta, J.C. and Ruiz Cobo, B., 2016. Inversion of the radiative transfer equation for polarized light. Living Reviews in Solar Physics, 13(1), p.4.

[14] Beck, C., Rezaei, R. and Puschmann, K.G., 2013. The energy of waves in the photosphere and lower chromosphere-IV. Inversion results of Ca II H spectra. Astronomy & Astrophysics, 553, p.A73.

[15] Beck, C., Gosain, S., Kiessner, C., "Fast Inversion of Solar CaIISpectra in Non-local Thermodynamic Equilibrium", 2019, The Astrophysical Journal, Volume 878, Number 1, 878, 60

[16] J.P. Cobos Carrascosa, B. Aparicio Del Moral, J.L. Ramos, A.C. López Jiménez, J.C. Del Toro Iniesta, 2014 IEEE 8th International Symposium on Embedded Multicore/Manycore SoCs, "A Multicore Architecture for High-Performance Scientific Computing Using FPGAs"

[17] Cobos Carrascosa, J.P., 2016. High-performance scientific computing on fpga aboard the solar orbiter phi instrument. (https://pdfs.semanticscholar.org/80c3/b974edbd921e0558245a48602801b68af3b6.pdf)

[18] https://developer.nvidia.com/cuda-zone

[19] https://www.nvidia.com

[20] https://www.studio1productions.com/Articles/NVidia-GPU-Chart.htm

[21] https://developer.nvidia.com/cuda-faq

[22] https://www.aanda.org/articles/aa/pdf/forth/aa35305-19.pdf

[23] https://www.nasa.gov/feature/goddard/2020/solar-orbiter-returns-first-data-snaps-closest-pictures-of-the-sun