# Hardware/Software Integration in Solar Polarimetry

Mark Shand and Laurent Moll
Digital Equipment Corporation, Systems Research Center
Palo Alto, California
{shand,Laurent.Moll}@acm.org

## Abstract

*A polarimetry system for solar astronomy is presented. The system is based on a reconfigurable coprocessor attached to a conventional workstation. Although the computationally intensive parts of the application are performed in the host processor, the reconfigurable coprocessor plays a key role in taking charge of tasks that the host performs poorly, such as cycle-by-cycle data marshalling and real-time synchronization.*

*The runtime environment supporting the reconfigurable coprocessor makes it easy to experiment with different implementation tradeoffs and postpone the final partitioning of functionality between the host and reconfigurable coprocessor until quite late in the design process.*

## 1 Introduction

Since 1993, DIGITAL has collaborated with the Royal Swedish Academy of Sciences Research Station for Astrophysics. Together we have constructed image acquisition systems based on reconfigurable technology for the Swedish Vacuum Solar Telescope (SVST), a solar observatory in the Canary Islands. In the process, SVST has become a showcase for reconfigurable technology, with reconfigurable coprocessors involved in all aspects of the operational data acquisition and telescope control system[1, 2, 3].

While the use of these coprocessors has proved to be very successful, SVST scientists and engineers have no vested interest in reconfigurable technology and throughout the collaboration have vigorously resisted suggestions to gratuitously move processing into FPGA-based coprocessors. As the custodians of a widely used international scientific resource, their resistance is based on entirely pragmatic concerns:

- The regular appearance of new models of host workstations using faster processors is guaranteed, and software-based applications can be trivially ported to these new machines yielding immediate performance benefits.
- Software programming is a far less esoteric art than hardware programming resulting in easier application maintenance from a far larger pool of skilled practitioners.

These concerns have led to a guiding principle in all our work for SVST:

> If it can be done on a microprocessor, it will be done on a microprocessor.

Put less succinctly, this means, if performing some part of an application on the host microprocessor has no negative impact on overall application performance, then it will be done on the host microprocessor. Consequently, we have accepted only those aspects of applications that cannot feasibly be implemented in a microprocessor as candidates for implementation in the reconfigurable coprocessor. These aspects tend to be cycle by cycle data marshalling and interfacing with external signals that are strongly time sensitive. From a number crunching perspective, these may not appear to be very taxing parts of a computation, but they are precisely the tasks that a microprocessor does extremely poorly. It is thus easy to gain huge benefits in overall application performance by moving them to a modest reconfigurable coprocessor.

In building I/O oriented coprocessors, we follow a long tradition in computer engineering with antecedents in the channel processors of IBM mainframes, the Peripheral Processing Units (PPUs) of Seymour Cray's Control Data 6600 and even the slave DMA controller of the PC ISA bus. The advantage in basing these coprocessors on FPGAs is that it is simple to do bit-level manipulations, to implement multiple concurrent processes and to handle arbitrary external clocking schemes.

To exploit this added flexibility fully, it is important to provide an environment that allows the FPGAs to be programmed as part of an application. Working this way, application developers can easily experiment with different implementation tradeoffs. Final decisions on the partitioning of an application between the FPGAs and the host can be postponed until quite late in the design process. At that time, decisions can be made based on actual measurements of the running system.

This ability to make late decisions is particularly valuable in the development of scientific instruments. In many cases, the devices used in the experiments are new and incompletely specified. Only by measuring them in-system, can correct design choices be made. Even though the main reason for using reconfigurable coprocessors is to provide a fast and reliable data acquisition environment during the experiments, the possibility to evolve the interfaces during the setup of the experiments turns out to be very important.

Our reconfigurable coprocessor may seem similar to commercial frame-grabbers, which are commonly implemented with FPGAs — a logical engineering choice given the small size of the high-end frame-grabber market. The key difference is that, in these commercial systems, the FPGAs are not open to the application. They are closed, fixed function devices with no facility for user programming. They do not permit the experimentation that is necessary in developing scientific instruments, and they lack the flexibility to implement the complex synchronization mechanisms required by our applications. Of course we are not alone in recognizing the utility of a modest FPGA resource for interfacing and real-time tasks[4, 5].

This paper focuses on one instrument built for SVST: a polarimetry system. TURBOchannel Pamette, the reconfigurable board we use, has been described elsewhere[1], and the support software for this and a more modern successors are available on the world-wide-web[6]. TURBOchannel Pamette consists of a simple $2 \times 2$ matrix of Xilinx XC4010 FPGAs. Section 2 describes the polarimeter requirements. Section 3 describes how the primary computational needs of the polarimeter are met through a combination of pre-formatting in hardware and optimized software. Section 4 describes the various synchronization needs of the polarimetry system and how they are met. Section 5 concludes the paper.

## 2 Polarimetry

The imaging polarimeter is an important instrument for the study of local variations in the polarization of light from the sun and the planets which is induced by variations in the magnetic field or fundamental scattering properties of light. The polarimeter functions by recording images of a region of interest in several polarization states. For example, the magnetic field permeating the region affects the relative intensity of light in different polarization states, and so pixelwise matrix computations on the images can be used to computer the strength and direction of the magnetic field.

The instrument setup can be seen in figure 1. The components are:

- A 12b/pixel, $789 \times 245$, 60 images/s Peltier cooled CCD camera, custom built for solar imaging.
- A fixed linear polarizer and two electrically tunable liquid crystal retarders with rapid response time.
- A DEC 3000 Alpha workstation, with a TURBOchannel Pamette, used for real-time control of the polarizer and acquisition of images coming from the camera.

One important issue in all earth based astronomy is image perturbation induced by thermal fluctuations in the earth's atmosphere, otherwise known as *seeing*. One consequence of seeing is that an image taken at one time will be distorted and blurred in a different way from an image taken at another time. In order that the polarization measurements are not affected by seeing, the images in different polarization states should be taken as close together in time as possible. Naively one might propose the use of multiple cameras to capture all the polarization states simultaneously. However, the difficulties of aligning up to four optical paths and compensating for variations across the multiple cameras renders this approach impractical, not to mention the cost of multiple specialized CCDs. Instead, a single CCD is used. Polarization states are imaged through narrow band filters and typically require several seconds of integration (exposure) time to accumulate enough photons to adequately resolve polarization variations. In this scheme, if several seconds of time elapse between the capture of an image and the next, then randomly varying seeing-induced distortion will completely swamp the weak polarization signals. The solution to this problem is to replace long exposures in different polarization states by sequences of minimal exposures across all the polarization states, accumulated in real-time, as shown on figure 2. The images still suffer seeing-induced degradations, but these are roughly the same for all the measurements which strongly reduces cross-talk between the polarization measurements.

The physical accumulation of photons on the CCD is thus replaced by additions of many small quantities performed digitally. This technique creates a high computational load in the digital domain, where we must accumulate 12 Mpixels/s (193305 pixels/image at 60 images/s).

The second important issue consists of the strong real-time constraints. The exposure of the CCD being continuous, it is critical to change the retardances of the liquid crystals at the same moment that the camera switches from one image to the following image. This must be done with

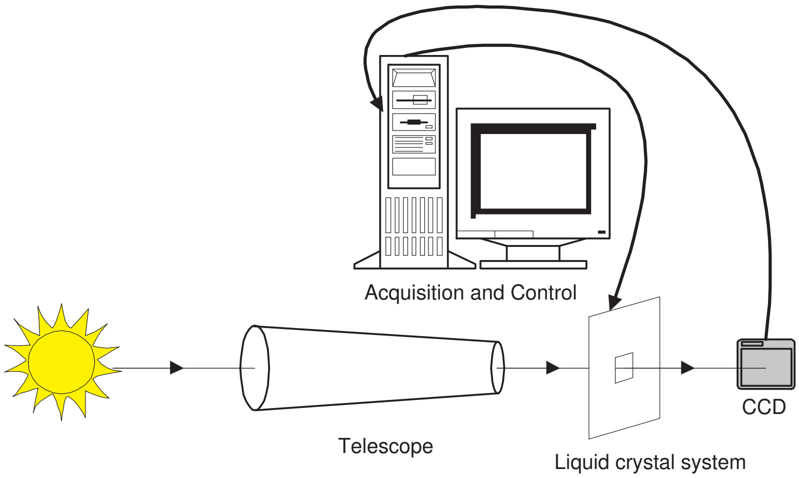


Figure 1: Polarimeter setup.

precision. At a frame rate of 60 Hz, the exposure time for an image is 16.7 ms. A skew of just 1 ms in the timing of the liquid crystal state transition relative to the commencement of a new frame results in the substitution of 6% of photons collected during a particular frame with photons from a neighboring polarization state.

## 3 Accumulating images at 12 Mpixels/s

The first issue with the polarimeter is to get the system (TURBOchannel Pamette and workstation) to compute the pixel accumulation in real-time. Following our principle of using software wherever possible we seek to do the bulk of the work on the host. Our intuition says the host (a DEC 3000 Model 900 workstation based on a 275 MHz Alpha processor) should have enough compute power and bandwidth through its memory and I/O bus to achieve the required throughput. The flexibility of the TURBOchannel Pamette used as a camera interface will allow us to use the workstation to its full potential[7].

As the incoming images use 12 bits per pixel, the format of the final accumulation buffer should use at least 24 bits per pixel to allow the accumulation of thousands of images without overflow. Consideration of natural machine wordsizes argues that pixels should be aligned on 32-bit word boundaries, so the final accumulation buffer should use 32 bits per pixel. In any case, the formatting leads to data sizes that are at most half the size of the 64-bit data word of the Alpha processor. With some alignment precautions, we can thus perform multiple pixel additions per machine addition operation, using the ALU of the processor as a SIMD machine operating on 64-bit words. This reduces by at least a factor of two the number of instructions executed on the CPU.

The choice of the data format is important because it determines not only the total size of the data to be manipulated, but also the required bandwidths within the processor, to and from memory, and on the TURBOchannel bus. Figure 3 shows the four formats we considered. For each of these methods, a simple design on the Pamette and a small C program have been used to evaluate that method's performance and hence its viability.

The methods represent various tradeoffs in bus bandwidth, memory bandwidth and CPU operations. These different solutions assess a range of the possible critical paths in this application. The bus bandwidth is directly calculated from the degree of packing and the pixel rate. The accumulation bandwidth estimate assumes that for each pixel read from the camera buffer, a read from the in-memory accumulation buffer and a write to the same accumulation buffer will be needed. Given the size of the buffers (5 to 10 images), the accumulation data is never in any of the caches by the time it is required again for the next incoming image. The *other processing* column shows additional operations that may consume CPU time on top of the memory accesses and additions.

For a frame rate of 60 Hz, the workstation has a budget of 16.7 ms for the accumulation of each frame — a little less if possible to allow for other concurrent uses of the machine.

- Method 1 optimizes the I/O bus bandwidth by packing 5 pixels in a 64-bit word (quadword). The high computation time is due to the extraction of the pixel fields by the microprocessor to prepare them for the addi-

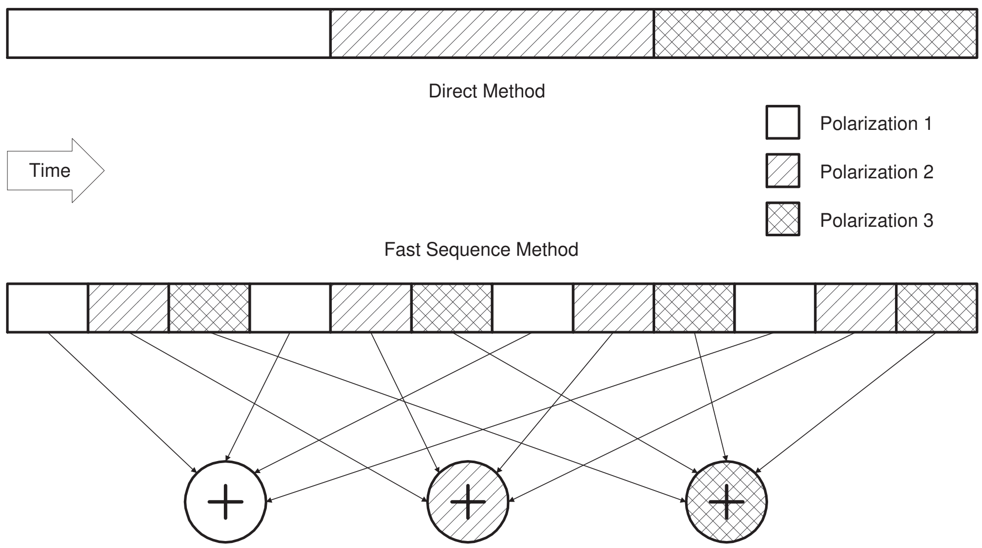


Figure 2: Polarimeter sequencing.

| | CCD | Bus | | Accumulation (memory) | | Tot. RAM | Other | Total |
|---|---|---|---|---|---|---|---|---|
| | Format | Format | Bandwidth | Format | Bandwidth | bandwidth | processing | time |
| 1 | 12b | 12b (5 pel/64b) | 19 MB/s | 32b | 115 MB/s | 134 MB/s | extraction | 26 ms |
| 2 | 12b | 32b (1 pel/32b) | 48 MB/s | 32b | 144 MB/s | 192 MB/s | no | 20 ms |
| 3 | 12b | 16b (2 pel/32b) | 24 MB/s | 32b | 120 MB/s | 144 MB/s | masking | 17 ms |
| 4 | 12b | 16b (2 pel/32b) | 24 MB/s | 16b (+32b) | 72 MB/s $+\epsilon$ | 102 MB/s | masking | **10 ms** |

Figure 3: Data formatting for the accumulation.

tions. The large total time shown for this method indicates that the CPU is overloaded, probably because of pixel extraction. It is thus necessary to find a solution using better aligned pixels.

- Method 2 tries to minimize data manipulations like field extraction. The TURBOchannel Pamette prepares the data so that the image buffer and the accumulation buffer formats correspond. The CPU just reads the data and performs the additions. This method is faster than the preceding one, but still too slow. This can be due to either the memory bandwidth (somewhat larger than for method 1), or the ALU usage (minimal, but maybe still too much).

- Method 3 lies between the previous two methods. It packs the pixels more aggressively than method 2, and by masking alternatively even and odd pixels, we can do the additions directly to a suitably organized accumulation buffer. The resulting CPU time is significantly better, which suggests that the time needed to compute the additions is within budget, and that method 2 is indeed limited by memory bandwidth.

- Method 4 is a software variant of method 3. Its purpose is to reduce the required global memory bandwidth and to use the CPU more efficiently. The principle is to use a two-level accumulation scheme. The first level uses an accumulation buffer in which each pixel uses 16 bits (instead of 32 for the standard no-overflow buffer), the same format as for the data coming from the Pamette. With the incoming 12-bit pixels accumulated into 16-bit fields, only 16 images can be accumulated without risking an overflow. In this 16-bit format, the 64b ALU of the Alpha can be used to process 4 16b additions at a time. Moreover, the data in the accumulation buffer and in the image buffer have the same format. This further reduces the amount of raw CPU power needed. After 16 accumulations in the first level accumulation buffer, it is necessary to accumulate the data in a second level 32 bits per pixel accumulation buffer. This operation is quite similar to the processing needed in method 3, and so quite costly, but is performed only 1/16th of the time. It can thus be amortized over the next 16 incoming images. Once this is done, the first level accumulation buffer can be reset to zero and the first level accumulation can continue. The time to perform the first level accumulation is around 9 ms, which is much better than the other methods. The second level accumulation requires 20 ms of processing, which amounts to a little more than 1 ms per first level accumulation when spread over 16 of them.

Only method 4 has a computation time within our 16.7 ms budget, and it is thus the only viable choice. We note in passing that had none of the host-based accumulation schemes met the time budget, we could have considered reconfigurable coprocessor based schemes, but this proved not to be necessary. The three elements that help method 4 meet the performance goal depend on both software and hardware optimizations:

- The data transfer scheme between the Pamette and the host memory uses a specific Direct Memory Access (DMA) engine with chained command lists so that minimal intervention is required from the processor. Consequently, the dependence on the real-time capacities of the operating system is also minimal. This leaves as much CPU time as possible for the accumulation task. Thanks to the versatility of the TURBOchannel Pamette board, we could implement a customized DMA engine, whose design was able to evolve in response to application needs even as late as the testing phase of the project.

- The data transfer format is chosen carefully to minimize bandwidths on both the I/O bus and the central memory, and to maximize the useful computation delivered by the CPU.

- To get the most from the CPU, it is important to use the ALU for functions for which it is well adapted, and to parallelize the processing as much as possible. The two important elements are to choose the data transfer format so that a minimal amount of reformatting is required from the processor (none in the chosen method), and to enable the use of the adder as a SIMD machine, which optimizes the use of the 64-bit ALU.

The overall result is an application that meets demanding performance constraints.

# 4 Real-time constraints

The other major issue in the polarimeter design is the control of the liquid crystals, the camera and the data transfer. The problem is to synchronize the hardware and software parts; in particular, the liquid crystal voltages must be updated with sub-millisecond precision

## 4.1 The liquid crystal system

The liquid crystal system, when combined with a linear polarizer, assumes a polarization state based on two applied external voltages, which in the case of the polarimeter instrument comes from a digital to analog converter driven by TURBOchannel Pamette. The liquid crystal exhibits a worst-case stabilization time on the order of 1 ms, but when moving between most states, particularly with an *increasing* voltage, this time is much smaller. On the other hand, the time window during which the camera does not expose any image is quite small. The scheduling of the polarization state transitions has thus to be very precise so that most of the change happens in this window. Matters are further complicated by the existence of a way to accelerate the move from one polarization state to the next: when the polarization needs to be incremented by a relatively small value (which is the case at all times except after the last transition in a sequence) the nematic effect can be used: it consists of briefly driving the command voltage well above the new target value before settling at the new value. This has the net effect of reducing the stabilization time of the liquid crystal. Both the standard and the accelerated command modes require a precise time base, as well as hard real-time responsiveness in the emission of control voltages to the liquid crystals.

## 4.2 Filter and camera control

From the outset, intuition told us that the image acquisition must be done by the reconfigurable board, and the accumulation could be done by the host processor. However, we had no such intuition about where best to deal with the control of the camera signals and of the polarizer. It can of course be done in the reconfigurable board, but a host-based solution is easier to program and test, and more versatile. In the latter case, an interrupt mechanism could be used to send synchronization information from the hardware to the software, which could then use processor writes to the registers of the board to send commands to the camera and polarizer.

Figure 4 illustrates a scheme of camera and liquid crystal synchronization realized entirely on the host processor. In this scheme, when the camera begins the exposure of an image, the Pamette sends a message to the driving software through an interrupt. The software then waits for the exposure to complete (16.7 ms in this case), then sends a message to the board to stop the exposure and start the data transfer. Some time after this, the camera automatically starts exposing a new image. Concurrently, the software

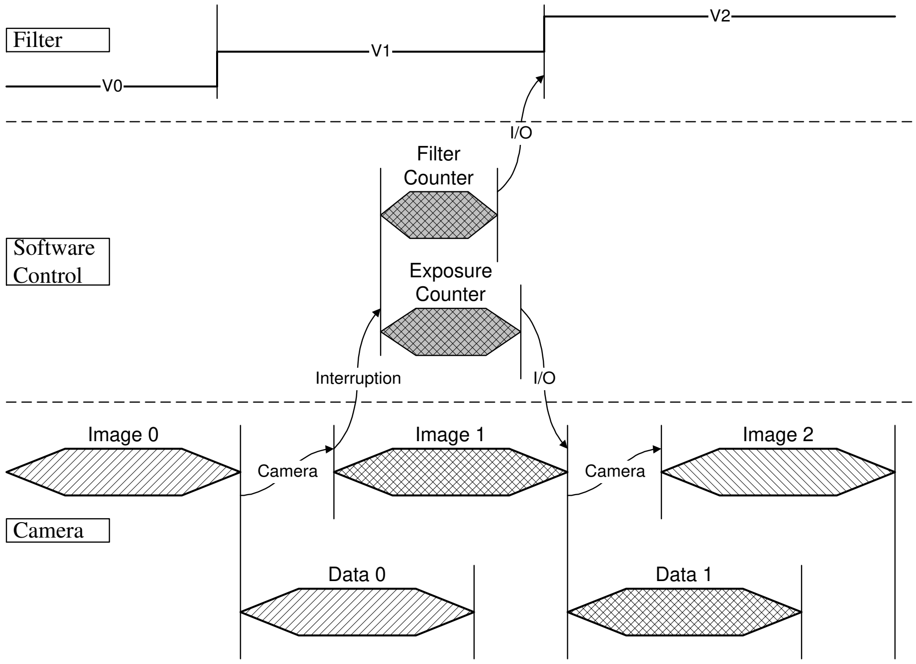


Figure 4: Software synchronization for the polarimeter.

waits for the appropriate moment to change the value of the liquid crystal control voltages, usually shortly before the end of the exposure.

An entirely host-based implementation of the synchronization relies on the real-time capabilities of the host operating system, Digital Unix 3.2 in our case. The operating system has a critical role in the dispatch time of the interrupts from the hardware to a user-mode process, and also in its ability to schedule the process with a high precision to allow it to write to a board's register at the appropriate moment. Unfortunately, tests on the system show that although most of the interrupts are dispatched in less than 100 microseconds, sometimes, this delay grows to many tens of milliseconds. These long delays are usually seen during heavy disk activity which the polarimeter application periodically generates while saving images from a previous set of accumulations. A more detailed study on the real-time capability of operating systems can be found in [8].

The polarimeter cannot tolerate such variations. It is thus impossible under our current operating system to use a software implementation of the polarizer and camera controls if we want a reliable instrument. It is straightforward to put the exposure time counter in the TURBOchannel Pamette. Of course, it must be programmable, but its period doesn't change within the same measurement. Accurate timing of the control of the liquid crystals is somewhat more challenging. We must apply a sequence of preprogrammed voltages with a high temporal accuracy. However, a hardware implementation of this task gives absolute confidence in the scheduling of voltage changes, and enables the use of a scheme exploiting the nematic effect. The timing control falls from milliseconds in software (the resolution of the Unix scheduler) to tens of nanoseconds (the frequency of the coprocessor clock), which is far better than what is needed for even the finest voltage control schemes.

Figure 5 shows the hardware implementation of the synchronization of the camera and liquid crystal controls.

### 4.3 Synchronization with the software accumulation

With exposure and liquid crystal control handled by hardware, the last problem related to real-time scheduling is the synchronization between the whole reconfigurable coprocessor machinery and the accumulation. Accumulation is the only element which is still processed by host software. Given the insufficient real-time performance of the operating system, the hardware and software parts must be as independent as possible; no hardware timing should critically depend on the timeliness of software.

Of course, some minimal synchronization must be provided between the parts of the application running on the host and those implemented in the reconfigurable coprocessor. Such synchronization avoids overflows or under-

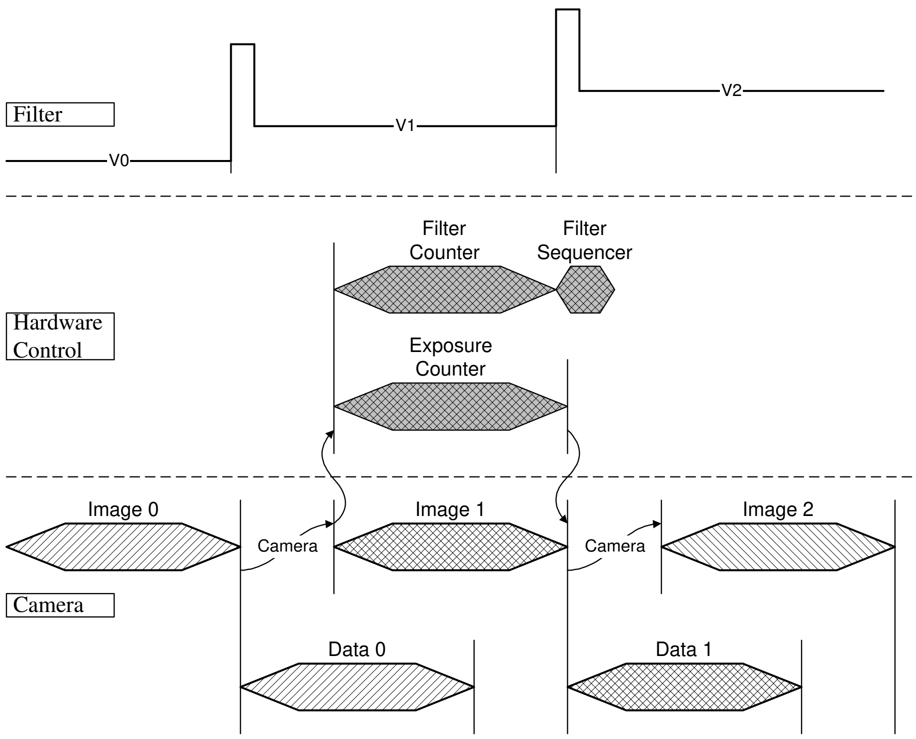


Figure 5: Hardware synchronization for the polarimeter (with nematic effect).

flows of the image buffers in the host memory used by the reconfigurable coprocessor to store the captured images. The continuity of the image capture within the same sequence is absolutely essential, because the accumulation method used to minimize the influence of atmospheric perturbations specifically relies on the temporal proximity of the images within a sequence. It is thus natural to insert synchronization points between sequences (which contain between 2 and 6 images recorded with increasing voltages for the liquid crystals). Moreover, although we can minimize the stabilization time of the liquid crystal by using a monotonically increasing ramp of voltages, eventually we must apply a decreasing voltage and suffer a relatively long stabilization time. When we make this transition, it is prudent to discard the associated image, so we must accept losing one frame per sequence. It is then safe to wait for the software to send an acknowledgement signal to start a new sequence. Most of the time, the signal will come after a minimal delay, and only the necessary frame will be lost, so the synchronization won't cost anything; sometimes however, the synchronization will take more than one frame, but this is rare and in any case it has no influence on the quality and the coherence of the resulting magnetograms.

Figure 6 shows where the software intervention is needed. At this point, software can delay the start of a new sequence, which is triggered by the loading of a new command array in the DMA engine.

## 5 Conclusion

Shortly before we began work on the polarimeter instrument, a group of SVST collaborators at the Instituto de Astrofísica de Canarias (IAC) had made a preliminary study of a polarimetry system very similar to the one just described, but using conventional engineering techniques. Lacking a reconfigurable coprocessor, they were unable to organize the incoming pixels in a manner that would allow treatment on a conventional workstation. Instead, they had been forced to opt for a dedicated image processor from Datacube Corp with integrated camera controller. Their system consisted of the following elements:

- A PC with a commercial parallel I/O card for control of the liquid crystals.
- Two liquid crystals.
- The Datacube system.
- A workstation to provide overall control of the system with a graphical user interface.

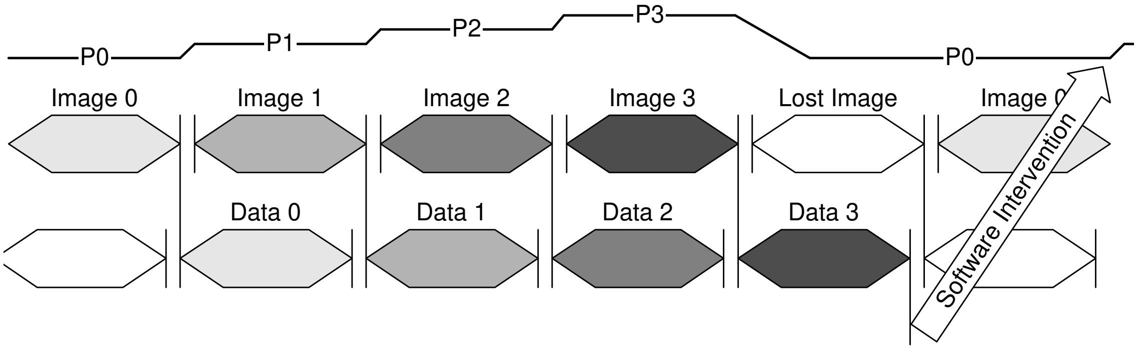


Figure 6: Software intervention for the polarimeter.

The project was estimated as requiring several man years of engineering. The resulting system is extremely heterogeneous and the real-time interaction between the various elements are difficult to guarantee.

The TURBOchannel Pamette solution was developed in just three man-months using the code of the existing image acquisition system as a starting point. Apart from the liquid crystals, no new hardware was required. Thus, the engineering effort was much less and future developments were facilitated due to the reprogrammability of the entire system.

The use of a reconfigurable coprocessor in the development of this sort of instrument has many advantages:

- Reduced development time and cost.
- Physical and logical design aided by the ability to rapidly prototype competing solutions.
- Simplified maintenance through a programmatic interface to the entire system even from remote locations.
- Homogeneity of the development platform and the possibility to easily add new functions.

For all these reasons, and because the system based on the TURBOchannel Pamette was already operational when their system was still in the specification phase, the IAC group chose to adopt a solution based on reconfigurable technology[9]. The IAC system is based around the successor to TURBOchannel Pamette, the PCI Pamette. Likewise, encouraged by the widespread adoption of the PCI bus in workstations and PCs, the SVST engineers are actively porting existing instruments and developing new instruments around PCI Pamette.

In figure 7, we see an example of the output of the polarimeter instrument. The array of images consists of a series of polarization measurements of the planet Venus. The weak fringe patterns are artifacts from the A/D converters on the CCD readouts.

## 6 Acknowledgements

Göran Scharmer, director of the SVST, conceived the basic polarimeter design and played an active role in the final stages of the instrument development during on site testing at SVST.

## References

[1] M. Shand, “Flexible image acquisition using reconfigurable hardware,” in *FPGAs for Custom Computing Machines (FCCM’95)*. IEEE, April 1995,

[2] M. Shand, W. Wei, and G. Scharmer, “A 3.8ms latency correlation tracker for active mirror control based on a reconfigurable interface to a standard workstation,” in *Photonic East Symposium ’95*. SPIE, October 1995.

[3] L. Moll, “Application des Mémoires Actives Programmables,” PhD thesis, Ecole Polytechnique, 1997 (in French).

[4] B. Heeb, C. Pfister, *Chameleon: A Workstation of a Different Colour*, in *Field-Programmable Gate Arrays: Architectures and Tools for Rapid Prototyping. Second International Workshop on Field Programmable Logic and Applications*, Vienna, Austria, August 1992.

[5] TSI TelSys *Reconfigurable Computing Systems* World-wide-web http://www.tsi-telsys.com/ 1997.

[6] Mark Shand *PCI Pamette V1* World-wide-web http://www.research.digital.com/SRC/pamette 1996 and 1997.

[7] Mark Shand *Configurable Digital Signal Interface using Field Programmable Gate Arrays to Reformat*

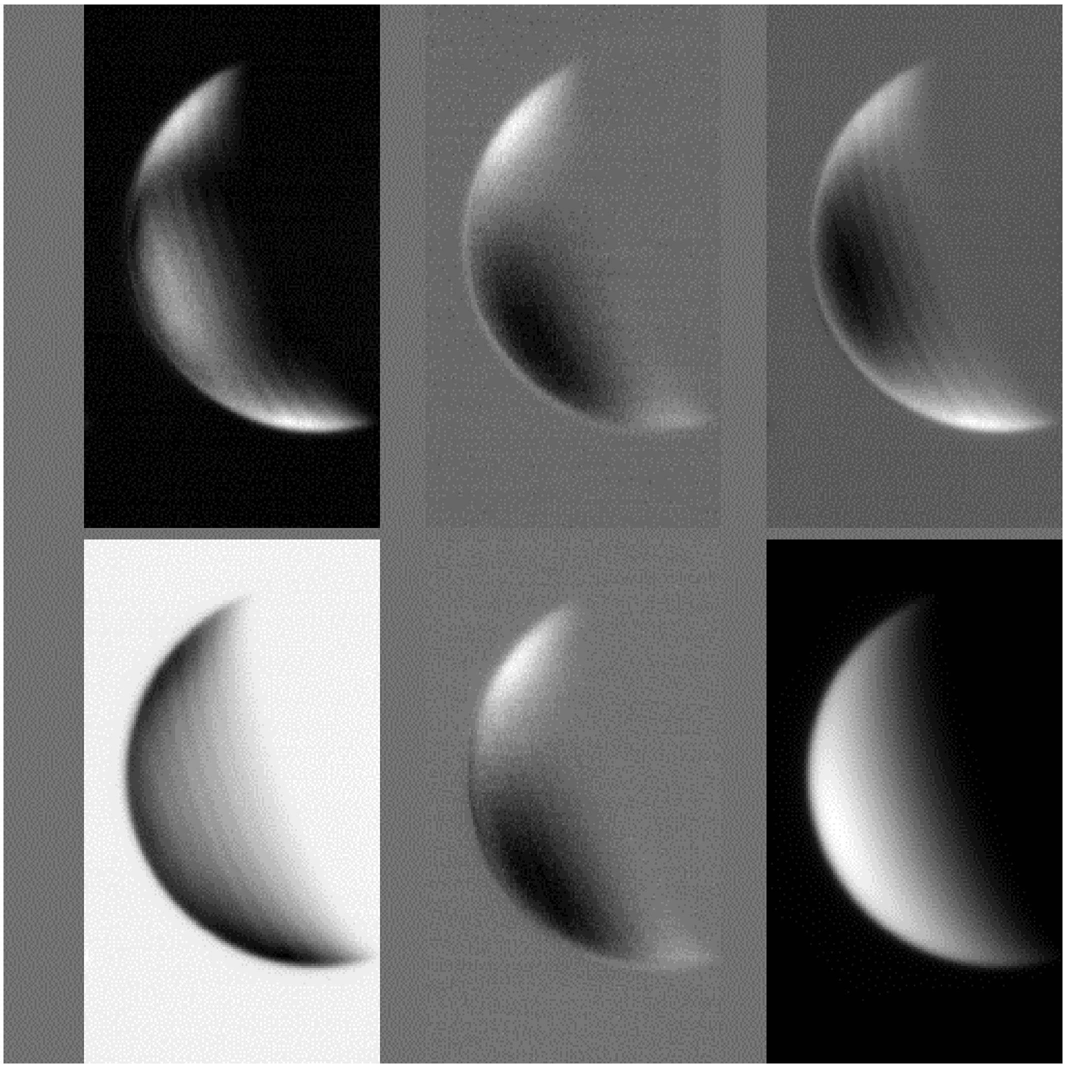

Figure 7: Polarization measurements of Venus.


*Data* United States patent No. 5,692,159, Nov. 25, 1997.

[8] L. Moll and M. Shand, “Systems Performance Measurement on PCI Pamette,” *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa, CA, April 1997.

[9] J. Sanchez Almeida, M. Collados, V. Martinez Pillet, V. Gonzalez Escalera, G.B. Scharmer, M.Shand, L. Moll, E. Joven, A. Cruz, J.J. Diaz, L.F. Rodriguez, J. Fuentes, L. Jochum, E. Paez, B. Ronquillo, J.M. Carranza and T. Escudero Sanz, “The IAC Solar Polarimeters: Goals and Review of Two Ongoing Projects,” *1st Advances in Solar Physics Euroconference*, eds. Schmieder B., del Toro Iniesta J.C., and Vazquez M., ASP Conf. Ser., 1997.