LRP 586/97 Octobre 1997

A HIGH-PERFORMANCE DIGITAL CONTROL SYSTEM FOR TCV

J.B. Lister, M.J. Dutch, P.G. Milne, and R.W. Means

Paper presented at the
10th IEEE Real Time Conference
Sept. 22 - 26, 1997, Beaune, France

A High-Performance Digital Control System for TCV

J.B. Lister[1], M.J. Dutch[1], P.G. Milne[2] and R.W. Means[3]
[1]CRPP-EPFL, Association EURATOM-Confédération Suisse, 1015 Lausanne, Switzerland
[2]Pentland Systems Ltd., Livingston, EH54 9DR, Scotland
[3]HNC Software Inc., San Diego, California 92121, USA

Abstract

The TCV hybrid analogue-digital plasma control system has been superseded by a high performance Digital Plasma Control System, DPCS, made possible by recent advances in off the shelf technology.

We discuss the basic requirements for such a control system and present the design and specifications which were laid down. The nominal and final performances are presented and the complete design is given in detail. The integration of the new system into the current operation of the TCV tokamak is described.

The procurement of this system has required close collaboration between the end-users and two commercial suppliers with one of the latter taking full responsibility for the system integration. The impact of this approach on the design and commissioning costs for the TCV project is presented.

New possibilities offered by this new system are discussed, including possible work relevant to ITER plasma control development.

I. INTRODUCTION

The first few years of operation of the TCV (Tokamak à Configuration Variable) tokamak [1] has produced some excellent results with the original TCV Plasma Control System [2,3]. This equipment was constructed using analogue circuitry controlled digitally, known as the Hybrid Analogue-Digital system, a part of the current TCV Plasma Control System (PCS). The central element of the Hybrid Analogue-Digital system is a set of DACs, programmed from the TCV host computers, multiplying analogue time-varying reference input signals by the digitally set gains. In spite of the 3000 analogue multiplying elements necessary, the hardware has proven to be extremely reliable in operation and has required very few technical interventions during its four years of plasma operation. The decision to design and fabricate this system rather than elect for a fully digital system was debated at the time of design, 1988, and the hybrid system was selected for the simple reason of bandwidth. It was felt at the time, rightly so, that no off the shelf digital system could provide the cycle time and minimal delay which was considered essential for TCV operation. The hybrid system has a bandwidth greater than 20 kHz and the matrix multiplications required for plasma control were simply executed using the large number of analogue elements.

We decided to revisit this choice of analogue versus digital control for a simple reason, namely that of algorithmic flexibility. The Hybrid Analogue-Digital system performs a matrix multiplication onto the raw measurement signal vector with a matrix of dimensions 24x120 (we use the convention that the matrix dimensions are given as outputs x inputs) in order to create an estimator which is compared with 24 analogue reference signals. The resulting feedback error vector is integrated, differentiated and amplified to produce 3 signals per error element, a total of 72 signals. These signals are then multiplied by a 24x72 matrix to obtain a correction vector. This form of controller is referred to as a generalised PID controller. The correction vector is finally multiplied by a 24x24 matrix to provide voltage signals which are added to a pre-programmed feedforward time-varying vector, also summing with a 24x24 multiplication onto a raw measurement vector. Although this appears very specific, all of the algorithms which were to be implemented at the start of TCV operation could be re-cast into this form, discussed in detail in [3].

Recently, the question of modern control techniques has been given more attention among plasma control specialists during the design phase of the ITER (International Thermonuclear Experimental Reactor) tokamak, since the control challenge of ITER is perceived to be more severe than that of currently operating tokamaks. Although today's tokamaks are adequately controlled using simpler control algorithms, it is considered that modern control techniques which could offer improved performance should be tested, to validate their approach as well as to test their claimed superior robustness on a real device. TCV is ideally suited to such studies since it is more flexible and more complicated than other tokamaks. Coupled with this clear interest in modern control techniques, the appearance of an affordable supercomputer, the SNAP-64 (SIMD Numerical Array Processor) [4], triggered this project of replacing the TCV Hybrid Analogue-Digital Control System with a digital system. The decision to explore such a solution was taken in June 1995 and the Digital Plasma Control System, DPCS, was defined and designed during the same summer and autumn. A final industrial contract was placed in November. The running equipment was delivered to TCV in August 1996 and the first real operation of TCV plasmas was in spring 1997.

We discuss the design requirements for the DPCS in Section II. The procurement history is outlined in Section III, before a detailed description of the final design and implementation in Section IV. Testing and delivered performance are summarised in Section V. The initial control implementation for TCV is described in Section VI and finally we discuss the experience gained in Section VII.

II. Design Requirements

As end-user, the CRPP (Centre de Recherches en Physique des Plasmas) laid down a number of design requirements during the conception of the DPCS, split among input/output specifications, timing restrictions, calculation performance and integration.

The input-output specifications are summarised as follows:

i) Input Analogue Channels. The DPCS should have more input channels than the analogue PCS, providing flexibility in the choice of the signals which could be used for feedback control and increasing the range of control variables. The 120 inputs of the PCS should increase to over 180 signals for the DPCS.

ii) Output Analogue Channels. The PCS controls 24 feedback channels and generates 40 preprogrammed feedforward channels. The DPCS should provide up to 64 analogue output channels mixed between feedback and feedforward as required for particular operation. This total number of channels is adequate for controlling the ECRH (Electron Cyclotron Resonance Heating) being developed for TCV.

iii) Input/Output Dynamic Range. Although the PCS has 12-bit resolution on the gains, the analogue throughput is not discretised. However, it was felt that a 12-bit input and output resolution would be adequate for TCV, provided the ADC was genuinely close to 12-bit. Particular attention was to be paid to this point.

iv) Digital Input/Output. A set of 64 digital inputs and 64 digital outputs should provide future possibilities for real-time interaction with the DPCS, as well as executing the current PCS plasma current disruption alarm.

v) Acquisition. Since all of the input and output data are already digitised, the DPCS should provide its own data acquisition, to be uploaded into the TCV host computers following each plasma shot.

vi) Synchronicity. It was considered essential that the analogue inputs be sampled simultaneously, rather than allow a ripple through multiplex digitisation. This avoids problems of discrete-time control algorithms with irregularly sampled data.

vii) Front Panel. Although the system would be operated remotely, front panel indication of the state of the equipment was imposed for commissioning.

Two timing specifications were defined. Firstly, there is the repetition time of the natural digital cycle, equal to the sampling interval of the analogue inputs and also equal to the hold time of the analogue outputs, referred to as the cycle time. Secondly, there is the delay between a sample trigger and the updated outputs corresponding to the information in that sample, referred to as the latency, including the conversion time, all of the data transfer times and the calculation time. The criticality of the timing specification is due to the decision to implement the fastest feedback for the vertically unstable movement of elongated TCV plasmas [3] in the same hardware and software. We rejected a partial splitting of the feedback as being not relevant to the ITER concept testing which partly motivates this implementation. TCV is now equipped with fast internal coils [5] which will be used to control plasmas with growth rates up to 3000 /sec. Some simple arguments suggested that a delay of 100 μsec would cause problems for the control loop and that a delay of 50 μsec should be adequate. Since early in the design phase it was clear that this performance figure was to be the implementation stumbling block, we chose to define:

i) The cycle time was to be as close as possible to 50 μsec with a modest calculation overhead.

ii) The latency was to be 80 μsec or better.

The calculation performance was estimated on the basis of a state control calculation with 180 inputs, 40 controller states and 64 outputs, with a linear combination of input signals to be used to define the errors. This calculation was to be executable within 50 μsec, to be demonstrated using benchmarks provided by the CRPP.

The integration itself was also subject to specifications. The equipment was not only to be integrated into TCV in a hardware sense, but the DPCS calculator was to be accessible to high level programming, in such a way as to require no one person to be fully occupied with the system. The learning curve of the full DPCS should not disrupt an already overloaded experimental group. A maximum of work was to be provided by the contractor(s).

Although some of these specifications appear somewhat arbitrary at first sight, they are nonetheless representative of what was considered feasible after a first round of discussions with potential suppliers.

III. Procurement

Following these tentative specifications and a description of the overall project requirements, potential suppliers were approached to propose equipment corresponding to our needs. Initial discussions indicated that a VME solution was realistic, allowing use of existing ranges of high performance equipment from multiple vendors. We identified the calculator, the SNAP-64, as a VME device and of course an interesting range of input-output cards and processors. However, the data rate on the buses in the VME crate was quickly identified as a limiting factor and this remained the case up to the completion of the project. Having identified the potential of the SNAP-64, the performance of the benchmarks provided by the CRPP was provided by the company, HNC Software Inc. (HNC) within 48 hours, together with the source illustrating the end-user implementation. No other potential supplier could offer anything resembling this and the SNAP-

64 was adopted as the DPCS calculator. This rapid benchmarking convinced the CRPP that the integration of the SNAP programming would not be too onerous.

Extensive exploration of the VME ADC market indicated that there was no unit capable of fully satisfying our requirements. The closest unit was from Pentland Systems Limited (PSL) who agreed to develop and supply a new daughter board, the VGD4, to supplement their growing range of new Vanguard products. This high channel density unit was to have 96 synchronously sampled inputs.

In order to move data to and from the input-output units and the calculator, as well as archiving them, a single processor was quickly seen to be inadequate. A dual system of data movers had to be provided and in fact the PSL board could also act as bus master, making a total of 3 data mover CPUs. Added to this, the SNAP-64 is hosted by a SPARC station from FORCE GmbH., also in the VME crate. By this time, the technical specifications were being refined to the point that the complete project complexity was apparent. At the same time, PSL showed interest in a turn-key contract and the CRPP unloaded the responsibility for system integration onto PSL as prime contractor, while retaining responsibility for the integration of the SNAP-64. After some due reflexion by the CRPP, PSL was approached again for taking on the integration of the SNAP-64 into the DPCS. From that time, PSL bore sole responsibility for the delivery of a complete functioning system to the CRPP, under the restriction that the design should progress step by step in close contact with the CRPP. This was to resolve compromises which would inevitably be required due to the tight technical specifications.

Due to an unforeseen rate of trivial failures of 3rd party equipment and also due to the tightness of the timing specifications, the project took double the time from contract to shipping. At the same time, the CRPP offered a performance enhancement premium if the latency were to be reduced below the contract specifications. During the development of the project, only 3 visits to the PSL factory were required, of which the last was to perform the pre-shipping tests. The first and second visits were required to establish compromises where necessary in the light of the CRPP requirements.

Due to the complexity of the equipment, PSL was contracted to perform the acceptance tests, defined by the CRPP as part of the initial contract, at the CRPP and at the same time to integrate the DPCS environment (UNIX based) into the TCV environment (VMS based). This phase lasted for 10 days at the CRPP. During this time, HNC also provided 10 days of consultancy and training on SNAP-64 programming, on the basis of the integration provided by PSL. The time from delivery to integration into TCV consumed 10 days, after which time the DPCS was operated from the VMS environment, synchronised with the TCV timing, embedded into the TCV data management system, and both archiving and real time control of the DPCS using VSYSTEM were exhaustively tested. The complexity of the system, as well as the presence of 2 new operating systems plus a new Application Programmer's Interface (API) gave the CRPP grounds for fearing total system reliability. Following the departure of the developers, an exerciser was written and run for a few weeks, demonstrating a level of reliability acceptable to the CRPP. At that time the DPCS was accepted and an initial TCV control algorithm was implemented, discussed in Section VI.

IV. Detailed Design

The block diagram of the final system design, shown in Figure 1, indicates the interconnection of the hardware components and the data flow paths. The Input-Output Controller (IOC) software divides into i) a Real Time Loop process, controlling the acquisition cycle with microsecond critical timing, ii) a Control process, to synchronise with slow external TCV signals, with timing to tenths of seconds and iii) a log process for upload and download of data, where the timing requirement is simply "as fast as possible".

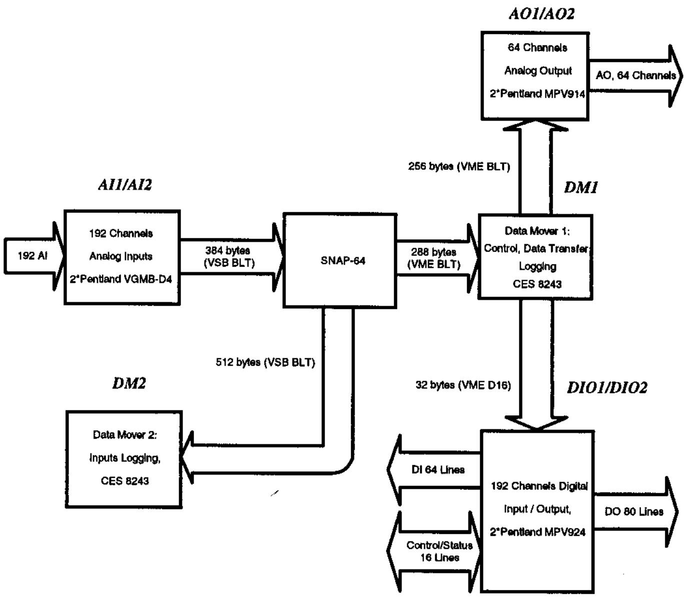
AO1/AO2
64 Channels Analog Output 2*Pentland MPV914
AO, 64 Channels
256 bytes (VME BLT)
DM1
AI1/AI2
192 AI
192 Channels Analog Inputs 2*Pentland VGMB-D4
384 bytes (VSB BLT)
SNAP-64
288 bytes (VME BLT)
Data Mover 1: Control, Data Transfer Logging CES 8243
512 bytes (VSB BLT)
DM2
32 bytes (VME D16)
DIO1/DIO2
Data Mover 2: Inputs Logging, CES 8243
DI 64 Lines
192 Channels Digital Input / Output, 2*Pentland MPV924
DO 80 Lines
Control/Status 16 Lines

Figure 1. System block diagram showing interconnection of the hardware components and data flow paths.

Some of the major design features are:

i) Input and Output processing trigger off independent events. The requirement that system latency should reduce as the outputs calculation time is reduced, meant that the outputs handling process should synchronise with the calculation process. The inputs handling process must synchronise with the sample clock, without using VME bus bandwidth. The cycle time requirement meant that interrupts could not be used, as this is of the same order as the interrupt handling time under OS/9. Hence, polling methods were required, which further increases bus loading.

ii) A high degree of overlap. Data for the next cycle must be being prepared while the SNAP is busy on the previous cycle. Output values from the previous cycle must be converted without interfering with new input values.

iii) A high average data rate (approximately 1200 bytes per 50 µsec cycle), which meant that the most efficient possible data transfer mechanisms were required.

iv) Reversible data file format. The CRPP developed a flexible portable file format based on the Sun XDR standard to handle the large volumes of logged data. The format allows transfer of all or part of a data set. The required data set may be any subset of channels and/or time samples.

The requirement for simultaneous data capture meant that a single convertor per channel architecture was chosen for the VGD4 ADC design. A 10 µsec convertor was a good compromise of conversion speed, power consumption and cost; a 2 µsec convertor saves 8 µsec but carries a heavy power consumption and heat dissipation penalty, particularly as nearly 200 were required. The core ADC chosen was a Burr Brown ADS7804. A digital output version was required for speed. The digital logic is quiet during conversion for the best noise performance. The design has the fastest possible transfer to FIFO, and VSB BLT (Block Transfer) transfer is software selected to begin on FIFO not empty to minimise latency. The board has mirror image components on each side and has 10 layers. Offset and gain controls are not included for reasons of space and cost, but a table of offsets and gains is provided in on-board PROM to allow software calibration.

The high average data rate meant that it was impracticable to use a single bus. Fortunately, the SNAP is a VSB slave and the VGD4 is a VSB master, giving a good path for low latency. Both VGD4 boards attempt to become bus masters concurrently, and the VSB bus arbiter hardware allows each board to make its transfer back to back. The overlapped nature of the system gives rise to a need to make both buses work concurrently – mandating two Data Movers. The design split logically into one Data Mover handling Inputs and using the VSB bus, the other handling Outputs and using VME. The triple-ported SRAM global memory on the SNAP was able to support concurrent accesses on VME, VSB and from the SNAP internal processor bus without noticeable delay.

The OS/9 RTOS operating system from Microware was selected to run the Data Movers which were selected as FIC 8243 from CES. Hosted on one of the real time target systems, OS/9 provides a cost effective solution providing real time, file and network services.

Data handling was the first task to be implemented. The reversible data file format made proof of data upload/download integrity easy. Tools to view the data were developed at an early stage, and were used extensively during the development. The critical part of the software is the Real Time loop, coded in assembler for speed. During execution, interrupts are masked to prevent any degradation of the cycle time. The loop was implemented by outputting state data to a parallel port on the CPU board, and state timing checked using a logic analyser, allowing easy check of performance relative to design and an accurate view of the effect of optimisations. Each data path was verified for correct operation and the transfer speed measured, and the time line (Figure 2) rechecked.

The decoupling of the Input and Output processes was achieved by making one Data Mover (DM1) responsible for outputs handling on the VME bus, and the second DM2 responsible for logging Analogue Input and Intermediate values via the VSB bus. During the real time cycle, DM1 also handled the less critical Digital Inputs, after the critical Analogue Outputs and Digital Outputs had been dispatched. A dedicated status cable was connected between the VGD4 and DM2 front panels, routing VGD4 status signals to the parallel IO port, to allow DM2 to poll for acquisition complete without consuming bus bandwidth.

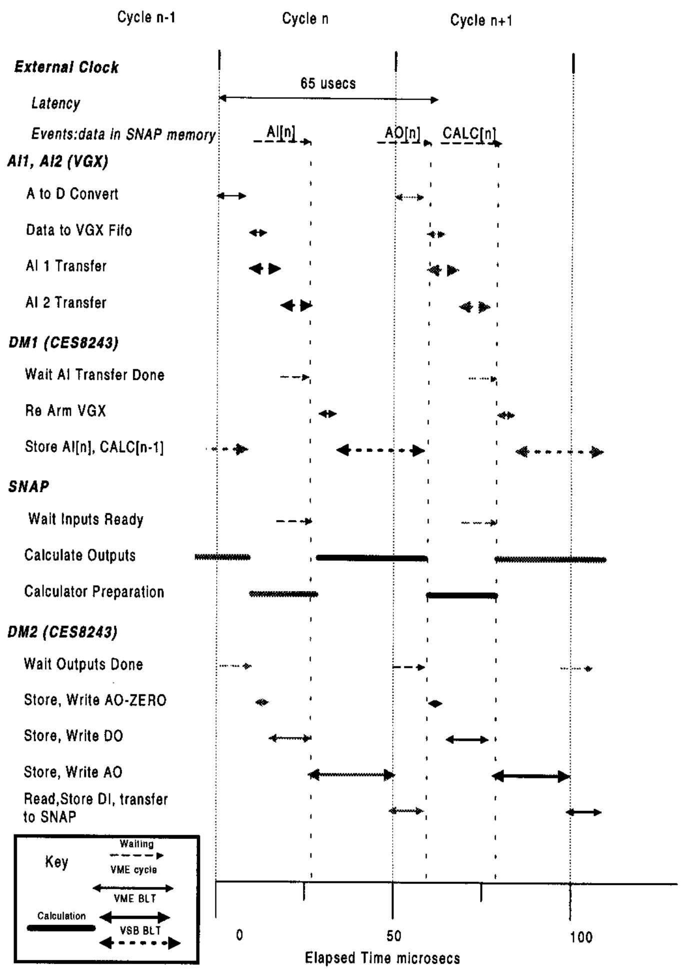

Figure 2. Timeline of the DPCS software, showing the activity on the buses, the mastering of the transfers and the SNAP calculations.

This arrangement meant that DM2 was responsible for the bulk of the logged data. However the memory usage of the boards was configured asymmetrically, assigning DM2 a large log memory area and a small area for program use, while DM1 was configured to have a larger area for program use, allowing DM1 to be used as the development platform.

The SNAP-64 is a Single Instruction Multiple Data (SIMD) parallel processor. The architecture, Fig. 3, comprises the 64 individual processors with a 20 MHz cycle, local memory for each processor, a nearest-neighbour bi-directional systolic bus, global memory accessible to any processor one at a time, a system controller and a host. The host is a SPARC CPU, supplied as a VME module with a local disk, terminal port access and thinwire Ethernet access.

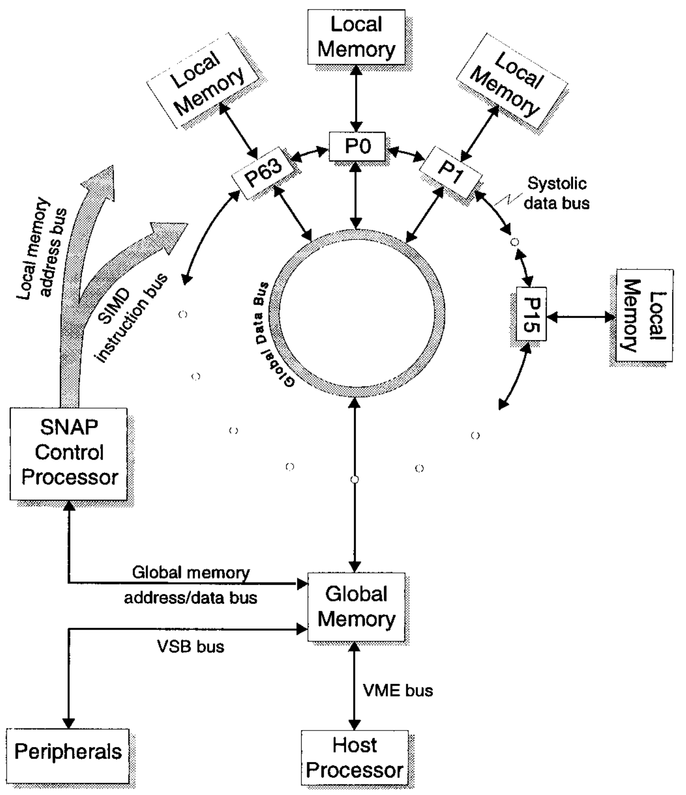

Figure 3. Overall architecture of the SNAP SIMD Numerical Array Processor.

The relatively slow processor speed of the SNAP hides an ability of the processors to perform several operations per cycle, such as multiply and accumulate. Given this feature, the unit is capable of providing a calculation performance of 64x2x20 MFLOPS, namely 2.5 GFLOPS. Against this impressive performance is a relatively significant overhead for each primitive function call, of the order of 1 µsec and the relatively slow serial transfer of data from global memory to local memory. For our application, all input and output data transfers must be to global memory, mapped to VME bus addresses. The question of the calculation performance of the DPCS implementation is taken up again in Sections V and VI.

The high function call overhead is mitigated by the very complete set of functions in the API, extending from basic mathematical operations up to single-pass neural network calculations. The API takes the form of C-callable routines, obviating the requirement of mastering a new language or a new Operating System. The functions of most interest for the DPCS initial implementation are matrix-vector multiply operations and standard vector operations.

A major feature of the SNAP API is that the application programmer has no knowledge of its parallel architecture. The API dynamically linked library is implemented in both a non-SNAP form, executing the API calls in C in the host for testing, and as the API calls to the SNAP itself.

The final step in using the SNAP in a real-time application is to remove the communication between the SUN host and the SNAP when executing repeated sequences of API function calls. This mechanism, referred to as chaining, simply collects the blocks of function calls and their associated data structures into a chained list which is memorised once it is found to have become repetitive. This list can be downloaded into the SNAP together with the run-time data and executed repetitively forever, after which no host-SNAP communication is necessary or even possible. At this point, the SNAP execution has become deterministic, an essential part of a real time application.

The application program lifecycle from editing the source to testing the real-time application is of the order of 2 minutes, allowing rapid development and testing of an application.

In order to integrate the SNAP-64 into the DPCS, a set of five acceptance algorithms was defined by the CRPP. These were also conceived to demonstrate the functionality of the data mover software itself. PSL provided all of the C-code which hid the copying of the data to and from the VSB bus as well as the simple demonstration calculations. The C-code necessary to provide the context for the real-time calculations amounted to about 300 executable lines structured to be common to all subsequent DPCS algorithms and only 20 executable lines to be replaced later by TCV algorithms. HNC provided the support necessary to PSL for this to be done with minimal effort by PSL. HNC subsequently provided the CRPP with the tuition required to understand certain features of the SNAP programming necessary for maximum efficiency of its use.

V. TESTING AND PERFORMANCE

The acceptance test procedures were created as part of the design specifications. Two dummy runs of the acceptance tests were carried out at PSL to verify the correct functionality of the equipment, to identify areas of confusion and to check on the analogue and timing performance. These tests can be divided into three classes: a) Functionality, b) Timing Performance and c) Analogue Input-Output Performance.

A. Functionality

The functionality tests were all binary, either the DPCS functioned as intended or not. During commissioning several functionality features were modified at CRPP request, due to unforeseen implications of some of the design features. The high level programming of all of the DPCS made this possible. These tests included the testing of the state machine of the Input-Output-Controller and the compatibility of the file transfers between DPCS and VMS hosts. Tests were performed to assure that no trigger samples were missed and that the decimation facility provided the same data as the original data. At the same time, the operation of the DPCS was completely integrated into the TCV data system for control of the DPCS from the VMS environment [3].

B. Timing Performance

1) System Integration Timing

It was found that the XDR file transfer to be used between UNIX and VMS was particularly time consuming and performing data-type-independent XDR transfers led to a factor of 3 improvement in the file reads/writes. Floating point data were transferred using the machine independent features of the XDR structure but the integer data were transferred as type-independent and were explicitly byte-reorganised once received by the VMS host. The data for a 61,000 point full acquisition requires 50 seconds to be re-configured between the Input-Output-Controller (IOC) memory and the SUN disk (about 40 MB). Reading and unpacking the data from this NFS-mounted file and storing it into the TCV data structures takes a further 390 seconds. The total time for this maximum quantity of data from IOC memory to TCV data disks is therefore 440 seconds. Normally, we will expect to upload about 10% of this; a 6,000 point data transfer takes 44 seconds to complete. There is certainly room for improvement here using other data transfer mechanisms if ever the performance gain were to justify the effort required.

The time between new data being created for the subsequent TCV discharge, corresponding to re-programming of the waveforms, for example, or modifying the control matrices, to the DPCS being ready for triggers was under 120 seconds, compatible with TCV operation. During this period, the SNAP code must be rebuilt with the new data and the new data must be transferred to the SUN host by FTP or NFS copy.

2) Real-time Timing

The cycle time is dominated by the Data Mover activity. Assuming a modest calculation time, a cycle time of 50 μsec was not achievable as hoped, mostly attributable to the unexpected slowness of the VME BLT to the SNAP. The figure of 65 μsec finally achieved was already predicted during development once this problem had been identified. For TCV, the extra cycle time from 50 – 65 μsec is not considered to be significantly penalising and it is not clear that the SNAP calculation for realistic control calculations is going to beat this figure anyway. The CRPP therefore decided to accept the 65 μsec basic cycle time limit in preference to speed enhancements proposed by PSL which would have been at the price of reductions in the basic functionality.

The latency was measured assuming a 35 μsec delay between the NEW_DATA synchronisation semaphore to the SNAP and the DATA_READY semaphore from the SNAP to the IOC. The figure of 84 μsec achieved was within the 85 μsec target laid down, but the performance enhancement to 80 μsec was not achievable without a reduction of functionality, which was again rejected by the CRPP.

The 65 μsec cycle time, together with the 1184 Bytes of data shuffled on the VME and VSB buses during each cycle, corresponds to an effective achieved bus bandwidth of 19 MB/sec, a bus occupancy of 31% compared with the theoretical bandwidth of 60MB/sec, assuming 64-bit transfers.

C. Analogue Performance

A programmable voltage source together with a 20-bit digital voltage meter provided a simple but reliable method of testing the calibration of the analogue input cards. Automated overnight tests were performed to monitor the whole ±10 volts range of the ADCs, to check the gains and offsets and to check the noise. The performance of the VGD4 ADCs was within the specifications for all parameters. The channel offsets ranged from –2 to +5 LSB. The channel gains ranged from 0.998 to 1.006 times the nominal gain.

A simple algorithm allowed us to program the DACs over the whole range of their outputs and we used the calibrated VGD4 inputs to record the DAC analogue outputs. This was feasible since the analogue specifications for the outputs were less precise than for the inputs, justifiably since the DAC output errors are recovered by closed loop operation, whereas the input errors lead to static control offsets. However, the DACs were found to be almost as good as the ADCs. The DAC channel offsets ranged from –9 to +3 LSB. The channel gains ranged from 0.99 to 1.02 times the nominal gain.

These two tests were used to calibrate the system, in preference to the on-board factory calibration, since the on-site calibration included part of the cabling and could be repeated.

VI. Initial Digital Plasma Control for TCV

Since TCV is an operating tokamak, it was essential to provide a seamless transfer from the Hybrid Analogue-Digital system to the DPCS. Four steps were necessary to ensure this. Firstly, the DPCS had to be interfaced to the existing data structures used to drive the PCS. Secondly, the DPCS was cabled physically in parallel with the PCS. Thirdly, the real-time algorithms were developed and tested on old plasma discharges to verify the SNAP software functionality. Fourthly, the DPCS was run in parallel to the PCS to check the operational reliability and hunt for any incompatibilities. After these steps, discussed in detail in what follows, the four 16-channel cables connected to the PCS outputs were simply disconnected from the PCS and connected to the DPCS. At that point, the TCV discharges, programmed for PCS, could be run on either the PCS or the DPCS.

Interfacing to the PCS data structures [3] meant programming the DPCS waveforms from the stored waveforms already used to drive PCS. These comprise sets of time-value vector pairs, each drawing a valid waveform. The waveforms are drawn freely with a 1 msec time discretisation, either by automated procedures or by hand, or by manual modifications of procedure-drawn waveforms. These were all already drawn in MKSA units for PCS with input-output scaling factors available in the data structures for conversion to and from voltages. In PCS the union of the time-vectors is formed and the specific hardware produces voltage signals with linear interpolation between points [2]. We remained with this philosophy for the DPCS, performing a union of the drawn timebases, but performing a staircase interpolation between drawn points. Typically, a TCV discharge comprises 100–200 points in the timebase union. The time-step of the staircase was chosen to be one DPCS cycle. At this point we should remark that the first implementation was made to be as close as possible to the PCS, not minimising the DPCS cycle time, to be discussed later. The second interface was the values of the matrices used in the PCS algorithms. These were also already stored in MKSA units in PCS.

The feedback error was calculated in PCS as

$$\varepsilon = \mathbf{A}.\text{Scaling.Inputs} - \text{References} \quad (1)$$

This was transformed in the DPCS to become

$$\begin{aligned}\varepsilon &= \mathbf{A}.\text{Scaling.(Gains.Inputs + Offsets)} - \text{References}\\ &= (\mathbf{A}.\text{Scaling.Gains}).\text{Inputs}\\ &+ (\mathbf{A}.\text{Scaling.Offsets} - \text{References}) \quad (2)\end{aligned}$$

which is a matrix multiply onto the input vector, accumulated onto an initial vector, in actual fact a primitive function call of the SNAP-64 API.

This error vector is then compared with a maximum allowed error vector, triggering the Disruption Alarm if exceeded, driving one of the digital output lines already used in PCS to inhibit the feedback loops.

The proportional-integral-derivative terms of the PCS controller were emulated by a state space controller, imposing a finite bandwidth on the derivative term to allow this transformation to be made. The PCS expression for the controller :

$$Y = \mathbf{M}.[\mathbf{P}.\text{diag}(1/\tau_P) + \mathbf{I}.\text{diag}(1/s\tau_I\tau_P) + \mathbf{D}.\text{diag}(s\tau_D/\tau_P)].\varepsilon \quad (3)$$

was replaced by the standard discrete-time controller form:

$$\begin{aligned}X_{n+1} &= \mathbf{A}_{kd} . X_n + \mathbf{B}_{kd} . \varepsilon_n\\ Y_n &= \mathbf{C}_{kd} . X_n + \mathbf{D}_{kd} . \varepsilon_n \quad (4)\end{aligned}$$

This choice was made deliberately so that the migration from PCS algorithms to state space control algorithms was already achieved, even when emulating the PCS algorithm.

Finally, the part of the feedforward voltages which is linearly related to raw inputs, implemented in the M-Matrix for coil resistance compensation, was created as an extension of the A-Matrix, to reduce the number of function call operations.

The architecture of the SNAP-64 imposes a minimum number of function calls for efficiency and the re-writing of the PCS algorithm into a minimum number of primitives caused no problems. As mentioned in Section IV, the latency was reduced by executing the minimum number of operations on new data before declaring the outputs ready. Operations necessary for preparing for the next cycle were then completed at the end of the cycle. The first operations before the output data are ready comprise only 10 calls to SNAP-64 primitives.

The second phase, preparing for the subsequent cycle comprised 97 calls, accounting for the cycle time. Additional work for control was distributed over a variable number of cycles in this phase. This work comprises:

i) Getting the next increments for the staircase waveform generation.

ii) Checking the errors against the alarm thresholds.

iii) Checking the feedback gate times against cycle number.

The code segment for the first phase of operations is shown in Appendix A to illustrate the simplicity.

The initial implementation of the PCS emulation gave the following performance. The latency was measured at 84 μsec. The cycle time for cycles with no extra work was 65 μsec. The longest additional work in a cycle was 100 μsec. The average cycle time for PCS emulation was reduced to 88 μsec by care and attention.

VII. Discussion

The procurement phase of this equipment was discussed in some detail since such a mixture of high technologies is normally integrated by the client laboratory. TCV was already an operating device and no time was allowed for staff to devote their time wholly to the DPCS, an initial constraint on the internal acceptance of this project. The CRPP role was therefore limited to an initial product survey, helped by colleagues on other tokamaks, a draft specification of the project, a final specification of the project and progress chasing. In retrospect, there is no way in which the CRPP could have realised this project in the same time, even if three professionals had been dedicated to it.

In total, porting the PCS algorithms to the DPCS involved under 500 lines of IDL (Interactive Data Language) code for preparing the data and emulating the old discharges to verify the implementation of the algorithm off line. Added to this are 107 lines of rather repetitive C-code for the implementation of the inner loop calculations. This effort amounted to about 4 man weeks of work to be added to the 10 days of commissioning to yield a total of 30 days of continuous effort from delivery to first operation. This slight effort was due to the closeness of the acceptance test algorithm delivered by PSL to the final implementation needed for TCV.

Such an exercise always results in some lessons learned. From the point of view of PSL, such a one-off turn-key operation always carries a risk since an accurate estimation of the effort involved requires the solution to be implemented, impossible at the time of the contract. Delays incurred through equipment problems are unpredictable, mitigated by the ability of the CRPP to accept these delays, since there was already an operating PCS. These delays notwithstanding, the total execution time was considered to be excellent by the CRPP. On the downside, the end-user takes over a complex device without the intimate knowledge of its functioning which would have been naturally acquired when taking on responsibility for the integration.

VIII. Acknowledgments

The authors wish to acknowledge the support of Prof. M.Q. Tran and the TCV team in the development of this project and the TCV authors are grateful for the full support of the two commercial companies involved. This work was partially supported by the Swiss National Science Foundation.

IX. References

[1] H. Weisen, et.al., "Effect of plasma shape on confinement and MHD behaviour in TCV", Invited Paper, 24th EPS conference on Controlled Fusion and Plasma Physics, Berchtsgaden, 9–13th June 1997.

[2] P.F. Isoz, J.B. Lister and Ph. Marmillod, "A hybrid matrix multiplier for control of the TCV tokamak", Proc. 16th Symp. On Fusion Technology, pp.1264–1267, London, 3–7th September 1990.

[3] J.B. Lister, et.al., "The Control of TCV Plasmas", Fusion Technology, vol 31, number 7 (1997).

[4] The SIMD Numerical Array Processor (SNAP) is manufactured by HNC Software Inc., 5930 Cornerstone Court West, San Diego, CA 92121–3728.

[5] A. Favre, et.al., "Control of highly vertically unstable plasmas in TCV with internal coils and fast power supply", Proc. 19th Symp. On Fusion Technology, Lisbon, 16–20th September 1996.

Appendix A

Code segment of the first phase of the SNAP calculation, showing the C-calls to SNAP primitives allowing matrix and vector operations on the data.

```
EIF(MatrixMultiply( IdAMMat, IOC.IdAnalogueIn, IdAMOutVec
));
EIF(Add( IdAMOutVec, IdAMRefScaled, IdErrVec ));
EIF(MatrixMultiply( IdBkdMat, IdErrVec, IdTempNS ));
EIF(Add( IdTempNS, IdXdVec, IdXdVec ));
EIF(Multiply( IdXdVec, IdIntGateVec, IdXdVec ));
EIF(MatrixMultiply( IdCkdMat, IdXdVec, IOC.IdAnalogOut ));
EIF(MatrixMultiply( IdDkdMat, IdErrVec, IdTemp64 ));
EIF(Add( IdTemp64, IOC.IdAnalogOut, IOC.IdAnalogOut ));
EIF(Add( IOC.IdAnalogOut, IdOutFfVec, IOC.IdAnalogOut ));
EIF(Clip(IOC.IdAnalogOut,     IdOutClipLow,     IdOutClipHigh,
IOC.IdAnalogOut ));
```