Architecture and Commissioning of the TCV Distributed Feedback Control System

J. I. Paley, S. Coda, B. Duval, F. Felici, J-M. Moret and the TCV Team

Abstract–A new modular, digital, distributed feedback control system has been developed and installed to control the TCV plasma. With many more inputs and outputs, it provides the possibility to build control algorithms using far more information on the plasma state than previously possible as well as the ability to control many more actuators, including the multi-megawatt, multi-launcher electron cyclotron heating and current drive system. This paper provides an overview of the new control system, its integration into the TCV systems and its successful application to control the TCV plasma discharge.

I. Introduction

THE basic tokamak control problem consists of generating magnetic coil currents to sustain a pre-programmed, time evolving plasma equilibrium, which is described by the plasma position, shape, current and density. With the requirement to operate near the limits of plasma performance in order to maximize investment returns, this control problem has evolved to require the ability to neutralize performance-limiting instabilities, which tend to develop when operating close to these limits in, for example, plasma pressure.

Since its inception, TCV has relied on the hybrid control system [1] to provide control of the magnetic coil currents and a gas valve; using analogue matrix multiplication of signals, a PID controller and digital switching of the matrix coefficients during a plasma discharge. This system has proven to be extremely reliable, however it is limited in the number of input signals (128), feedback control outputs (20) and to linear PID control only. Until recently the electron cyclotron (EC) heating and current drive system (ECRH/ECCD) at TCV was driven by pre-programmed waveform generators, with the power levels and launcher angles entirely specified pre-shot and triggered at the appropriate time during the plasma discharge. In order to respond to changing plasma conditions, such as the development of neoclassical tearing mode instabilities (NTMs) and to attempt to neutralize these instabilities, the ECRH/ECCD system must operate in a feedback control loop.

The expansion of not only the actuator set, but also the capability to use many other diagnostics including soft x-ray and diamagnetic measurements in control algorithms, has motivated the development of new control hardware for TCV, in which digital, non-linear and procedural algorithms can be used. This paper describes the 'Système de Contrôle Distribué' (SCD) or distributed control system which has now been installed, integrated and successfully used to control TCV plasmas.

The first section in this paper provides a description of the actuator set available at TCV as well as the current hybrid control system. The paper will then go on to describe the architecture of the new distributed control system, its integration with TCV and the development of control algorithms. The final section describes the successful tests to control TCV plasmas.

II. TCV Actuators – Magnets & EC Systems

The TCV actuator set consists of magnetic coils, gas valves, gyrotrons and EC launchers. There are 16 magnetic coils which control the plasma shape and position (E and F coils), 2 Ohmic transformer circuits which drive the plasma current (formed by coils A, B, C & D), the toroidal magnetic field coils, the internal fast vertical position control coils and 4 gas valves. Fig. 1 shows the position of the poloidal magnetic field coils and EC launchers around the TCV vacuum vessel.

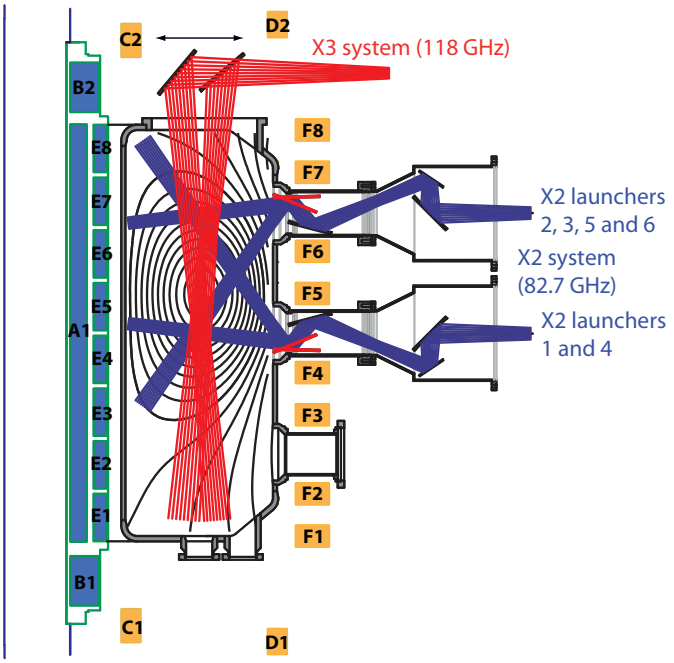

Fig. 1. Cross section of the TCV vacuum vessel, poloidal magnets and the EC launchers. Note the EC launchers are distributed around 4 toroidal locations. Each of the final launcher injection mirror angles can be controlled in real time. A typical plasma equilibrium is also shown.

For each magnetic coil circuit, an independent power supply generates the coil currents. The coil power supplies have various modes of operation, requiring either a voltage and/or current reference signal. In the voltage mode, it is necessary to provide a voltage reference for the power supply together with

Manuscript received May 27, 2010. This work was partly supported by the Swiss National Science Foundation.

The authors are with the Ecole Polytechnique Fédérale de Lausanne (EPFL), Centre de Recherches en Physique des Plasmas, Association Euratom-Confédération Suisse, CH-1015 Lausanne, Switzerland.

a current polarity switch signal, to request the power supply to change the direction of current in the coil. A set of internal coils driven by a fast power supply provide feedback control of the plasma vertical instability and the toroidal magnetic field is provided by the toroidal magnet system and power supply.

The TCV ECRH/ECCD system consists of 2 subsystems operating at the 2nd and 3rd Electron Cyclotron resonances in X-mode (X2 and X3) [2]. The 3MW X2 system consists of 6 diode gyrotrons with individual waveguides and launchers together with 2 independent power supplies (RHVPS). The injected EC power from each set of 3 gyrotrons may be specified by controlling the gyrotron cathode voltage provided by the RHVPS. The 1.5MW X3 EC system is provided by 1 RHVPS power supply, 3 triode gyrotrons with independent anode power supplies, waveguides and single injection launcher.

TABLE I. NUMBER OF CONTROL CHANNELS AND BANDWIDTH REQUIRED FOR EACH ACTUATOR

Actuator	Required bandwidth	Number of channels
A,B,C,D,E,F coils	5kHz	36 (18x2)
Internal coils	200kHz	1
Gas valves	100Hz	4
EC power	1-20kHz	6
EC launchers	20Hz	7

III. THE HYBRID CONTROL SYSTEM

The hybrid control system was designed to provide feedback control of the coil currents and plasma density. It uses measurements of the magnet coil currents together with in-vessel flux loops, magnetic probes and a far infrared laser interferometer fringe counter to achieve this. The hybrid controller is based upon matrix multiplication of signals and a PID controller. A detailed description may be found in the reference [1]. A simplified description follows, together with the Simulink representation in Fig. 6: The first 'A' matrix generates observables such as plasma current, vertical position and plasma density using linear combinations of the measurements. Error signals are generated by subtracting the observables from reference signals which originate from a waveform generator. A PID controller and second 'G' matrix multiplication generates the required command quantities, such as power supply voltages. The final 'M' matrix corrects for the mutual inductance between each coil. Finally, the feedback command signals are summed with feedforward signals from the waveform generator before being sent to the actuators. Further signals from the waveform generators are used for specifying the EC powers and launcher injection angles as well as the toroidal magnetic coil current, further gas valves and power supply currents/current polarity signals, which are required in certain modes of operation of the power supplies.

A controller to replace the hybrid system would be required to at least specify the voltage and current waveforms, in feedback, for each of the 18 shaping and Ohmic power supplies, as well as the 4 gas valves and 7 launcher injection angles together with 6 ECRH RHVPS and anode power supply voltages.

We would also like to use many of the existing TCV diagnostic systems to generate real time observers, for example the soft x-ray diagnostic could be used to detect magnetohydrodynamic instabilities such as the sawtooth instability. This diagnostic generates 64 analogue signals, far more than available inputs on the existing hybrid controller. These diagnostics are also physically distributed around the tokamak and it would be difficult to route all of the analogue signals into the hybrid controller.

One of the main difficulties in the TCV control problem is the vertical plasma position control, which for extremely shaped, elongated plasmas in TCV has stringent requirements on the control loop. The power supply for the internal fast vertical control coils can respond to voltage changes within <5μs, requiring a controller bandwidth ~200kHz. A recent multi-DSP based controller was developed specifically to provide this bandwidth for the control of the plasma vertical stability [3]. This control loop can be decoupled from the plasma shape and position control and therefore it is possible to use an independent controller.

IV. THE TCV DISTRIBUTED PLASMA CONTROL SYSTEM (SCD)

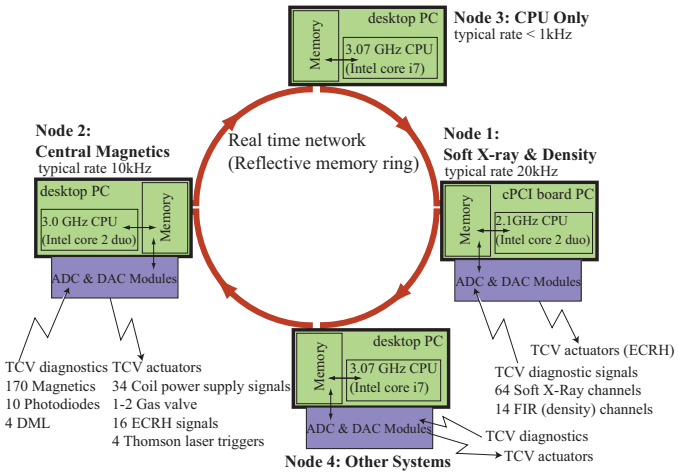

Fig. 2. Each node is connected to the real time network (Reflective memory). The various node configurations are shown together with the typical diagnostic and actuator systems to which they are connected.

TABLE II. CONTROL SYSTEM I/O COMPARISON. THESE ARE THE TOTAL CURRENTLY AVAILABLE INPUTS/OUTPUTS ACROSS ALL NODES IN THE NETWORK. AS THE SCD IS MODULAR, IT IS POSSIBLE TO ADD ADDITIONAL I/O AS REQUIRED

	Hybrid	SCD
Analogue inputs	128	384
Feedback outputs	20	144
Feedforward only outputs	84	NA
Digital outputs	0	384
Digital inputs/outputs	0	96

To provide feedback control over all of the actuator systems described above and to allow the addition of a large number of physically distributed diagnostics with large numbers of analogue signals into the control system, the TCV plasma

control system upgrade uses a modular network of real time PC nodes (RT nodes) linked by a real time network (Fig. 2). The expansion of controller inputs and outputs in comparison to the hybrid control system is described in Table II.

A. The Real Time Nodes

Each RT node is a 32bit Linux PC (currently Fedora 12), either embedded on a Compact-PCI (cPCI) module or as a desktop computer with Intel CPU, hard disk, memory, real time network card and Ethernet port for offline communication. During the real time process, all of the interrupts to the CPU are suspended, using the set of Linux kernel functions disable_irq(). The mlockall() function is also used to prevent memory paging. In this way, we can guarantee that hardware will not interrupt the RT algorithm, dedicating 100% of the CPU to the real time process. This does prevent the use of interrupts in the transfer of data to/from hardware, e.g. from the ADCs, but we can use direct memory access (DMA) transfers together with CPU polling instead of the typical interrupt on DMA complete. Alternative techniques, for example real time Linux kernel extensions and tailoring the interrupts to specific CPU cores could instead be used for specific applications. A brief specification of each RT node in the TCV network is also shown in Fig. 2. When connected to ADC and DAC hardware, the node is an acquisition & control node. Without ADC/DACs, it is a CPU only node. Both types will be described in the subsequent sections, after a description of the real time network.

B. The Real Time Network

The real time network linking each RT node is provided by GE reflective memory [4] (RFM) which provides a 128MB memory area that is shared across all the nodes. A fiber optic ring network links the RFM network cards (PCIe or PMC) in each node. Data written by one node to a memory address within this shared memory area will automatically appear at the same memory address within all the other after a very short delay. As the shared memory area onboard the RFM card is not accessible directly from Linux user space, the RFM drivers and API provide the ability to map the RFM memory space to a user space pointer, as well as functions to copy a memory block to/from the RFM (using a PIO or DMA transfer).

The shared memory provides a powerful and generic method of sharing data around the network, however care must be taken to ensure there are no collisions in reading and writing to the RFM network. There are many ways of achieving this, with each method having certain advantages and disadvantages. The following paragraph describes the current implementation for the SCD, which has been optimized to provide a generic interface, however it is of course possible to use a custom implementation if, for example, the performance is not suitable.

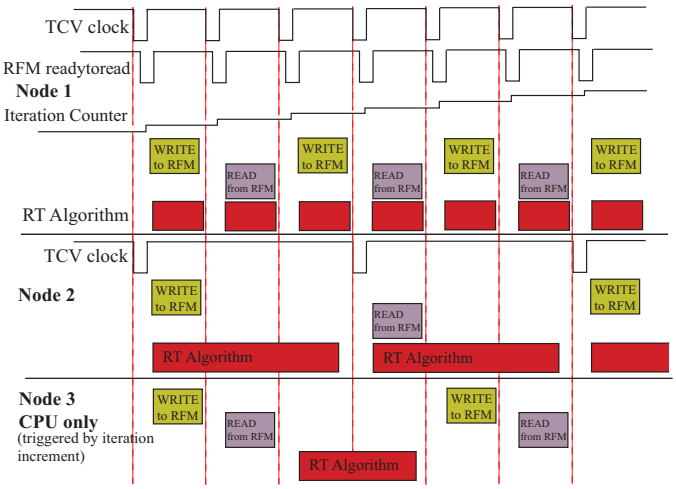

Fig. 3. Reflective memory network timing diagram for 3 nodes. One RT node is specified as the RFM control node (Node 1) and on the TCV clock, updates the RFM iteration counter together with the iteration number *readytoread* semaphore. The second node is triggered by the TCV clock, operating at 1/3 of the RFM control node rate such that the read and write cycles are synchronized across the network. As the CPU only node does not have an external clock, it is triggered by the incrementing iteration counter

The RFM is partitioned in a small control parameter section and large data area. Each node is assigned a separate section within the data area in which to write data. This immediately prevents the nodes from overwriting data areas outside their assigned write area. On each clock of the RFM master node (this is assigned by the operator, but is typically the central magnetics node), we alternate a read or write from/to the RFM using DMA. Every node in the network must be synchronized with the read and write iteration, such that all the nodes read (or write) at the same time, although nodes may skip iterations depending upon their differing controller rates. This requires certain restrictions on the control rate of each node and on the trigger times. Fig. 3 shows a timing diagram for 3 nodes in the network, including the iteration counter and semaphore.

The original Linux kernel drivers from GE were modified to allow DMA transfers to take place between the host PC memory and the RFM card without the use of interrupts, which are required to indicate when the data transfer is complete. Usually the DMA interrupt is required to signal to the CPU that the transfer is complete, at which point it may, for example begin to operate on the data. As we disable all of the interrupts, we instead initiate the DMA transfer and then simply allow sufficient time to elapse for the transfer to complete – this time is the (fastest) external TCV clock for the acquisition & control nodes. By ensuring that we do not transfer too much data, or operate with an external clock that is too fast, we can guarantee the data is transferred to/from the RFM by the next clock.

The CPU only nodes in the network require the RFM iteration number in the shared memory, to enable the node to trigger when the iteration number increments as they do not have an external clock. This also requires a semaphore to prevent collisions when reading/writing the iteration counter. At each TCV clock, the RFM master node increments the iteration counter as follows:

1) The *readytoread* semaphore is changed to false.
2) The iteration number is incremented.
3) The *readytoread* semaphore is changed to true.

The RFM has been driven at up to 20kHz, with each node writing 128x32bit data to the RFM and reading 3x128x32bit data. This corresponds to a throughput of at least 15MB/s. The limits in performance of the RFM in this configuration have not been fully explored, but this value is well below the manufacturer specifications. Also, due to the ADC/DAC DMA transfers that take place in the control cycle, competing for the PC bus resources it is unlikely we could achieve the maximum specification. Further investigation will take place in order to explore the limits.

C. *Acquisition & Control Nodes*

The acquisition & control nodes are built around the D-tAcq 250/500kHz ACQ196 6U compact PCI digitizers. These cards have 96 differential 16bit ADCs [5]. One or more acquisition cards are plugged into the slave slots in a cPCI crate together with optional analogue and digital output boards. A 16 channel, 16bit DAC rear transition module (D-tAcq RTM-AO16) slots into the rear of the cPCI crate, inline with an acquisition card to provide 16 DAC outputs and 32 digital input/outputs. Alternatively a full (slave) cPCI module with 32 DAC channels and 2 x 64 digital outputs can be used (D-tAcq AO32). At least one ACQ196 board must be in each cPCI crate to use the AO32s and each AO16 board requires one ACQ196. The TCV central magnetics node uses 2xACQ196, 2xAO16 and 2xAO32 in a 6U, 6 slot cPCI crate to provide 188 analogue inputs and 96 analogue outputs.

The cPCI embedded PC is slotted directly into the cPCI crate master slot, otherwise, a PCI bus extender is used to connect the crate to a standalone PC. The acquisition cards in the cPCI crate are then directly discovered by the PC BIOS at boot time. Communication with the acquisition cards through the D-tAcq provided Linux kernel driver then follows the typical Linux standards, as with any PCI peripheral.

The ADCs are triggered by the external TCV clocks and triggers. Immediately after ADC acquire, the data is transferred to host PC memory via a DMA transfer and the DAC command data is transferred from host PC memory to the acquisition board before being distributed to the DAC card. The DACs are immediately updated with the new values. The host CPU polls the local PC memory looking for a (ADC) DMA data transfer complete flag, at which point the ADC data is copied to a buffer, the RFM DMA transfer is initiated and the control algorithm is called. The buffer allows the ADC data to be stored for post shot analysis which is very useful for debugging the algorithm, as described in Section V. Once the RFM DMA transfer is initiated the data transfer will take place in the background without using the CPU. A timing diagram is shown in Fig. 4.

One ACQ196 board together with an RTM-AO16 can operate at up to 100kHz, with no RFM transfers and a simple control algorithm. Note the time between an ADC acquisition and the DAC update using data from that ADC acquisition is longer than 10µs. This is due to the latency in the data transfer of the DAC data from the host PC to the DAC output. An alternative mode of operation exists in which the DAC will be updated at the next TCV clock.

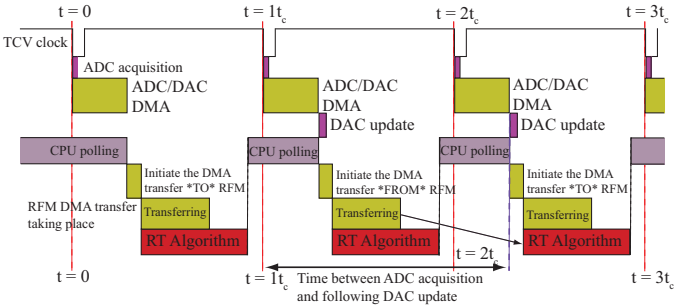

Fig. 4. System timing diagram showing the first 4 iterations in the shot sequence. When the RT node receives the ADC data DMA complete flag in local memory, it first initiates the RFM DMA transfer and then starts the control algorithm. The RFM DMA data transfer takes place in the background. When the algorithm is complete, the CPU starts to poll again for the DMA complete flag. The arrow indicates that the data obtained in the DMA transfer from the RFM can be used in the following RT algorithm.

D. *CPU only Nodes*

Several tokamak control problems require the ability to run CPU heavy, time consuming codes such as real time equilibrium reconstruction, ray tracing and fast Fourier transforms to detect MHD etc. To provide the capability to run such CPU intensive codes, a real time CPU-only node is dedicated to these tasks.

The CPU only node consists of a high performance PC together with a RFM network card. Again we suspend the interrupts during the real time process. As we have no external clock to this PC, the algorithm is triggered when the RFM iteration counter in the control parameter area of the RFM *increments*; i.e. we trigger on the rising edge of the iteration counter. A semaphore is also read both before and after reading the iteration number to ensure the iteration number itself was not written during the read phase. Although several (PIO mode) data reads take place in order to poll the RFM, the data is transferred each time in approximately 5µs which is much faster than the fastest RFM period of 100µs. In the worst case scenario, the *readytoread* semaphore could change immediately after it is read (before reading the iteration number) and there would be a delay of 5 polls, or <25us before the increment is detected. One of the RT nodes must be nominated as the RFM controller node as discussed previously to provide the iteration counter.

At this point a DMA transfer is initiated and the RFM data is transferred to host memory. At the next RFM iteration, the control algorithm is initiated.

When the algorithm is complete, we must wait until the next RFM write iteration before writing the results to the RT network – again guaranteeing the data is consistent as long as the DMA transfer does not take longer than the RFM controller period. This is summarized in the timing diagram of Fig. 5.

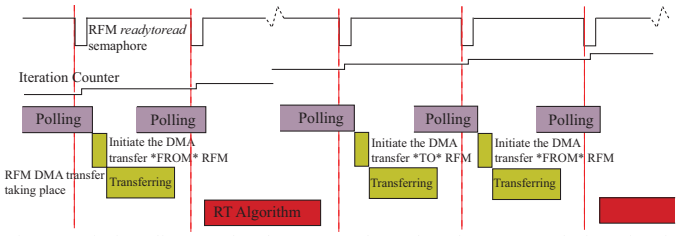

Fig. 5. Timing diagram for the CPU only node. The CPU node iteration is triggered by an increment in the RFM iteration counter, which the CPU polls together with the *readytoread* semaphore. When the iteration number increments to the pre-determined value, the RFM transfer is initiated. At the next iteration, the RT algorithm is executed. When the RT algorithm is complete, we wait until the RFM iteration increments to a write cycle, at which point we may write to the RFM. The cycle is then repeated.

The CPU only node is also used to display real time shot information on a control room display, when not being used in the real time control loop. The RFM is polled by a Matlab script and when a change in data is detected for a particular signal, the data is collected and plotted on a control room display.

V. Generating Feedback Control Algorithms

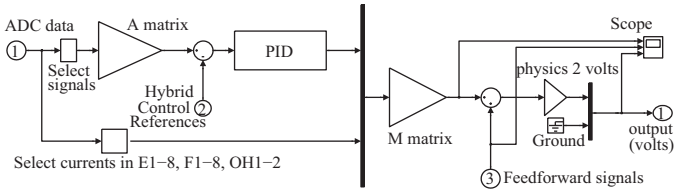

Fig. 6. Simulink diagram of the matrix operations and PID controller in the TCV Hybrid control algorithm. The Scope provides the capability to analyze various signals in the algorithm, either before or post-shot using the actual real time data acquired by the system and stored in the MdsPlus [7] tree.

The feedback plasma control algorithms are developed entirely in Simulink [6]. This provides a powerful environment in which to build control algorithms, using a multitude of prebuilt Simulink blocks including digital filters, PID controllers, matrices, switches, lookup tables, linear solvers and neural networks to name but a few. The interfaces and simulated environment including the ADCs/DACs and RFM network are provided in a TCV Simulink template and a library containing several useful blocks e.g. ECRH control and soft x-ray diagnostic has been developed. Fig. 6 shows the Simulink model of the hybrid control algorithm, which is used to replicate the analogue hybrid control system. The Real-Time Workshop Embedded Coder automatically generates real time C code for each RT node from the TCV Simulink model. The RT node code is then compiled as a 32bit Linux shared library (.so file) and distributed to the RT nodes. When the TCV shot cycle is progressing, an application on each node is executed and dynamically loads the shared library at runtime using the Linux functions dlopen() and dlsym().

The auto-generated Simulink C code contains 3 hooks which are called by the node application: rt_InitModel() to initialize the Simulink algorithm, rt_OneStep() which is called at each control iteration and rt_TermModel() to cleanup if necessary after the shot.

After the shot, the offline Simulink model on the development machine can be re-run with the actual ADC data acquired and stored by the real time machine. The signals inside the algorithm can then be analyzed using Simulink scopes or exported to the Matlab workspace. This provides an extremely useful debugging environment. Signals can also be attached to a memory block such that during the real time shot, the signal will be saved to a memory buffer after the algorithm is completed at each control iteration. The signal may then be immediately plotted and analyzed after the plasma shot.

VI. Integration with TCV

The SCD has been integrated within the TCV systems to allow parallel use of the new and old control systems. To this end, each of the input signals to the original hybrid controller is replicated and sent to the new SCD system. A set of summators provide the functionality to select between the controllers for individual actuators (however due to the mutual inductance corrections inside the M matrix, it is unlikely that we would control some of the shaping and Ohmic coils with both the hybrid and SCD systems during the same shot.) Fig. 7 shows a simplified schematic of the hybrid and SCD systems, showing the signal replications and summators.

When operating with the SCD controller, the fast vertical position control coil is controlled by either the hybrid or DSP system.

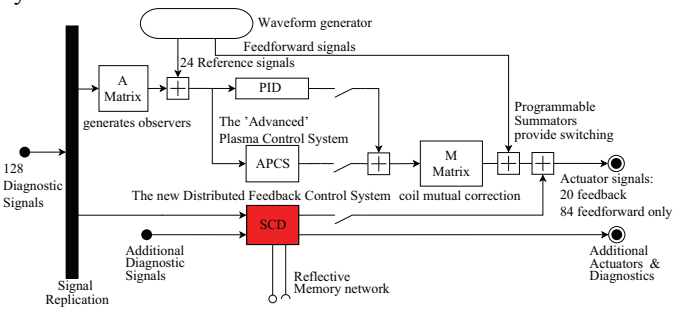

Fig. 7. Simplified schematic of the TCV central magnetic control systems. The ability to switch between controllers (between shots) is provided by programmable summators. All the hybrid input signals are replicated for the SCD controller together with the addition of other input signals. The outputs are passed through the summator, to provide switching between the control systems.

The SCD is controlled by a finite state machine allowing the system to be activated, initialized, executed etc through a set of scripts and functions, as shown in Fig. 8. The TCV state machine [8] operating the shot cycle provides the functionality to setup and execute many aspects of the tokamak systems including the state machines of individual systems. It initializes the acquisition systems, waveform generators, motor generator systems, power supplies etc, sets up the clocks and triggers, triggers the TCV shot and executes post-shot routines. The SCD state machine is controlled by the TCV state machine, in much the same way as other TCV systems.

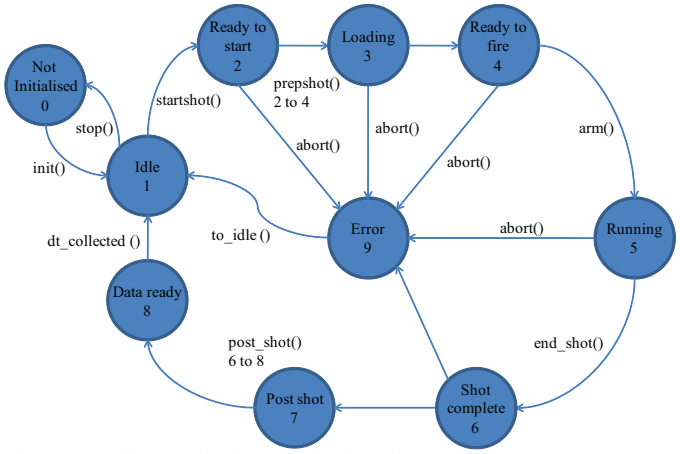

Fig. 8. State diagram for the TCV distributed control system.

The workflow from developing the algorithm to plasma shot is as follows:

1) Develop a Simulink model of the controller.
2) Test the algorithm using previously acquired data and/or models of the plasma response.
3) In the TCV shot cycle:
   a) Load the Simulink model (prepshot()).
   b) Initiate the build process (prepshot()).
   c) Compile and distribute the library to each active RT node (prepshot()).
   d) Setup the TCV clocks & triggers (prepshot()).
   e) Execute the real time application on each node (arm()).
   f) The TCV clocks and triggers are sent.
   g) PLASMA.
   h) After the shot, save the real time acquired ADC/DAC data and memory channels to an MdsPlus tree (post_shot()).

VII. Successful Control of the TCV Plasma Discharge

Individual nodes have been used for feedback control of plasma instabilities and plasma profiles in the past [9],[10]. For example, an extremum seeking algorithm was developed to maximize the period of the sawtooth instability, a plasma MHD instability, actuating on the EC launcher injection angle. The next step was to control the magnetic coil currents for the entire TCV plasma discharge.

Using the Simulink model of the hybrid control algorithm shown in Fig. 6, the controller was first run in parallel to the analogue hybrid control system in order to verify the outputs. This led to a successful attempt to control the entire TCV plasma discharge during the so-called standard shot – a basic plasma discharge that is created at the start of each experimental day. Each of the coil currents and gas valve was successfully controlled.

VIII. Discussion

The development and installation of the new distributed feedback control system has provided TCV with the capability to control all of its actuators, including magnets and ECRH/ECCD systems in real time. The integration of many more diagnostics into the control system provides far more information on the plasma state, from which algorithms to control plasma shape, instabilities and performance can be constructed. The ability to construct algorithms in the Simulink environment has proven to be an essential component in developing powerful control algorithms, without spending large amounts of time in the real time implementation.

Control of the TCV plasma discharge has already been demonstrated and the system will now be used in advanced, distributed feedback control experiments involving multiple diagnostic systems and the entire TCV actuator set.

Acknowledgment

The authors would like to thank the TCV team for their efforts and assistance in developing the new control system. In addition the authors acknowledge discussions with Sang-hee Hahn, B. Penaflor, K. Kurihara, R. Felton and W. Treutterer which concerned their experiences in the implementation of tokamak plasma control systems in several labs around the world.

This work was partly supported by the Swiss National Science Foundation.

References

[1] J..Lister et al., “The control of tokamak configuration variable plasmas”, *Fusion Technology* 32 (1997) 321
[2] T.P. Goodman and the TCV team, “Experience in integrated control of the multi-megawatt electron cyclotron heating system on the TCV tokamak: the first decade”, *Nucl. Fusion* 48 (2008) 054011
[3] N. Cruz et al., “Using APCS for plasma vertical control at TCV”, *this conference.*
[4] GE Intelligent Platforms http://www.ge-ip.com
[5] D-tacq Solutions Ltd http://www.d-tacq.com
[6] The MathWorks http://www.mathworks.co.uk
[7] MdsPlus http://www.mdsplus.org
[8] Y.R.Martin et al., “A new plant control software for the TCV tokamak”, ICALEPCS 2005, *Tenth International Conference on Accelerator and Large Experimental Physics Control Systems,* Geneva, October (2005).
[9] J.I. Paley et al., “Real time control of plasmas and ECRH systems on TCV”, *Nucl. Fusion* 49 (2009) 085017
[10] J.I. Paley et al., “From profile to sawtooth control: developing feedback control using ECRH/ECCD systems on the TCV tokamak”, *Plasma Phys. Control. Fusion* 51 (2009) 124041