UNIVERSITÀ DEGLI STUDI DI PADOVA

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

Università degli Studi di Padova

Centro Ricerche Fusione (CRF)

Università degli Studi di Napoli Federico II

JOINT RESEARCH DOCTORATE IN FUSION SCIENCE AND ENGINEERING

Cycle XXXVI

Integration of Open-source Frameworks in Industrial Plant Control Systems for Large Experimental Facilities ("Dottorato Industriale" - CRFX)

Coordinator: Prof. Gianmaria De Tommasi

Supervisor: Dr. Gabriele Manduchi

Supervisor: Prof. Paolo Bettini

Ph.D. Student: Giulio Martini

Padova, February 2024

To my family and friends, with love.

"Many people have the wrong idea of what constitutes true happiness. It is not attained through self-gratification, but through fidelity to a worthy purpose."
Helen Keller

Acknowledgements

I stand at the culmination of my Ph.D. journey, a path that has been as challenging as it has been rewarding. As I reflect on these years of growth and discovery, I am filled with profound gratitude towards those who have been pillars of support along the way.

First and foremost, I extend my deepest appreciation to my supervisors, Dr. Gabriele Manduchi and Prof. Paolo Bettini. Prof. Bettini, your insightful guidance has been fundamental in pointing out the best path. Dr. Gabriele, your expertise and constructive critiques have significantly enhanced the quality of my work. Your mentorship has shaped my doctoral study and my approach to scientific inquiry.

I would like to thank Dr. Cesare Taliercio, Prof. Adriano Luchetta, Modesto Moressa, Mauro Breda and all the colleagues at work who followed me throughout my Ph.D. You were always helpful and ready to give precious advice, blending professional respect with personal warmth.

I owe a huge debt of gratitude to my friends, both within and outside the academic world.

To my colleagues at the research centre - especially Luca, Alessandro and all the others - thank you for the collaborative spirit, the stimulating discussions, and the shared laughter during lunch and coffee breaks. Your company has made every working day a nurturing and innovative space.

My heartfelt thanks go to my closest friends, Riccardo, Alex, Adriano, Jappa and Carlotta. Your camaraderie, laughter, and constant presence have always rocked and rolled my life. Thank you for the endless adventures: the best is yet to come.

However, none of these accomplishments would have been possible without the support of my family. From my earliest memories, my parents have always been there, inspiring me to follow my passions and offering me a secure environment to explore them. I am deeply grateful to you, Mom and Dad. To my sister, Valeria, thank you for your constructive support and always believing in me, even from far away.

Last but certainly not least, to my partner, Sara. Your love, patience, and understanding have been my sanctuary. Your ability to listen and provide comfort during challenging times has been a guiding light. I am fortunate to have a lifetime ahead to express my gratitude to you, for it will surely take that long.

This thesis is not just a reflection of my effort but a testament to all these individuals' support, guidance, and love. Thank you.

Abstract

Modern large-scale experimental facilities, such as ones dedicated to magnetically confined nuclear fusion, often present demanding control requirements, including real-time control, extensive data acquisition and management.
Moreover, it is necessary to address the coordination of the many devices involved, which can consistently differ from each other and evolve rapidly due to the ever-changing software and hardware developments.
Furthermore, it is not uncommon to have both industrial off-the-shelf devices and custom-built ones working side by side, possibly connected to the same network. Even when involving industrial devices, communication between vendors is sometimes complicated, especially when dealing with legacy hardware.
Considering all these issues is paramount when choosing and developing a state-of-the-art Control and Data Acquisition System (CODAS) for the experiment to avoid introducing architecture inefficiencies and difficulties in system maintenance.
The three approaches commonly adopted consist of relying on industrial plant control systems (provided by well-known brands, such as Siemens), open-source ones (such as EPICS) or, in some cases, integrating these, which is the choice at ITER with CODAC Core System.
However, deciding which strategy is best is complex since each presents advantages and disadvantages.
Even when the choice falls on a combined solution, evaluating different factors affecting the final integration level between systems is still necessary.
Open-source frameworks often provide great flexibility in customization; in addition, they rarely require a paid licence for their use. On the other hand, they sometimes lack the scalability and robustness granted by commercial industrial systems, which can be expensive in return.
This work assesses the optimal integration between industrial systems and open-source frameworks in developing CODAS for a modern nuclear fusion experiment.
For this reason, it considers the CODAS of NBTF (the Neutral Beam Test Facility for ITER) and RFX-Mod2 (the largest Reversed Field Pinch experiment in the world) as case studies, conducting an in-depth comparison of their implementations and emphasizing the two approaches' differences, advantages and disadvantages.
Since NTBF has to comply with ITER guidelines, its CODAS relies on ITER's CODAC Core System. RFX-mod2, on the contrary, presents more freedom of choice for its CODAS (SIGMA) development but faces challenges related to preexisting hardware and software.

Focusing on the revamping strategies of SIGMA for RFX-mod2, this work analyses the architectural, hardware and software novelties introduced to replace obsolete solutions, meet new requirements and ensure a sustainable system for future years.

Sommario

Le moderne strutture sperimentali su larga scala, come quelle dedicate alla fusione nucleare confinata magneticamente, spesso presentano requisiti di controllo impegnativi, tra cui il controllo in tempo reale, l'acquisizione e la gestione estesa dei dati.
Inoltre, è necessario affrontare il coordinamento dei numerosi dispositivi coinvolti, che possono differire notevolmente l'uno dall'altro ed evolversi rapidamente a causa dei continui sviluppi software e hardware.
Inoltre, non è raro avere sia dispositivi industriali standard sia dispositivi realizzati su misura che lavorano fianco a fianco, possibilmente collegati alla stessa rete. Anche quando si tratta di dispositivi industriali, la comunicazione tra fornitori diversi è talvolta complicata, soprattutto quando si ha a che fare con hardware obsoleto.
Considerare tutti questi problemi è fondamentale quando si sceglie e si sviluppa un sistema di controllo e acquisizione dati (CODAS) all'avanguardia per l'esperimento, per evitare di introdurre inefficienze dell'architettura e difficoltà nella manutenzione del sistema.
I tre approcci comunemente adottati consistono nell'affidarsi a sistemi di controllo di impianti industriali (forniti da marchi noti, come Siemens), open source (come EPICS) o, in alcuni casi, integrarli, che è la scelta di ITER con il CODAC Core System.
Tuttavia, decidere quale sia la strategia migliore è complesso poiché ciascuna presenta vantaggi e svantaggi.
Anche quando la scelta ricade su una soluzione combinata, è comunque necessario valutare diversi fattori che influenzano il livello finale di integrazione tra i sistemi.
I framework open source offrono spesso una grande flessibilità nella personalizzazione; inoltre, raramente richiedono una licenza a pagamento per il loro utilizzo. D'altra parte, a volte non hanno la scalabilità e la robustezza garantite dai sistemi industriali commerciali, il cui utilizzo però può rivelarsi costoso.
Questo lavoro valuta l'integrazione ottimale tra sistemi industriali e framework open source nello sviluppo del CODAS per un moderno esperimento di fusione nucleare. Per questo motivo considera i CODAS di NBTF (la Neutral Beam Test Facility di ITER) e RFX-mod2 (il più grande esperimento Reversed Field Pinch al mondo) come casi di studio, conducendo un approfondito confronto delle loro implementazioni e sottolineando le differenze, i vantaggi e gli svantaggi dei due approcci.
Poiché NTBF deve conformarsi alle linee guida ITER, il suo CODAS si basa sul

CODAC Core System di ITER. RFX-mod2, al contrario, presenta maggiore libertà di scelta per il suo sviluppo CODAS (SIGMA), ma deve affrontare sfide legate all'hardware e al software preesistenti.
Concentrandosi sulle strategie di revamping di SIGMA per RFX-mod2, questo lavoro analizza le novità architetturali, hardware e software introdotte per sostituire soluzioni obsolete, soddisfare nuovi requisiti e garantire un sistema sostenibile per gli anni futuri.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF LISTINGS

ACRONYMS

ADC Analog-to-Digital Converter. 41–43, 67, 78–80, 111

AGPS Acceleration Grids Power Supply. 69

ALS Advanced Light Source. 52

API Application Programming Interface. 72, 77, 95, 96

APS Advanced Photon Source. 52

ASKAP Australian Square Kilometre Array Pathfinder. 52

AXI Advanced eXtensible Interface. 80

BOS Beam Operation State. 68

CA Channel Access. 51, 56, 64, 71, 72, 109

CAMAC Computer Automated Measurement and Control. 76, 81, 104, 110

CD Current Drive. 20

CERN Conseil Européen pour la Recherche Nucléaire. 49

CIC Cascaded Integrator-Comb. 80

CIN Central Interlock Network. 64

CIS Central Interlock System. 24, 61, 65, 67, 68, 70, 73

CODAC Control, Data Access and Communication. 4, 39, 55–57, 62, 64, 71, 72, 109, 110, 114, 116, 118, 122

CODAS Control and Data Acquisition System. 3–6, 24, 39–41, 59–62, 64, 66–68, 70–73, 75–77, 83, 103–105, 107, 108, 110, 114, 117, 118, 121, 122

COS Common Operating States. 68, 73

CP Communication Processor. 88, 89

CPCI CompactPCI. 76–78, 80, 81, 104, 110

CPU Central Processing Unit. 56, 77, 80, 81, 88, 89

cRIO CompactRIO. 71

CSN Central Safety Network. 64

CSS Control System Studio. 55, 56, 65

CSS Central Safety System. 24, 61, 67, 68, 73

DAC Digital-to-Analog Converter. 42, 43, 82

DAN Data Archiving Network. 62, 64, 71, 72, 83, 109, 110, 118

DB Data Block. 91, 93, 94, 126

DEMO DEMOnstration Power Plant. 17, 115, 117, 122

DMA Direct Memory Access. 78–80

DTT Divertor Tokamak Test facility. 46, 115, 117, 122

EAST Experimental Advanced Superconducting Tokamak. 46

ECRH Electron Cyclotron Resonance Heating. 15, 20

EPICS Experimental Physics and Industrial Control System. 4, 50–52, 55–57, 64, 65, 67, 71, 72, 94, 108–110, 112, 113, 116–118, 122

ESRF European Synchrotron Radiation Facility. 50

ESS European Spallation Source. 50

FB Function Block. 91, 92, 125, 127, 128

FC Function. 92, 93, 128

FIFO First In, First Out. 80

FPGA Field-Programmable Gate Array. 75, 78–81, 104, 111, 121

FSM Finite State Machine. 89, 91, 92, 115, 126–128

GDC Glow Discharge Cleaning. 89

GOS Global Operating States. 68, 73

HMI Human-Machine Interface. 48, 53, 62, 65, 127

HNB Heating Neutral Beam Injector. 22–24, 61, 67, 71–73

HPN High-Performance Network. 71, 73

I&C Instrumentation and Control. 24, 47, 55–57, 61, 67, 70, 73

ICRH Ion Cyclotron Resonance Heating. 15, 20

IOC Input Output Controller. 51, 56, 57, 72

ITER *"The Way"* in Latin, formerly International Thermonuclear Experimental Reactor. 4, 6, 7, 16–18, 20, 22–24, 39, 49, 52, 55–57, 59–62, 64, 67, 68, 71–73, 79, 82, 104, 109–117, 122

JET Joint European Torus. 43

KSTAR Korean Superconducting Tokamak Advanced Research. 52

LabVIEW Laboratory Virtual Instrument Engineering Workbench. 50

LACP Link Aggregation Control Protocol. 84

LH Lower Hybrid. 20

LHC Large Hadron Collider. 49

Linux MRG Linux Messaging, Realtime and Grid. 64, 104

LTM Long Term Maintenance. 68

MARTe Multi-threaded Application Real-Time executor. 43, 44, 81

MDSplus Model Driven System plus. 5, 6, 45, 46, 64–67, 71, 72, 76, 77, 81, 82, 94–97, 99–105, 109–114, 116–118, 121, 122, 125, 137, 138, 141, 144

MHD Magnetohydrodynamics. 25, 26, 28, 33

MITICA Megavolt ITER Injector & Concept Advancement. 6, 23, 24, 59, 60, 67–73, 107–118, 121, 122

MOP Mode Of Operation. 89, 94

NBI Neutral Beam Injector. 5, 15, 20

NBTF Neutral Beam Test Facility. 4, 6, 22–24, 46, 59, 60, 62, 73

NGA NextGenArchiver. 96, 97, 100, 117, 137

NI-NBI Negative Ion Neutral Beam Injection. 22

OB Organization Block. 91, 92, 125

OPC Open Platform Communications. 94, 112

OPC UA Open Platform Communications Unified Architecture. 54, 57, 77, 94, 95, 110, 117

OUC Open User Communication. 5, 88, 89, 91, 92

PCDH Plant Control Design Handbook. 62, 67, 73, 114

PCF Plant Controller, Fast. 55

PCIe Peripheral Component Interconnect Express. 71, 77, 82

PCS Plasma Control System. xv, 39, 41–43, 78, 104, 113

PDC Pulse Discharge Cleaning. 89

PID Proportional-Integral-Derivative Controller. 30, 78

PLC Programmable Logic Controller. 5, 6, 47, 48, 53–55, 62, 71, 76, 77, 83–89, 91, 93, 94, 103, 104, 110, 114, 115, 117, 121, 122, 125

POC Plant Other Controllers. 55

PON Plant Operation Network. 56, 64, 70, 71, 73, 83, 109

PSH Plant System Host. 55, 56

PSOS Plant System Operating States. 68, 73

PTP Precision Time Protocol. 64, 71, 109, 110

PV Process Variable. 94, 111, 113, 116

pvA pvAccess. 51, 56

PXI PCI eXtensions for Instrumentation. 62, 64

PXIe PCI eXtensions for Instrumentation Express. 71

RFP Reversed Field Pinch. xv, 4, 6, 7, 13, 24–28, 33–35, 89

RFX Reversed Field eXperiment. 28, 34, 35, 84, 89, 101, 102

RHEL Red Hat Enterprise Linux. 56, 64

RTS Real-Time System. 43

SCADA Supervisory Control And Data Acquisition. 5, 6, 39, 47–50, 52, 75–77, 83, 93–96, 104, 107, 108, 110, 115–117, 122

SCL Structured Control Language. 87, 88, 91

SDD Self-Description Data. 57, 116

SDN Synchronous Data Network. 71, 72, 82, 83, 109

SIGMA Sistema di Gestione, Monitoraggio ed Acquisizione Dati. 4, 5, 76, 85–89, 91, 93, 115, 122, 125, 127

SIL Safety Integrity Level. 64

SLAC SLAC National Accelerator Laboratory, originally the Stanford Linear Accelerator Center. 50, 52

SNL State Notation Language. 116

SNMP Simple Network Management Protocol. 84

SNR Signal-to-Noise Ratio. 80

SPIDER Source for Production of Ion of Deuterium Extracted from Rf plasma. 6, 23, 24, 59–67, 70, 72, 73, 107–118, 121, 122

STL Statement List. 87, 91

STM Short Term Maintenance. 68

TCN Time Communication Network. 62, 64, 65, 71, 72, 109

TCP-IP Transmission Control Protocol-Internet Protocol. 64, 70, 88, 109

TCS Conditioning State. 68

TCV Tokamak à Configuration Variable. 46

UDP User Datagram Protocol. 71, 82, 104, 109

UDT User Defined Data Type. 91, 125

VLAN Virtual Local Area Network. 82–84, 108–110

VME Versa Module Eurocard. 77, 78, 81

WinCC OA WinCC Open Architecture. xvi, 5, 6, 50, 52–54, 76, 77, 94–98, 100, 102–106, 111, 112, 115–118, 122, 125, 137, 138

GLOSSARY

MARTe1 First version of the MARTe framework. 44

MARTe2 Second version of the MARTe framework. 5, 44, 64–66, 71, 72, 76, 77, 80–82, 104, 105, 109, 112–114, 117, 118, 121

RFX-mod First enhancement of RFX (Reversed Field eXperiment). 7, 27–30, 33–35, 75, 76, 78, 80–82, 84, 85, 89, 93, 94, 104, 105

RFX-mod2 Second enhancement of RFX (Reversed Field eXperiment). 4–6, 34, 35, 46, 75–78, 80–86, 103–105, 107–118, 121, 122

Part I

Prologue

CHAPTER 1

INTRODUCTION

1.1 Challenges and Strategies in Control and Data Acquisition for Nuclear Fusion Research

In modern experimental facilities, particularly those focusing on the cutting-edge field of nuclear fusion via magnetic confinement, there is an essential demand for highly sophisticated control systems. These systems are critical for managing and maintaining the delicate balance required in fusion experiments. They need to provide real-time control and effectively handle the complexities of vast data acquisition and management.

The challenge is further compounded when dealing with diverse devices within these experimental setups. Each device has its unique specifications and operational requirements, and they are constantly evolving. Rapid advancements in software and hardware technologies drive this evolution.

Moreover, the mixture of readily available industrial devices with bespoke, custom-built solutions within a single network adds another layer of complexity. This situation is even more challenging when ensuring effective communication between different vendors, a task that becomes particularly intricate when older, legacy hardware systems are involved.

A critical aspect of managing these complex systems lies in selecting and developing an advanced CODAS (Control and Data Acquisition System). The design of

such a system must be meticulous to avoid unintentional design inefficiencies and to ensure ease of maintenance and adaptability for future advancements.

In addressing these challenges, the study explores three prevalent approaches in this intricate field. The first involves relying on industrial control systems from established brands, such as Siemens, known for their reliability and robustness. The second approach leans towards open-source frameworks, like EPICS, praised for their adaptability and the absence of licensing fees. The third hybrid approach combines elements of both and has been notably implemented in the ITER (*"The Way"* in Latin, formerly International Thermonuclear Experimental Reactor)'s CODAC (Control, Data Access and Communication) Core System. However, determining the most effective strategy is not a simple task. Each method brings its unique set of benefits and drawbacks. When a combined solution is considered, an in-depth evaluation of various factors influencing the integration level between the systems is crucial.

While offering significant flexibility and cost savings, open-source frameworks occasionally need to improve in scalability and robustness compared to their commercial counterparts, which, although more reliable, come with higher costs.

1.2 Thesis Contribution

This research aims to thoroughly analyse the optimal balance between industrial systems and open-source frameworks in the development phase of CODAS for contemporary nuclear fusion experiments. This investigation thoroughly examines the CODAS implementations at two leading-edge facilities: the NBTF (Neutral Beam Test Facility) of ITER and RFX-mod2, the world's largest RFP (Reversed Field Pinch) experiment. The analysis compares their systems, emphasising each approach's advantages and limitations.

The NBTF operates within a unique context, adhering to the stringent guidelines of ITER. As such, its CODAS is heavily reliant on ITER's CODAC Core System. On the other hand, RFX-mod2, at the forefront of experimental innovation, enjoys greater autonomy in developing its CODAS system, SIGMA (Sistema di Gestione, Monitoraggio ed Acquisizione Dati). However, this autonomy brings challenges, particularly in integrating and updating pre-existing hardware and software systems.

A significant part of this study explores the revamping strategies employed in the SIGMA system (RFX-mod2's CODAS). This more in-depth breakdown is motivated by the fact that, despite participating in the development of the CODAS for the NBI experiments, I dedicated most of my work to renewing SIGMA.
SIGMA is undergoing extensive innovations across all its sectors. The overarching goal is to create a robust and sustainable system which will meet the current demands and be adaptable and resilient enough to evolve with future technological advancements.
These enhancements span various areas, including Data Acquisition, Fast Control, Plant Control, Plant Supervision, and System Coordination. The new features of the RFX-mod2 CODAS encompass several key aspects:

- Real-time Plasma Control: This involves integrating data acquisition with real-time frameworks in the software and incorporating new hardware solutions like Zynq. It utilises MDSplus (Model Driven System plus) [1] and MARTe2 for adequate control.

- Data Acquisition: Development includes custom RedPitaya solutions and upgrades to the existing CPCI-based systems for enhanced data handling.

- Plant Control: New Programmable Logic Controllers (PLCs) are being introduced as plant system controllers

- Plant Supervision: The legacy SCADA (Supervisory Control And Data Acquisition) system will be replaced with WinCC OA (WinCC Open Architecture). This system ensures integrated communication with MDSplus.

In particular, a deeper analysis is dedicated to the development of the new SIGMA PLC software (called *Scheduler*) and of the custom backend allowing for direct communication between MDSplus and WinCC OA, being these the specific areas of development to which I majorly contributed during my PhD research activity.
The new *Scheduler* PLC software constitutes an innovative solution, combining modern aspects such as OUC communication and "object-oriented" PLC programming while maintaining backwards compatibility with its previous version running on older devices.
The new WinCC OA-MDSplus backend is the first direct integration of these two powerful frameworks, allowing for direct data acquisition into MDSplus of signals related to slow control and plant supervision.

1.3 Thesis Structure

The subsequent sections of this work are structured as follows:

- Chapter 2 (*Magnetically Confined Nuclear Fusion*) begins by addressing the pressing issue of transitioning away from fossil fuels. It then delves into the fundamental principles of nuclear fusion reactions and introduces fusion experiments, focusing on magnetically confined fusion. Specific attention is given to tokamak and RFP configurations, exemplified respectively by ITER and RFX-mod2.

- Chapter 3 (*The Control and Data Acquisition System (CODAS) for Fusion Experiments*) presents an overview of CODAS structure in modern fusion experiments, highlighting its key features and functionalities. Particular focus is on Data Acquisition, Plasma and Fast Control, Plant and Slow Control Systems, and SCADA and Plant Supervision systems. It then briefly describes the implementations of CODAS in existing fusion experiments.

- Chapter 4 (*SPIDER and MITICA CODAS - A Detailed Analysis*) provides an in-depth description of the development of CODAS for the NBTF experiments, SPIDER (Source for Production of Ion of Deuterium Extracted from Rf plasma) and MITICA (Megavolt ITER Injector & Concept Advancement).

- Chapter 5 (*CODAS Upgrade in RFX-mod2*) outlines the roadmap for the upgrade of RFX-mod2. It details the challenges faced and the innovative solutions being introduced. Particular attention is paid to the redevelopment of the Slow Control, Plant Supervision, and SCADA systems, especially the new PLC state machine software (Scheduler) for Plant Supervision and the development of a software backend that integrates WinCC OA (the new SCADA system) with MDSplus (the Data Acquisition framework used at RFX-mod2).

- Chapter 6 (*Discussion: A Comparative Overview of CODAS Strategies for SPIDER, MITICA and RFX-mod2*) performs a comparative analysis of CODAS implementations in SPIDER, MITICA and RFX-mod2, discussing their strengths and weaknesses to outline the best development strategies.

- Chapter 7 (*Conclusions*) offers concluding thoughts and reflections on the project, summarising the essential findings and implications of the work.

CHAPTER 2

MAGNETICALLY CONFINED NUCLEAR FUSION

This chapter explores the critical role of magnetically confined nuclear fusion in addressing the growing challenges of energy production and environmental sustainability. The diminishing reserves of conventional fossil fuels and the pressing imperative to curtail greenhouse gas emissions compel the exploration of alternative energy sources, with nuclear fusion emerging as a prominent contender. Section 2.1 sets the stage by discussing the broader energy problem, while Section 2.2 examines the specifics of the nuclear fusion process. Further, Section 2.3 offers an in-depth analysis of various magnetically confined nuclear fusion experiments, particularly focusing on tokamak and RFP configurations and highlighting notable examples like ITER and RFX-mod. This chapter aims to comprehensively understand the complexities, advancements, and potential implications of nuclear fusion technology in the context of global energy demands and environmental considerations.

2.1 The Energy Problem

An ever-growing energy demand underpins modern society's relentless drive for development. Historically, fossil fuels, such as petroleum, coal, and natural gas, have been exploited to meet this demand. However, their diminishing availability, coupled with the detrimental environmental impact of greenhouse gas emissions, mainly

CO_2, has compelled us to seek alternative energy sources. The consequences of the increased concentration of CO_2 in the atmosphere manifest as global warming, with CO_2 trapping Earth's infrared radiation. This phenomenon has shown a clear correlation with a rising global temperature, as shown in Figure 2.1 [2].

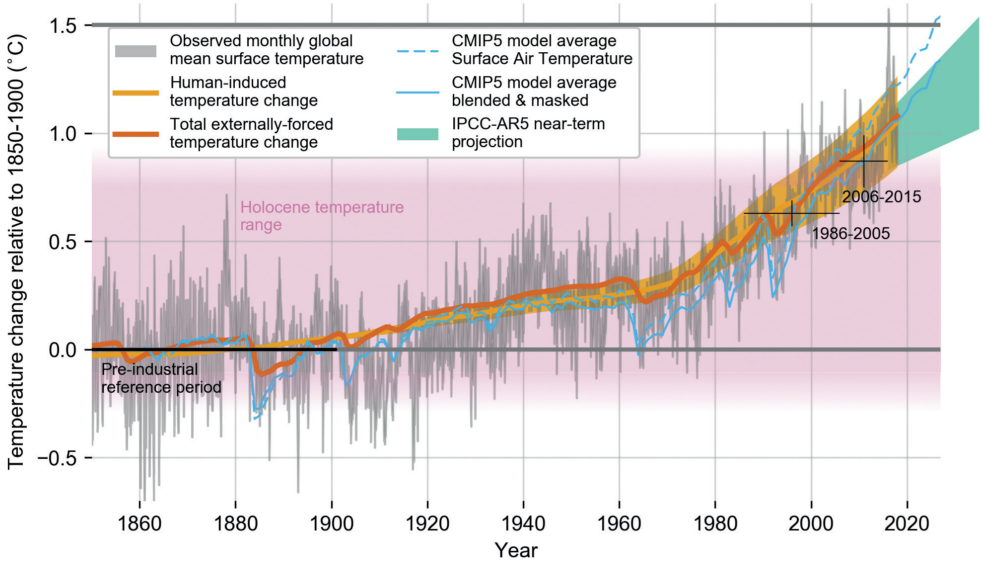


Figure 2.1: Evolution of global mean surface temperature (GMST) throughout instrumental observations.

Amidst the global urgency to address these challenges, spurred further by rapid population growth, industrialisation, and technological advancements, there has been a pronounced shift towards harnessing renewable energy sources. Solar, wind, hydropower, geothermal, and nuclear power cater to current needs and are critical for a sustainable future. This energy transition requires carefully intertwining energy security, economic growth, and environmental conservation.

The looming threat of climate change and its ramifications have bolstered international commitments to reshape energy paradigms. The European Union's ambitious Energy Roadmap is a testament to this, targeting an 80% reduction in energy-related greenhouse gas emissions by 2050 [3]. Achieving this demands an innovative energy portfolio, with pivotal roles played by decarbonised power sources, including those with Carbon Capture and Storage (CCS).

Nuclear energy, especially nuclear fusion, stands out as an up-and-coming contender. Thermonuclear fusion, particularly the tokamak concept, has garnered sig-

nificant interest due to its potential as an abundant, continuous, and clean energy source. Alongside fusion, the advancements in nuclear fission through generation III+ and IV reactors underscore the potential of atomic power in satiating global energy demands while mitigating environmental degradation [4].

In sum, as we grapple with escalating energy demands and the imperatives of environmental preservation, a harmonised approach employing both renewable and novel energy solutions seems pivotal. This amalgamation might chart the path towards a sustainable energy future.

2.2 Nuclear Fusion

Nuclear fusion is a nuclear reaction where two or more atomic nuclei combine to create a heavier element. This process releases energy because of the mass disparity (mass defect) between the initial and resulting nuclei. The change in the binding energies of the nuclei before and after the reaction leads to this mass defect. For fusion to occur, two positively charged nuclei must overcome their mutual Coulomb repulsion.

In the sun, the proton-proton fusion chain process is made possible due to the extremely high core density maintained by gravitational forces. However, replicating this process on Earth is unfeasible since we cannot achieve the required densities. The most practical fusion reaction for terrestrial use involves the interaction between the hydrogen isotopes deuterium and tritium. These isotopes are preferred because they have a larger fusion cross-section than other potential reactions. The reactions that occur at a rate high enough to compensate for the limited fuel density, which is typical in magnetic confinement devices, are:

$$^2D + {}^3T \rightarrow {}^4He + {}^1n + 17.6\,\text{MeV} \tag{2.1}$$

$$^2D + {}^2D \rightarrow {}^3He + {}^1n + 3.27\,\text{MeV} \tag{2.2}$$

$$^2D + {}^2T \rightarrow {}^3T + {}^1H + 4.03\,\text{MeV} \tag{2.3}$$

$$^2D + {}^3He \rightarrow {}^4He + {}^1H + 18.3\,\text{MeV} \tag{2.4}$$

In line with the momentum conservation principle, energy is liberated as kinetic energy in the resulting particles of the reaction. It is distributed among them inversely proportional to their respective masses. The likelihood of a fusion reaction occurring within a given interaction area between reactants is characterised as the

nuclear cross-section. Figure 2.2 illustrates the cross-sections of these previously mentioned reactions compared to the relative velocity of the two reacting nuclei [5].

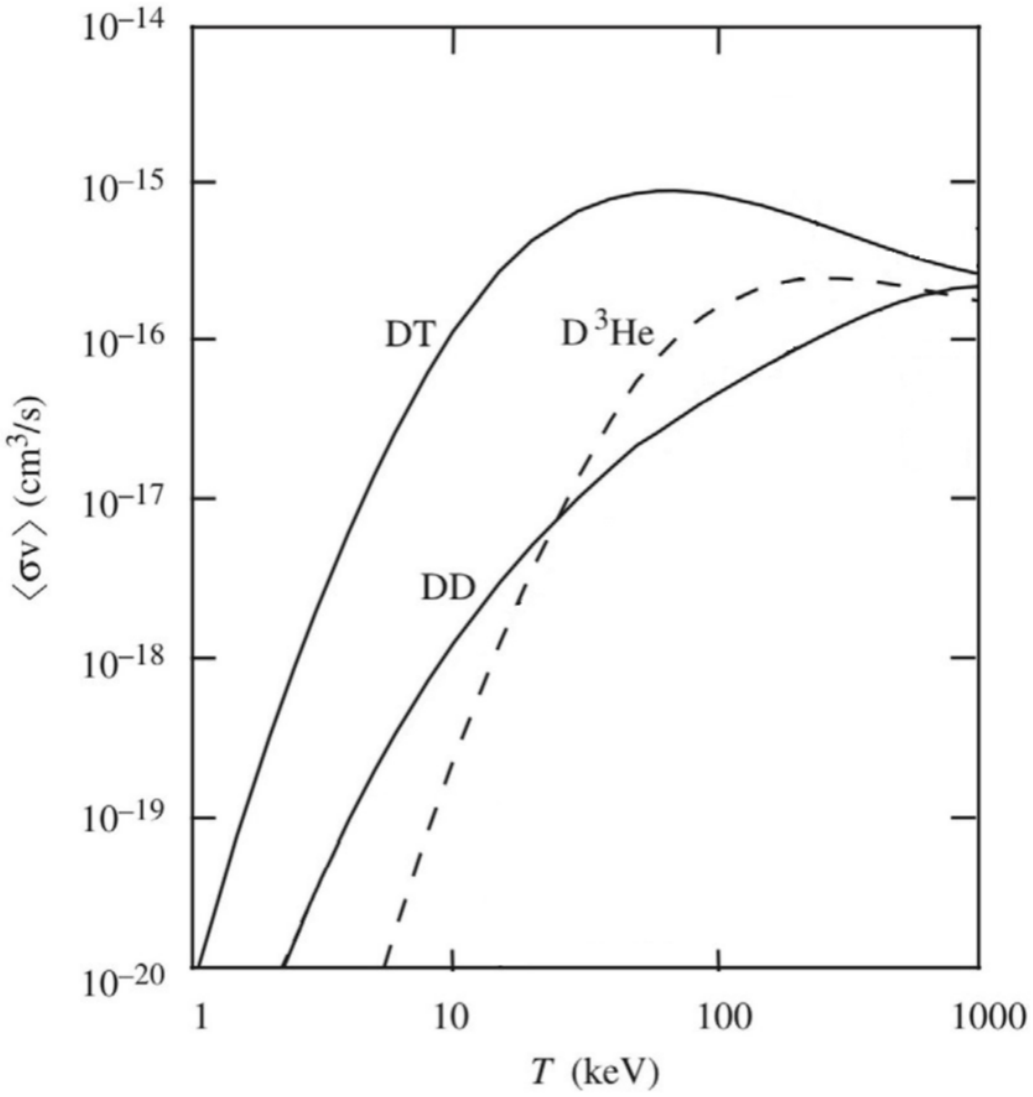


Figure 2.2: Cross-sections of fusion reactions $D-T$, $D-D$ and $D-3He$. The $D-D$ curve represents the sum of the cross-sections of the two $D-D$ reactions.

Technological constraints dictate that we can only effectively confine a substantial plasma volume for an adequate duration at temperatures below 100 keV. Under such conditions, the deuterium-tritium mixture emerges as the most reactive fuel, exhibiting the highest cross-section at lower temperatures. The requisite average temperature to generate a substantial number of fusion reactions in a fusion reactor is approximately 10 keV [6], even though the Deuterium-Tritium curve peaks around

a particle energy of 100 keV. At this temperature, sufficient high-energy ions are within the particle velocity distribution's high-energy tail and capable of engaging in fusion. Consequently, present research predominantly centres on the Deuterium-Tritium reaction.

Achieving a positive energy balance is contingent upon facilitating the reaction of fuel particles before their energy dissipation. In this context, particles must conserve energy and remain within the reaction region for an adequate duration. More precisely, the product of this time and the density of reacting particles must reach a sufficient threshold. The primary method for delivering the necessary energy is by elevating the temperature of the Deuterium-Tritium fuel to a level where the thermal velocity of the nuclei is sufficient to trigger the requisite reactions. This mode of inducing fusion is termed thermonuclear fusion.

The concept of ignition entails self-sustaining plasma combustion, where the energy for heating is exclusively derived from fusion reactions, with no external systems involved. The *Lawson criterion* articulates the conditions necessary to achieve ignition, which demand that temperature (T), density (n), and energy confinement time (τ_E) satisfy the following relation:

$$nT\tau_E > 10^{21}\,\mathrm{keVs/m^3}.$$

Figure 2.3 graphically depicts the ignition curve as a function of the triple product parameters.

The primary challenge in nuclear fusion lies in confining the intensely heated plasma, composed of ionised Deuterium and Tritium gases, to sustain fusion reactions. Given the extreme temperatures involved, no known materials can endure them. There are two primary methods for plasma confinement, both of which circumvent the materials issue:

1. **Inertial Confinement:** In this approach, a small amount of frozen Deuterium and Tritium fuel is rapidly imploded by a focused barrage of powerful laser beams. The inertia of the fuel sustains compression for a brief period (on the order of nanoseconds), creating conditions for a specific number of fusion reactions before the explosion [7].

2. **Magnetic Confinement:** Plasma is contained using externally generated

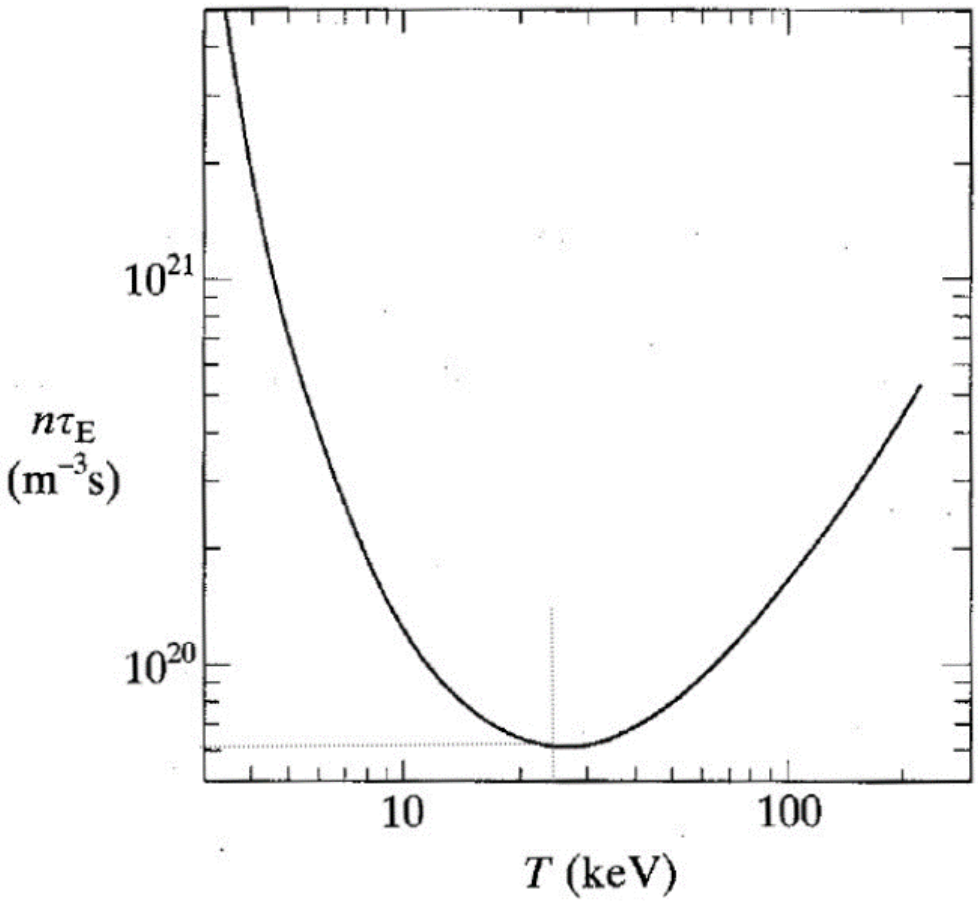


Figure 2.3: The required value of $n\tau_E$ for ignition as a function of the temperature T.

magnetic fields, preventing contact with the walls of the reactor's vacuum vessel. This confinement relies on the charged nature of the plasma, enabling interaction with the externally imposed magnetic fields.

This PhD work focuses on the control challenges of magnetic confinement fusion devices.

2.3 Magnetically Confined Nuclear Fusion Experiments

Among the various proposed solutions for magnetically confined fusion devices, the torus has emerged as the topologically advantageous choice. The torus is the only compact, connected, and orientable manifold where defining a continuous vector field is possible without encountering critical points.

In this context, the field configurations are comprised of a combination of magnetic toroidal and poloidal components. Figure 2.4 illustrates a reference system in pseudo-toroidal coordinates (r, θ, φ), where:

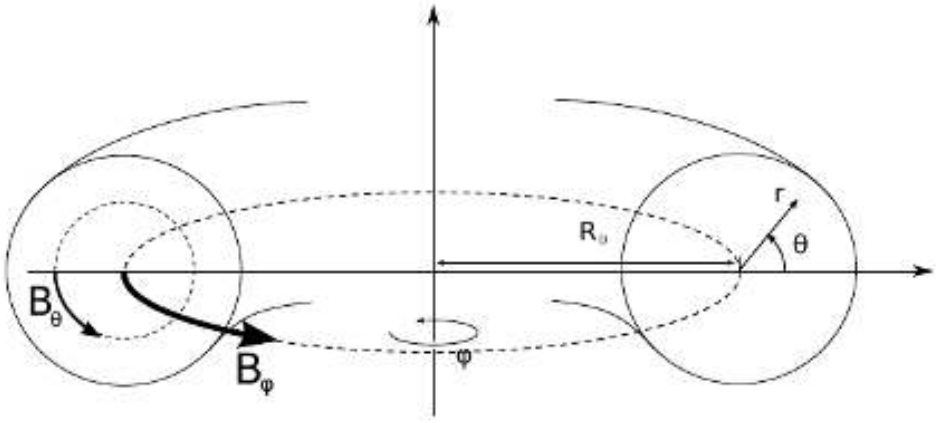


Figure 2.4: Toroidal coordinates reference system.

- r = radial distance from the central circular axis,
- θ = poloidal angle,
- φ = toroidal angle.

Magnetic confinement machines, depending on their specific characteristics, fall into three primary categories:

- **Tokamak:** In a tokamak, both the toroidal and poloidal magnetic field components are present. However, there is a clear toroidal predominance, and these magnetic fields primarily stabilise the plasma. This configuration is characteristic of devices like the tokamak, where the toroidal field plays a significant role.
- **RFP (Reversed Field Pinch):** In an RFP, the toroidal magnetic component is roughly equal in strength to the poloidal component. The toroidal field is mainly self-generated by the plasma and is concentrated in its core, often exhibiting a pinching effect. As the field extends towards the boundary, it gradually decreases and changes its sign. Hence the name "reversed field."
- **Stellarator:** The stellarator achieves a helical structure in the magnetic field without relying on plasma current from external sources. Instead, this configuration involves a complex arrangement of non-planar coils to create the required magnetic field geometry.

These three categories represent different approaches to achieving magnetic confinement in fusion research, each with advantages and challenges.

2.3.1 Tokamak Configuration

In tokamak systems, the toroidal magnetic field B_φ is generated by winding a toroidal solenoid around the vacuum chamber. In contrast, the poloidal field B_φ is established by the toroidal current I_p flowing within the plasma. This current is induced by solenoids linked to the plasma ring, often combined with a central coil, as shown in Figure 2.5. Notably, this current also contributes to plasma heating through the Joule effect, known as Ohmic heating, where the power $P = \eta J^2$, with η representing plasma resistivity.

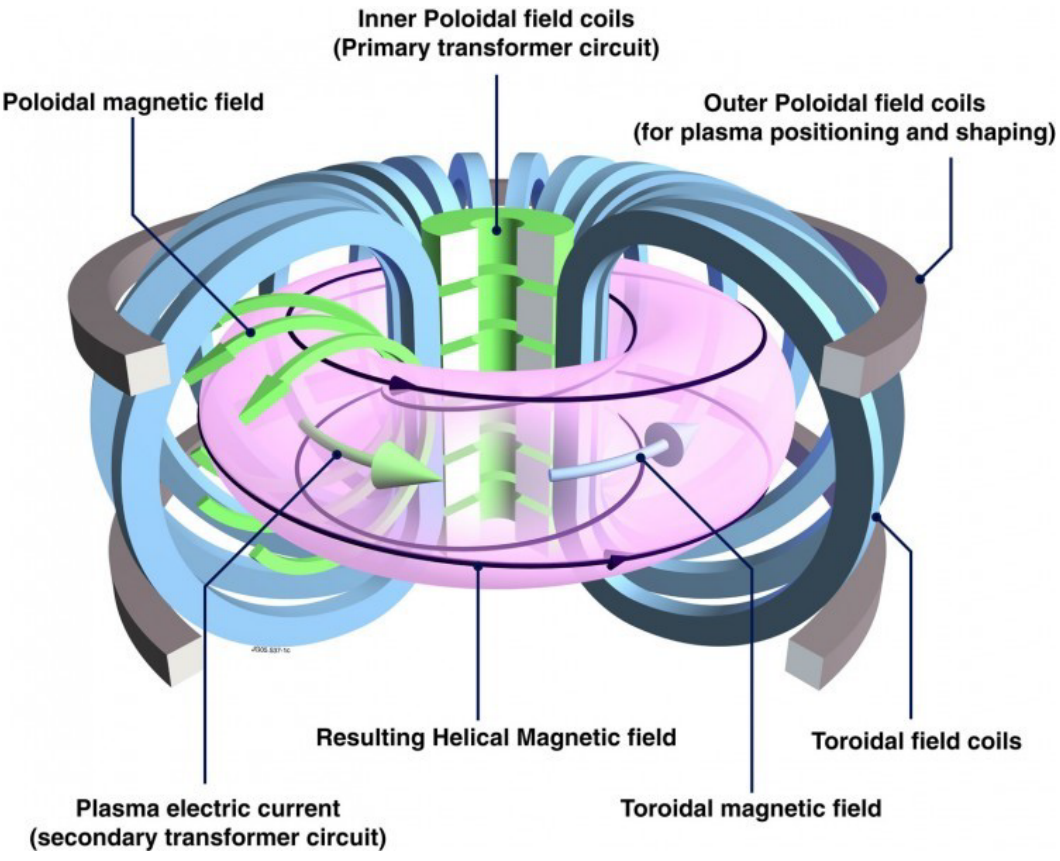


Figure 2.5: Conceptual design of a tokamak, showing all the main coil systems.

A crucial parameter for assessing the ability to sustain plasma at a given pressure p with a specific current is defined as follows:

$$\beta_p = \frac{\langle p \rangle}{\frac{B_\theta(\alpha)^2}{2\mu_0}}.$$

Here, $\langle p \rangle$ denotes the average plasma pressure within a poloidal section. β_p measures how efficiently the magnetic configuration balances the kinetic pressure exerted by the plasma against the magnetic pressure required to confine it. Typically, β_p falls within 1 to 3, with higher values leading to plasma instability. Another critical factor in determining the strengths of the magnetic configuration is the *"safety factor"* denoted as q:

$$q(r) = \frac{RB_\varphi(r)}{R_0 B_\theta(r)}.$$

As elaborated in subsequent sections, the radial profile of the safety factor q significantly impacts overall plasma stability. Specifically, a stable plasma configuration requires the $q(r)$ function to remain strictly monotonous. Furthermore, the value of q at the plasma edge $q(a)$ must exceed 3 to ensure configuration stability.

In summary, once the toroidal field is established, the maximum attainable current is governed by the $q(a)$ limit. The β limit sets that current level's maximum permissible plasma pressure for that current level. Figure 2.6 illustrates that in the tokamak configuration, maintaining the monotonous nature of the safety factor is guaranteed by having a high ratio of B_φ/B_θ. However, this high ratio necessitates keeping the plasma current below a certain threshold, which, in turn, places limits on Ohmic heating. As a result, alternative heating methods such as NBIs and radio-frequency emissions resonating with cyclotron frequencies of electrons or ions (ECRH, ICRH), among others, are needed to supplement the heating process.

In tokamak experiments, a notable reduction in energy loss has been observed once the heating power surpasses a certain critical level, resulting in an increased plasma confinement time τ_E. The plasma states above and below this critical threshold are termed *H-mode* and *L-mode*, respectively [8]. An edge pedestal distinguishes the H-mode, a narrow region at the plasma edge where significantly sharper gradients exist, indicative of an edge transport barrier. Empirical formulas govern the shift between these plasma modes and their respective confinement times [9].
To maintain a highly pure plasma environment and minimise dilution and radiation losses in the tokamak, creating a high vacuum is crucial and ensuring the plasma remains distinct from the vacuum vessel walls. One approach to achieve this is through a limiter configuration, where the plasma boundary is defined by a material

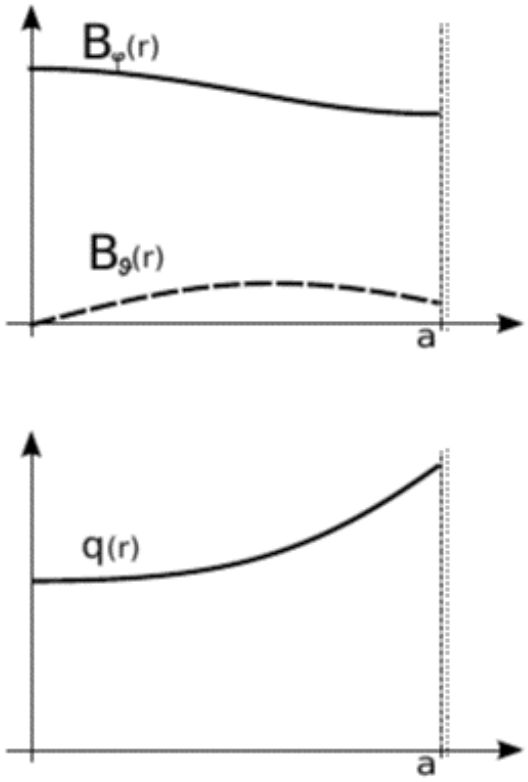


Figure 2.6: Profiles of B_φ and B_θ magnetic fields from the core to the chamber edge in tokamak toroidal confinement devices.

target that intersects the magnetic field lines. However, this method has become less prevalent with the emergence of the divertor configuration, which has shown superior effectiveness in maintaining optimal confinement and purity. In the divertor setup, modifications to the magnetic field are employed to keep particles from the vacuum vessel, creating what is known as the *X-point*. This arrangement segregates regions with closed magnetic surfaces from those with open ones.
Consequently, in contrast to the limiter configuration, impurities are less likely to enter the closed surface areas before ionising and are directed to specific targets. Furthermore, this configuration effectively channels plasma losses to a designated section of the vacuum vessel designed to withstand high energy and particle fluxes. The limiter and divertor configurations are depicted in Figure 2.7.

2.3.2 Tokamak Example: ITER

Currently under construction in Cadarache, France, ITER (*"The Way"* in Latin, formerly International Thermonuclear Experimental Reactor) represents the world's largest tokamak and a fundamental step in fusion research [10]. This highly ambi-

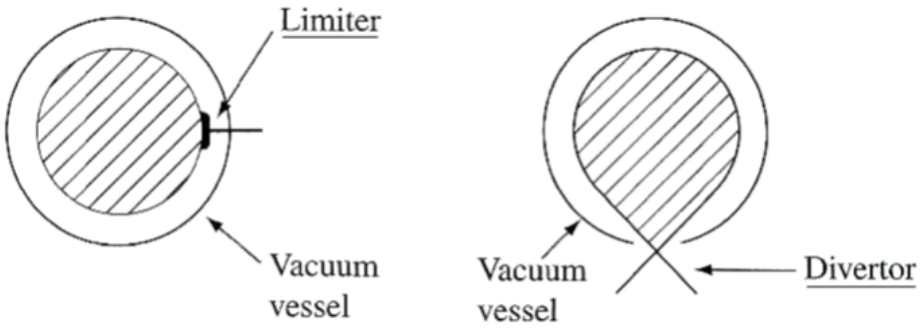


Figure 2.7: The tokamak limiter (on the left) and divertor (on the right) configurations.

tious energy project, initiated in 2006, involves a global partnership of 35 nations, among which are the European Union, the USA, China, Japan, Russia, South Korea, and India.

ITER's primary objectives include generating ten times more thermal energy (up to 500 MW) from fusion than the auxiliary power input (50 MW), achieving steady-state plasma with non-inductive plasma current, maintaining fusion pulses for 400-600 seconds, and validating tritium breeding concepts.
Additionally, it aims to improve neutron shield/heat conversion technologies and develop essential fusion power plant technologies, like advanced materials, superconducting magnets, and remote handling techniques. The insights gained from ITER will pave the way for the DEMO (DEMOnstration Power Plant) project, envisioned to demonstrate practical electricity generation from fusion.

Marking a first, ITER will operate predominantly with deuterium-tritium mixtures containing significant amounts of radioactive material. Its magnetic confinement system consists of a strong toroidal magnetic field ($\sim 5.3\,\mathrm{T}$) at the torus centre, supplemented by a poloidal field generated by a maximum plasma current of 15 MA and six poloidal field coils made of $NbTi$, a superconducting material. Eighteen coils of Nb_3Sn, another superconductor, will create the toroidal field. These materials are also used for the central solenoid, essential for inducing the plasma current. All coils are maintained at 4 K to achieve superconductivity. A ITER tokamak schematic design is provided in Figure 2.8.

The Plasma Facing Component (PFC) will encounter high heat fluxes and neutron irradiation, necessitating suitable materials for the ITER tokamak's first wall

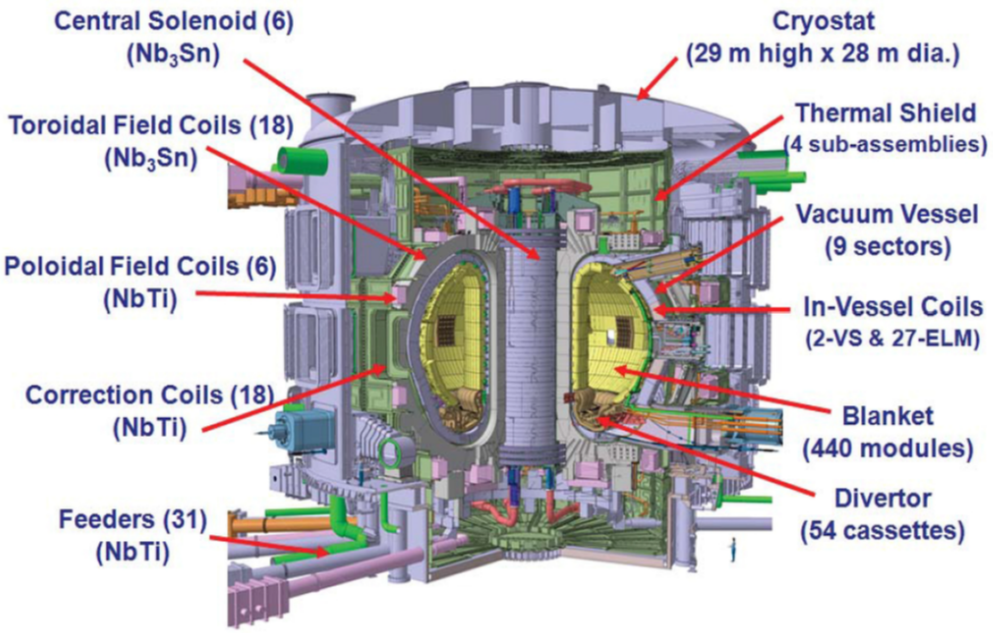


Figure 2.8: Design of the ITER tokamak device.

to prevent plasma contamination. With a height of 26 m and a diameter of 29 m, ITER's plasma volume ($\sim 103\,\mathrm{m}^3$) significantly exceeds previous experiments. This large volume means higher thermal and magnetic energy release during plasma instabilities, posing potential damage risks. The structure must withstand intense mechanical and thermal stresses due to plasma instabilities, making the real-time survey and control system crucial for reactor reliability.

A blanket between the first wall and the coils will test tritium breeding and shield the coils from heat and radiation fluxes from the plasma. The divertor, responsible for extracting plasma impurities and fusion-produced helium, will face intense heat fluxes up to several $\mathrm{MW/m}^2$. The first ITER experiments are scheduled for the end of 2025, marking a significant milestone in fusion research.

2.3.3 Additional Heating Systems

To achieve ignition, plasma pressure and confinement time must meet specific requirements, with a simplified calculation indicating that the minimum is at a plasma temperature of around 10 keV[6], not considering bremsstrahlung radiation losses. According to this calculation, alpha heating becomes the primary heating source at temperatures exceeding 5-7 keV, while the contribution from fusion power below

this temperature range is almost negligible [6]. Additional heating sources are indispensable to attain such temperatures. The simplest form of heating, particularly in tokamaks, is ohmic heating. As plasma is a conductor, the flow of plasma current produces heating, described by the equation $P = \eta j^2$, where η is the plasma resistivity and j is the current density.

However, the efficiency of ohmic heating is limited because plasma resistivity decreases with increasing temperature ($\eta \propto T^{-3/2}$). This reduction in resistivity means that ohmic heating alone can only achieve maximum plasma temperatures of about 3 keV, given typical reactor parameters. Therefore, a gap in the achievable temperature remains, which must be filled by auxiliary heating systems. Without these systems, igniting a reactor is not feasible. Figure 2.9 provides a schematic overview of a toroidal plasma's ohmic and external heating sources. There are various methods for plasma heating in use today, which can be categorised into two main types: coupling electromagnetic waves to the plasma and injecting high-energy particles into the plasma.

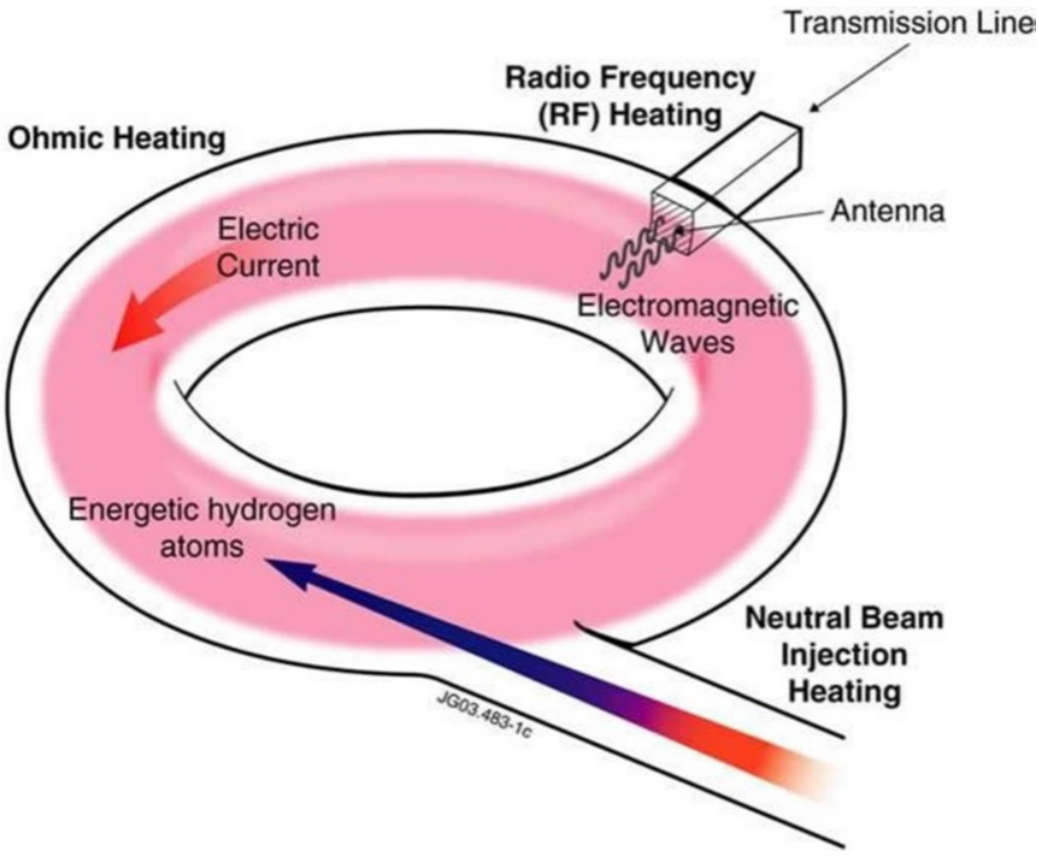


Figure 2.9: Schematic overview of ohmic and external heating sources).

The former category includes electron cyclotron resonance heating Electron Cyclotron Resonance Heating (ECRH), Ion Cyclotron Resonance Heating (ICRH), and Lower Hybrid (LH) heating. The latter primarily encompasses Neutral Beam Injector (NBI) heating. These systems are multifunctional, not only heating the plasma but also capable of driving plasma current when specifically designed for this purpose. This feature is critical in steady-state tokamak (advanced) scenarios where the plasma currents are fully non-inductive. In these advanced scenarios, auxiliary heating systems are the primary means for generating non-inductive current in conjunction with the bootstrap current, an off-axis current generated by natural radial plasma transport dependent on density and temperature gradients. Thus, auxiliary heating systems are essential for plasma heating and Current Drive (CD).

ITER plans to use a combination of ICRH, ECRH, and possibly LH heating systems to deliver up to 20 MW of heating power to the plasma.
ITER's heating and current drive system will include more than one NBI: at least two of them are planned, with the possibility of adding a third one, as illustrated in Figure 2.10.

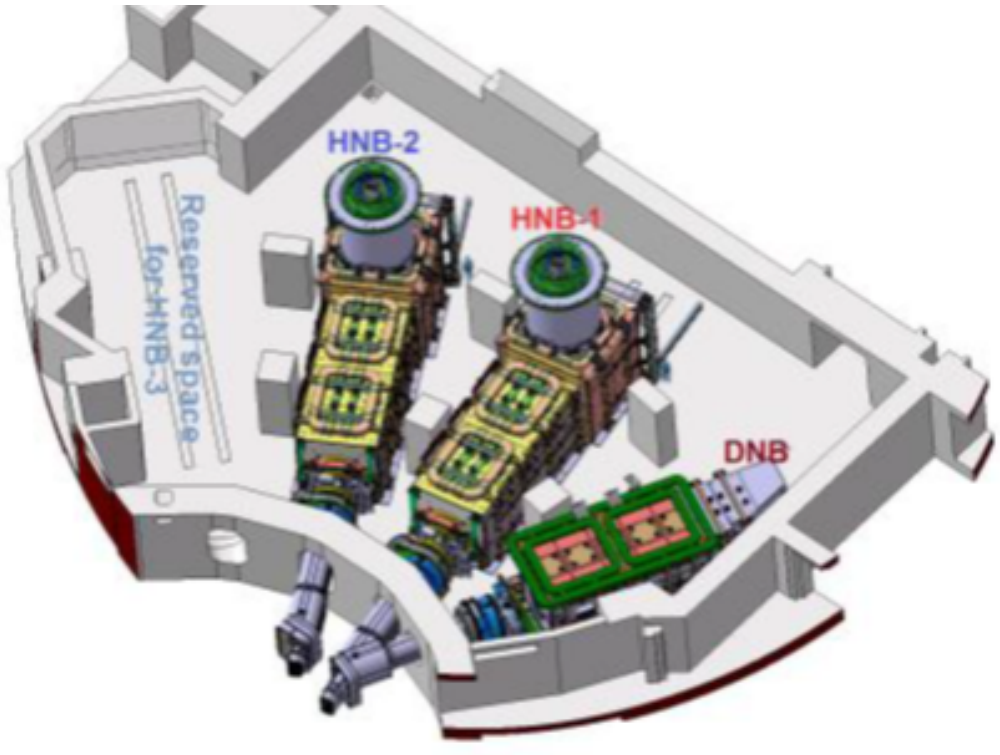


Figure 2.10: Schematic view of ITER NBIs).

2.3.3.1 Neutral Beam Injector

Developed in the early 1970s, neutral beam heating has become a primary technique for heating plasma in fusion experiments. This method involves directing a stream of high-energy neutral particles, typically with energies surpassing the target plasma temperature of 15 keV, into the plasma. As these particles are uncharged, they are unaffected by the magnetic field and move in straight lines until they collide with plasma particles, becoming ionised. Once part of the plasma, these particles contribute to the high-energy end of the deuterium distribution.

Trapping neutral beam particles H_b^0 in the plasma involves specific reactions, denoted as:

$$H_b^0 + H_p^+ \rightarrow H_b^+ + H_p^0 \quad \text{(charge exchange)} \tag{2.5}$$

$$H_b^0 + H_p^+ \rightarrow H_b^+ + H_p^+ + e^- \quad \text{(ionization by ions)} \tag{2.6}$$

$$H_b^0 + e^- \rightarrow H_b^+ + 2e^- \quad \text{(ionization by electrons)} \tag{2.7}$$

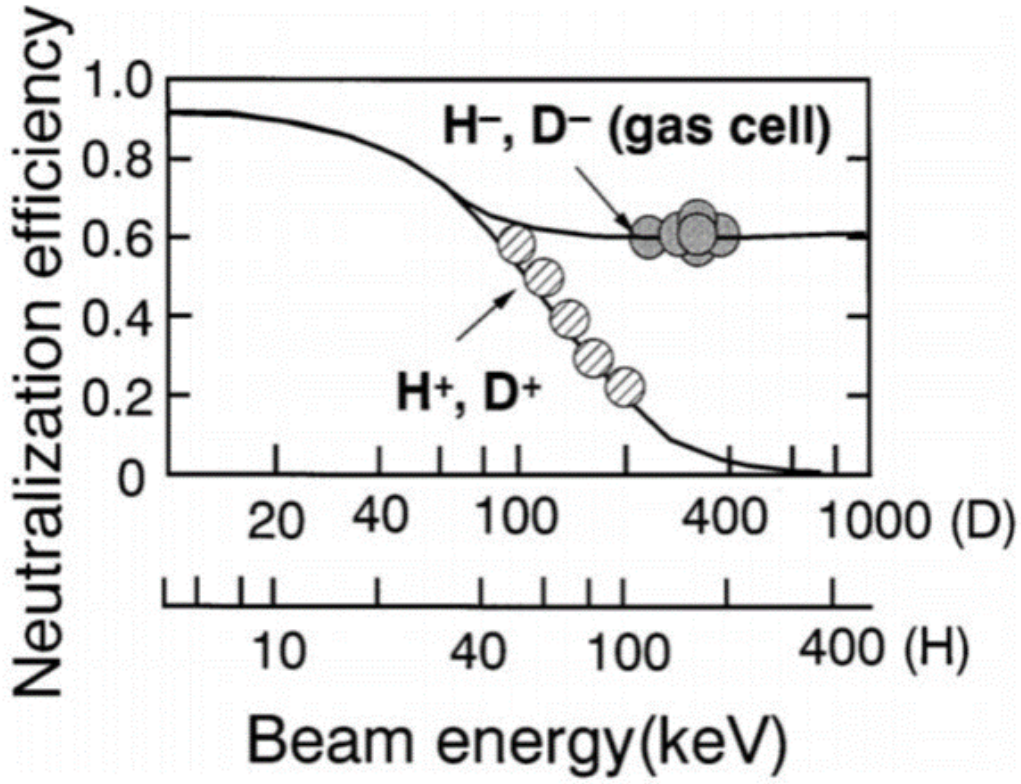


Figure 2.11: Neutralisation efficiency as a function of the beam energy.

where subscripts b and p indicate beam and plasma, respectively [11]. Negative Ion Neutral Beam Injection (NI-NBI), primarily using $H-$ ions, is highly effective for generating high-energy neutral beams, thanks to better neutralisation efficiency above 100 keV [12]. This characteristic is evident from the represented neutralisation efficiency for positive and negative deuterium ions as a function of their energy, shown in Figure 2.11. The low binding energy ($A = 0.75\,\text{eV}$) of $H-$ ions facilitates their transition to neutral states. Despite challenges in generating high-energy beams of negative ions compared to positive ones, NI-NBI systems with beam energies up to 1 MeV are essential for next-generation tokamaks like ITER.

A NI-NBI system includes a negative ion plasma source, an accelerator, and a neutraliser. Negative ions produced in the plasma source are extracted and accelerated to high energies before passing through a charge exchange cell for partial neutralisation. The remaining ions are magnetically deflected to a dump. Post-neutralization, the beam comprises roughly 60% neutral atoms, 20% negative hydrogen ions ($H-$), and 20% positive hydrogen ions ($H+$) [13].

The NI-NBI system is comprised of four main components:

1. **Plasma Source**: This generates the ions to be accelerated, either through hot cathodes (heated filament or bow discharge) or radio-frequency (RF) antennas placed inside or outside the discharge area.

2. **Acceleration Stage**: Negative ions from the source are accelerated through a voltage drop between grids in the accelerating column, forming a beam or beamlets that combine into a larger beam.

3. **Neutralizer**: Here, the ion beam is neutralised while retaining its high kinetic energy and focus, allowing the energetic neutral particles to travel through the NBI duct into the tokamak's plasma.

4. **Residual Ion Dump System**: A specialised magnetic field redirects unneutralised negative and positive ions to a collection system.

2.3.3.2 The Neutral Beam Test Facility (NBTF)

Located in Padova, Italy, at the Consorzio RFX site, the Neutral Beam Test Facility (NBTF) is a research and development complex funded by the ITER Organisation. The NBTF scope is to develop and test the ITER Heating Neutral Beam Injector (HNB), whose requirements are far beyond the current HNB technology. It mainly

consists of two large scale experiments: Source for Production of Ion of Deuterium Extracted from Rf plasma (SPIDER), the full-scale prototype of the HNB negative ion source, and Megavolt ITER Injector & Concept Advancement (MITICA) [14, 15, 16], the one-to-one prototype of ITER HNB [17].
Therefore, it plays a pivotal role in achieving ITER's objectives, specifically in the area of additional heating systems essential for reaching high-confinement regimes and significant fusion reactions.
The additional heating systems of ITER include two heating neutral beam injectors, each with a heating power of 16.5 MW, and ion and electron cyclotron resonant heating systems, providing 20 and 24 MW of power, respectively.

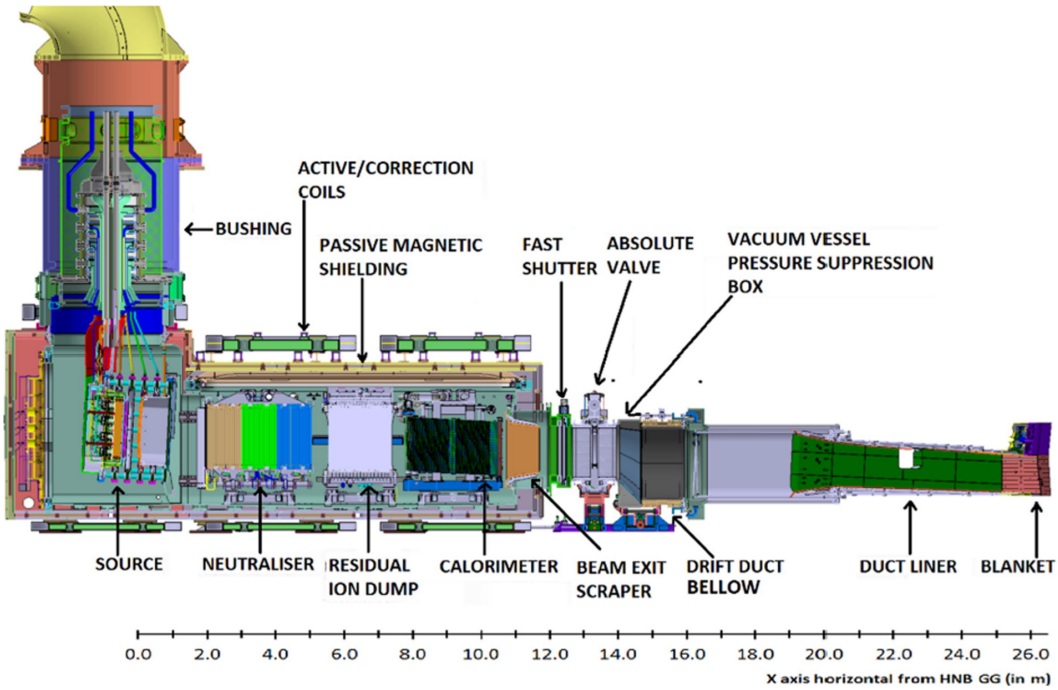


Figure 2.12: Sectional view of the HNB beam line.

The HNB at the NBTF exhibits fundamental differences in its device components, namely the constituent plant units: they can be considered a more extensive set than the HNB's. It encompasses additional elements like service plant units, which include the cooling system, gas and vacuum system, cryogenic plant, and medium voltage power distribution grid.
It also features diagnostic plant units, such as beam and source thermal sensors, RID electrostatic probes, visible and infrared imaging, source and beam emission spectroscopy, cavity ring-down spectroscopy, beam tomography, and neutron detection. In contrast, in the HNB, these service functions are supplied by independent ITER

plant systems, such as vacuum, water cooling, and cryogenics. Notably, the HNB does not integrate diagnostic systems due to reliability concerns. Moreover, the HNB at the NBTF necessitates the integration of central Instrumentation and Control (I&C) systems, including the Control and Data Acquisition System (CODAS), Central Interlock System (CIS), and Central Safety System (CSS). These systems are distinct and operate independently from the HNB in the ITER framework.

The HNB at the NBTF is designed to produce a high-energy (1 MeV) and high-power neutral beam in either hydrogen (H) or deuterium (D). The operational principle of the ITER neutral beam injector involves creating plasma in the ion source through radio frequency. Negative ions ($H-/D-$) are generated in the ion source by introducing Caesium, which reduces the surface work function, aiding in the creation of negative ions. These ions are then accelerated electrostatically from a negative potential (-1000 kV) to ground potential through five acceleration gaps. The resulting ion beam, with a current up to 40-50 A, undergoes neutralisation by charge exchange, while residual ions are electromagnetically dumped. Key operational goals for the HNB include achieving one-hour continuous operation, maintaining low beam divergence, and limiting the fraction of co-extracted electrons. Negative ions are preferred in high-energy beam injectors for fusion due to their high neutralisation efficiency ($\sim 60\%$) at energies up to 1 MeV. In contrast, positive ions' neutralisation efficiency significantly decreases above 100 keV. A sectional view of the HNB beam line is provided in Figure 2.12.

The NBTF is in an advanced stage of construction and encompasses a two-phase R&D program, reflecting the development of the two main experimental facilities. The first phase, focused on optimising the engineering and operation of the ion source, is undertaken through the SPIDER experiment. SPIDER, sharing the same ion source as the HNB but operating at a reduced acceleration potential gap (-100 kV), is currently in the phase of integrated commissioning. The second phase involves constructing and operating the complete prototype of the ITER HNB, conducted through the MITICA experiment, which is presently under construction.

2.3.4 RFP Configuration

The basic idea behind the RFP configuration, similar to the tokamak, is plasma confinement achieved through the so-called pinch effect: the passage of an unidirectional current in the plasma creates an azimuthal magnetic field that constricts (pinches) the plasma, resulting in magnetic confinement.

Historically, the toroidal pinch was the first to utilise this effect. Extensive research on this concept has been ongoing since the 1950s. Early versions of the pinch, including devices like the Zeta-experiment and HBTX-I in the UK, ZT-I and ZR-II in the US, and Eta-Beta-I followed by Eta-Beta-II in Padova, encountered serious stability issues due to kinks and other Magnetohydrodynamics (MHD) modes. In the Soviet Union, this research line led to the tokamak configuration's development, while European and American experiments branched into other directions, notably the RFP configuration.

An RFP device is an axis-symmetric toroidal system where plasma confinement is achieved through a poloidal magnetic field B_θ, primarily generated by the plasma current itself. This poloidal field combines with a toroidal field B_ϕ produced by external coils. The two components in the RFP configuration are approximately of the same magnitude ($B_\theta \approx B_\phi$), a distinct contrast to the tokamak configuration where the magnetic field profiles are markedly different, as illustrated in Figure 2.13 [18].

Two parameters characterise the RFP configuration:

1. The pinch parameter Θ, defined as $\Theta = \frac{B_\theta(a)}{\langle B_\varphi \rangle}$
2. The reversal parameter, defined as $F = \frac{B_\varphi(a)}{\langle B_\varphi \rangle}$.

These parameters are derived from the toroidal and poloidal fields measured at the boundary, where $B_\varphi(a)$ and $B_\theta(a)$ represent the respective field components. A notable aspect of the RFP is that the value of Θ is significantly greater than in a tokamak, signifying that the same magnetic flux produced by the central solenoid in an RFP induces a higher current. As a result, the safety factor profiles in RFPs typically show a monotonic decrease, with $q_{RPF} \ll 1 \ll q_{TOK}$ due to the relatively higher plasma current.

A distinctive feature of the RFP configuration is that the toroidal magnetic field often reverses at the plasma boundary, as shown in Figure 2.14, resulting in a negative safety factor ($q(a) < 0$). This operation beyond the *Kruskal-Shafranov limit* is facilitated by a combination of a sheared magnetic field, achieved through the reversal of B_φ, and a conducting shell at the plasma boundary. This setup contrasts with the tokamak configuration, where a perfectly conducting external wall would stabilise specific modes. However, feedback stabilisation becomes necessary in prac-

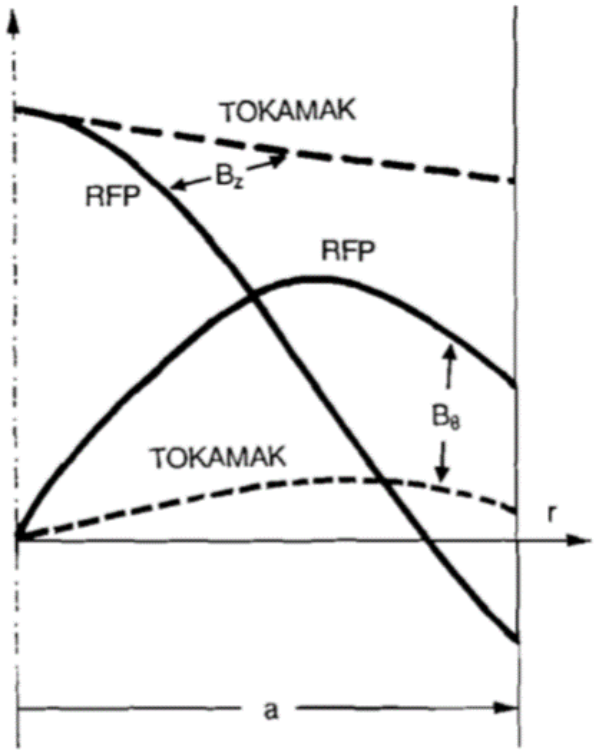


Figure 2.13: Radial field profiles for toroidal and poloidal components of the magnetic field. The dashed line represents tokamak, and the solid line RFP.

tical scenarios involving walls with finite resistivity.

In RFPs, these operational conditions lead to the development of $m = 1$ resistive kinks, manifesting as helical deformations of the plasma. These modes evolve non-linearly, generating a fluctuation-induced electric field $E_f = -\langle v \times B \rangle$, where v is the plasma velocity, integral to sustaining the RFP configuration by eliminating parallel current gradients. This phenomenon, known as the dynamo effect, is particularly pronounced at low current, where the RFP exhibits a Multiple-Helicity regime characterised by the simultaneous instability of numerous resistive modes with varying toroidal numbers n. Early theoretical models for RFP equilibria, such as Taylor's relaxation theory [19], started from a perfectly conducting plasma model and assumed the conservation of total magnetic helicity and magnetic flux, leading to unique equilibrium states characterised by the aforementioned reversal and pinch parameters. Taylor's theory has shown significant alignment with experimental results [18, 20], though it only qualitatively captures RFP physics. More accurate predictions are provided by MHD models, which have forecasted the presence of Single Helicity states [21]. These states have been experimentally observed in large

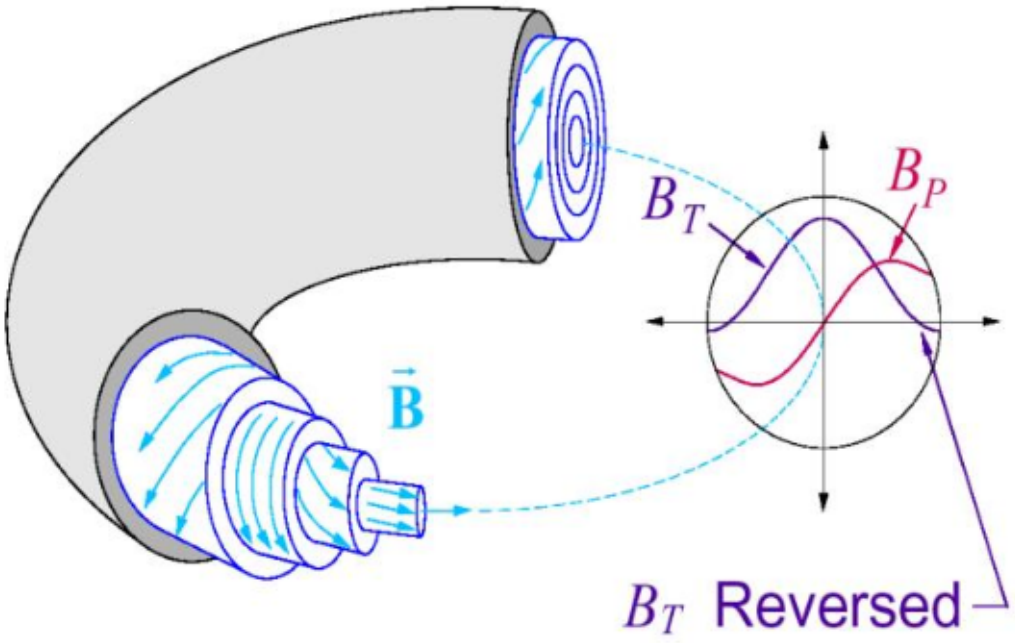


Figure 2.14: Radial profile of magnetic field components in an RFP (B_T is the toroidal magnetic field, B_P the poloidal one).

RFP devices, where transitions from Multiple Helicity to Quasi Single Helicity states [22, 23, 24], occur at high currents (1 MA), persisting throughout the entire discharge flat top [25]. Recent developments in devices like RFX-mod have demonstrated the experimental verification of a self-organised helical state that develops at high currents (> 1.5 MA) linked to the presence of an internal transport barrier [26]. The optimisation of feedback control systems in these experiments has facilitated high-current operation and the persistence of quasi-single-helicity states [27, 28, 29].

RFP fusion experiments present advantages and disadvantages compared to tokamak experiments. For instance, RFPs require smaller currents in their external coils to generate the necessary toroidal fields, contrasting with tokamak's reliance on large superconducting coils. However, due to transport phenomena, RFPs face challenges achieving high energy confinement times (τ_E values). To address these challenges, modern RFP devices, such as RFX-mod, are equipped with complex active control systems designed to stabilise Resistive Wall Modes, enhancing pulse duration. These systems can manage multiple instabilities simultaneously in an RFP, making them ideal for benchmarking environments in tokamak stability studies involving multiple unstable modes. Additionally, RFPs are inherently pulsed due to the necessity of a time-varying magnetic field sustained by the primary transformer for generating

plasma current.

In terms of equilibrium modelling, RFPs can be described by a minimal one-dimensional model based on the MHD equilibrium equation $\nabla p = \vec{j} \times \vec{B}$ in cylindrical coordinates, complemented by parametric descriptions of current and pressure profiles [30]. This approach offers a simplified, zero-order representation of RFP equilibrium. More complex models incorporate toroidal geometry, resulting in an outward radial shift of magnetic flux surfaces known as the Grad-Shafranov shift [31]. However, a realistic depiction of RFP equilibrium demands a full 3-dimensional description, addressing symmetry-breaking caused by tearing modes at resonant surfaces. It leads to magnetic stochasticity and reduced confinement in RFPs. Symmetry is restored in specific RFP states, such as Single Helicity, described by a helical Grad-Shafranov equation [32, 33], representing a 3D-solution to the MHD equilibrium equation.

2.3.5 Reversed Field Pinch Example: RFX

In the late 1970s, research on small RFP experiments like $\eta\beta$ and $\eta\beta 2$ began in Padua. This research led to the development of RFX (Reversed Field eXperiment), a reversed field experiment machine with a major radius R0 of 2 m and a minor radius of 0.5 m, designed to handle a plasma current of 2 MA. However, the maximum plasma current achieved was only 1.2 MA.

Between 2001 and 2004, the same machine underwent modifications, becoming RFX-mod (Figure 2.15), with an active feedback control system that significantly improved its performance, enabling it to reach its original target current of 2 MA.

RFX was constructed with a vacuum vessel made of Inconel 6257 and internally covered with 2016 carbon plates to shield it from sudden heat impulses generated by the plasma. It could withstand temperatures exceeding 350 °C and maintain ultra-high vacuum conditions ($P \approx 10^{-8}$ mB).

The structure (called the liner) comprised 72 cuneiform elements arranged at 5-degree intervals in the toroidal direction, with an internal layer of 1 mm and an external layer of 2 mm, connected by a 0.5 mm corrugated ring. The vacuum vessel remained intact in RFX-mod.
Figure 2.16 depicts RFX-mod, showcasing its key active and passive components.

On RFX, the toroidal complex was held in place by a thick aluminium shell,

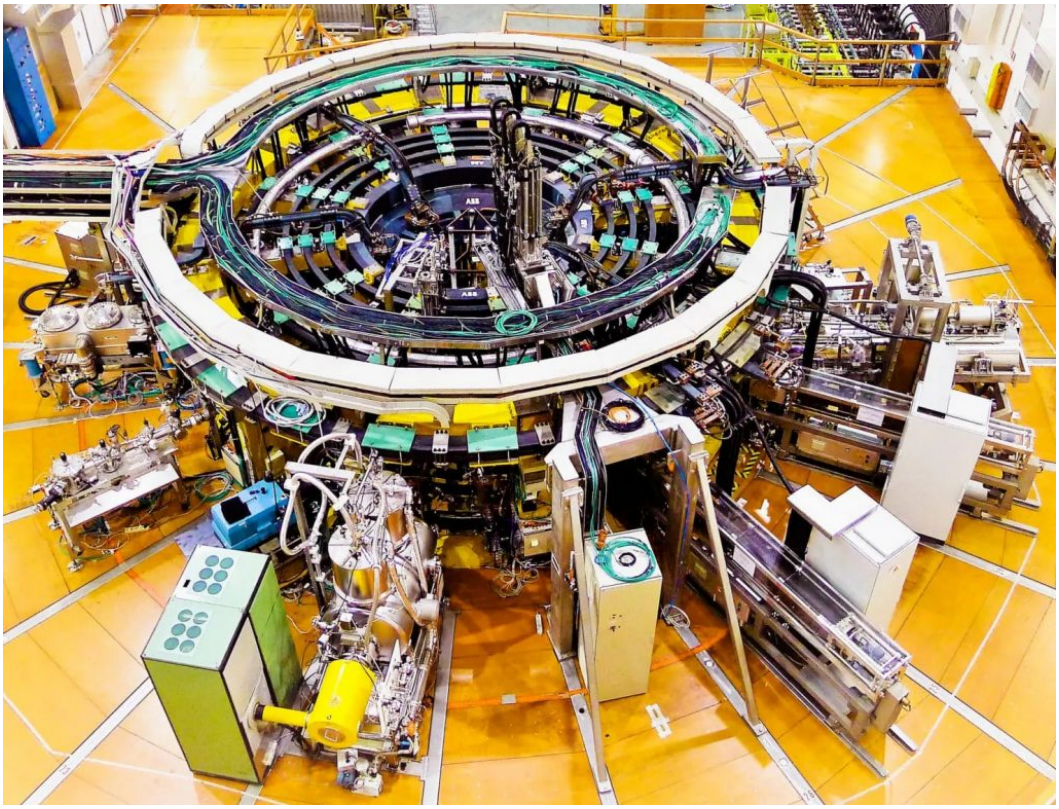


Figure 2.15: Photo of RFX-mod.

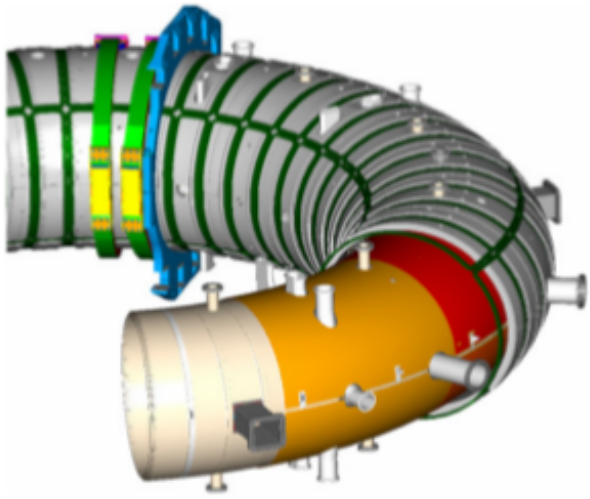


Figure 2.16: Graphical representation of RFX-mod active coils (green), passive support structures (grey), copper shell (orange) and vacuum vessel (yellow).

which also acted as a passive stabiliser with a time constant of 500 ms. RFX-mod replaced it with a stainless steel mechanical support structure and a 3 mm copper shell (with a time constant of 50 ms), displayed in Figure 2.17, wrapped around the

vacuum vessel. The whole RFX-mod vessel complex is illustrated in Figure 2.18. Its shell consisted of four parts, each spanning 180 degrees in the poloidal direction, with an overlap in the toroidal direction to reduce error fields induced by plasma perturbations. Two gaps were left unwelded, allowing magnetic fields to penetrate the shell. The induced currents in the shell countered plasma displacements, albeit with limited effectiveness on fast-growing perturbations and in the early stages of a pulse.
Table 2.1 contains a detailed listing of the primary technical specifications for the most recent version of the machine in operation.

Minor radius (a)	2 m
Major radius (R)	0.427 m
Plasma current	≤ 2 MA
β	10%
T_e	1 keV
Applied toroidal field	≤ 0.55 T
Magnetic circuit flux	15 Wb

Table 2.1: RFX-mod technical specifications.

A dynamic feedback system was implemented to stabilise equilibrium and counteract instabilities over extended periods [34]. This system incorporates 192 active saddle coils [35], as depicted in Figure 2.19a, systematically organised into four toroidal arrays and 48 poloidal segments. These coils are situated externally to the copper shell and are affixed to a toroidal support structure made of stainless steel. Every coil operates independently and can handle up to 400 A, generating a peak radial field of approximately 50 mT.

Specialised probes are utilised to control modes, measuring each aspect of the magnetic field (poloidal, toroidal, radial), with an ensemble of 192 sensors for each component. The measurement of the poloidal and toroidal components is carried out using pick-up probes. In contrast, the radial field measurements are conducted using saddle coil probes positioned beneath each active control coil mentioned earlier. Consequently, these saddle probes mirror the periodicity of the active coils, encompassing identical poloidal and toroidal angles. This setup allows for assessing the average radial field across the area covered by the probes. The entire system is governed by a PID controller, which operates effectively in both real space -

Figure 2.17: Exploded view of the RFX-mod copper shell.

by adjusting gains to the field signals - and in the Fourier space denoted by (m; n), applying gains to Fourier harmonics within the range of $n = [-23; +24]$ and $m = [-1; 0; +1; +2]$.
The versatility of this actuator system is noteworthy, as it can accommodate several experimental requirements. They include using a smaller set of coils or the so-called "supercoils" - essentially neighbouring coils powered to simulate a larger, singular coil.

Additionally, three other sets of coils play a crucial role in establishing the magnetic configuration. The toroidal field coils, 48 in total and illustrated in Figure 2.19c, encase the entire torus, creating a maximum magnetic field of 0.55 T. These coils are organised into 12 groups, each comprising four, and are powered independently. The magnetising coils, shown in Figure 2.19d, generate the time-varying magnetic flux necessary for initiating the plasma current. With a maximum current of 50 kA coursing through these coils, they generate a flux of 15 Wb. Furthermore, a secondary poloidal field, essential for managing the plasma's position

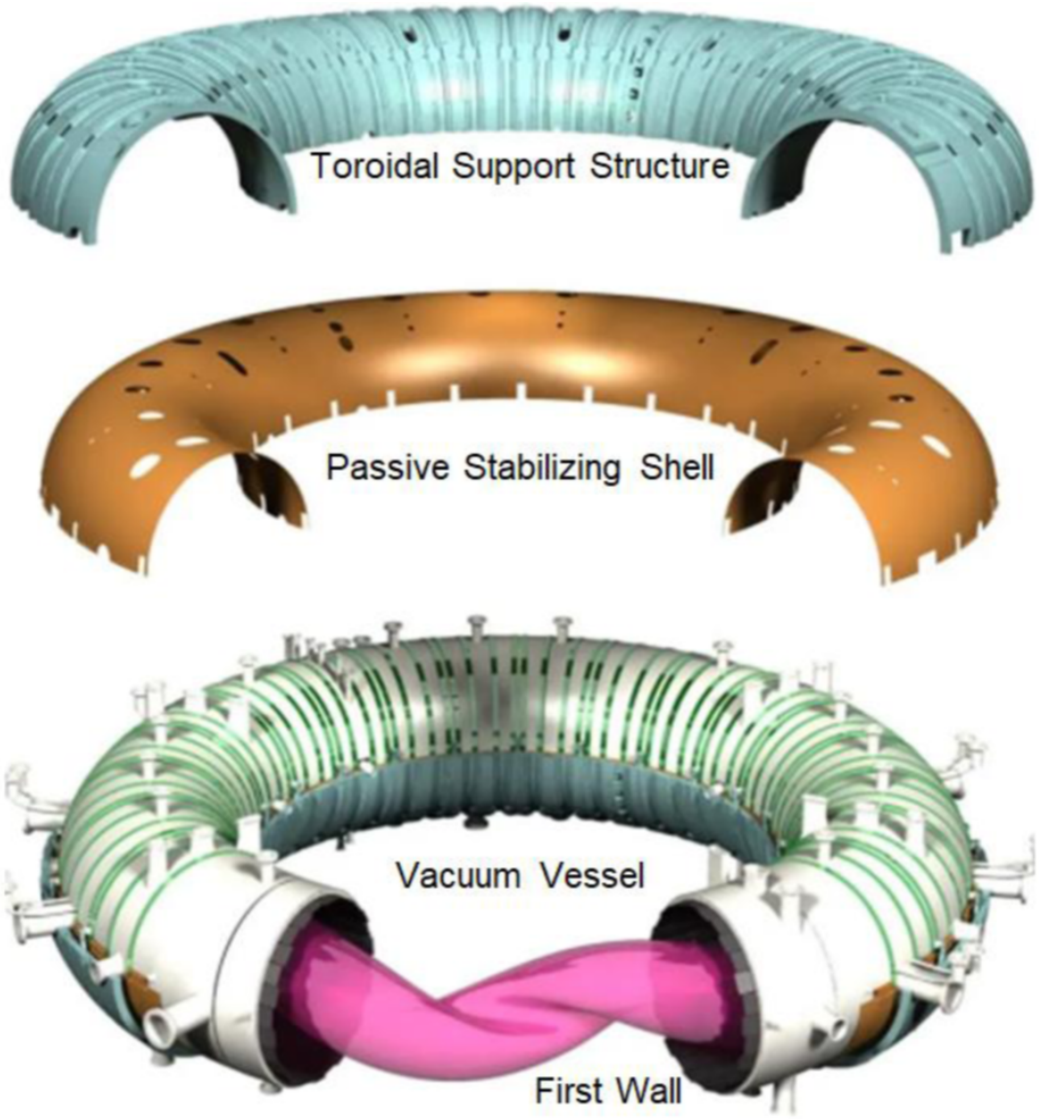
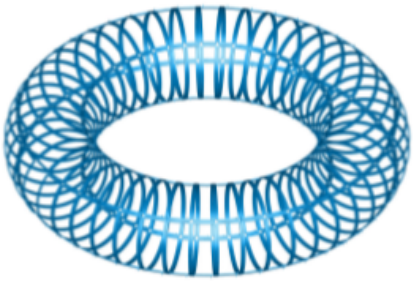


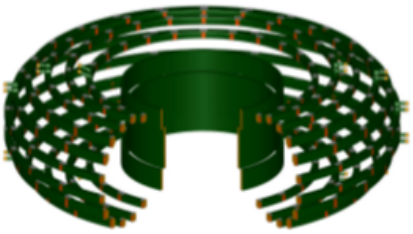
Figure 2.18: The RFX-mod vessel complex, comprising the external support structure, the passive stabilising shell, the vacuum vessel and the first wall.

during discharges, is produced by what is referred to as the primary winding, as seen in Figure 2.19b.

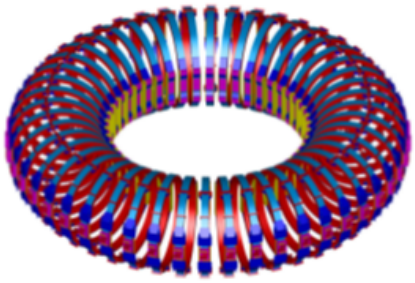
The system was fine-tuned and optimised, but the control algorithm had to be simplified for real-time implementation. One of the simplifications involved performing calculations in the cylindrical geometry of the machine, which introduced small but still finite spurious harmonic components when mapping fields into the proper plasma toroidal geometry and calculating harmonic components in the correct geometry required solving a complex system of differential equations, which was



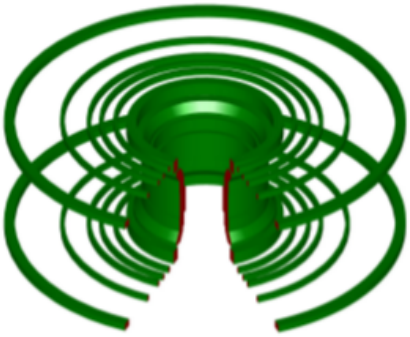
(a) The 192 actuators of the active control system.



(b) The field shaping winding



(c) The 48 toroidal field coils



(d) The ohmic heating winding

Figure 2.19: The RFX-mod coil sets.

significantly slower and had unpredictable execution times. Consequently, alternative solutions need to be explored.

The RFX-mod apparatus's adaptable nature permitted the investigation of various magnetic configurations based on the safety factor [27]. Particularly in RFP conditions, and more so at elevated plasma currents, there have been observations and analyses of transitions to enhanced confinement helical states [36], aligning with theoretical and computational forecasts [37]. This device has consistently achieved stable, very low q (edge $q < 2$) ohmic tokamak discharges through active management [38]. Research into ultra-low q regimes has also been conducted [36]. Introducing a polarised insertable electrode has led to the attainment of H-mode in tokamak plasmas [39].

The characteristics of RFP plasmas in RFX-mod have been significantly affected by residual MHD instabilities, specifically Tearing Modes. These modes' amplitude and phase dynamics, which are highly nonlinear, are significantly impacted by the features of the toroidal structure enclosing the plasma. The Inconel vacuum vessel's exceptionally high resistivity, the most among all RFP devices, resulted in Tearing

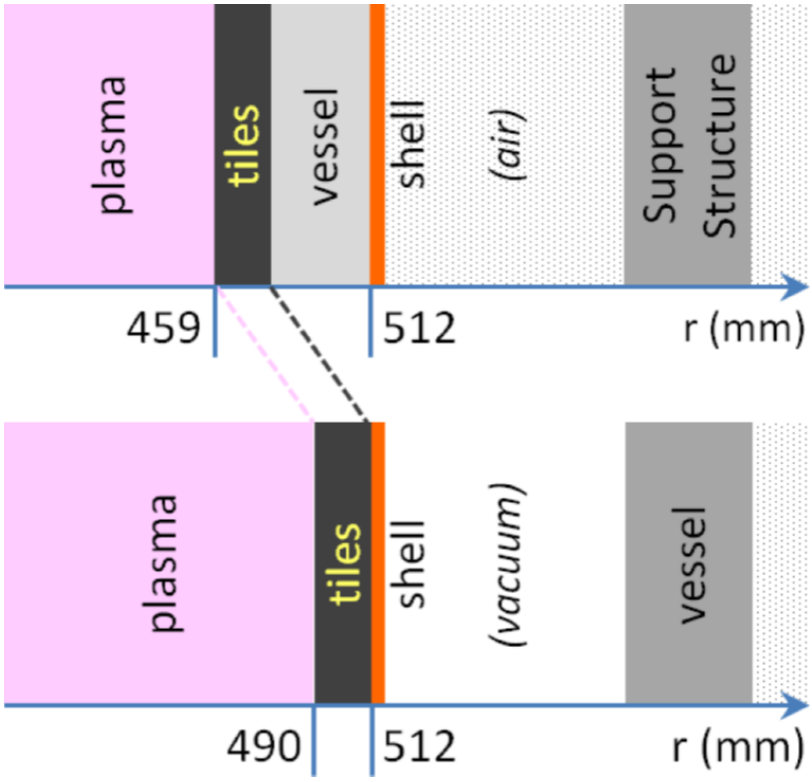


Figure 2.20: Schematic concept of the evolution of the magnetic front-end layout in RFX-mod2: RFX-mod (above), RFX-mod2 (below).

Modes being consistently wall-locked across all plasma current regimes examined by RFX. The active control within RFX-mod has facilitated mitigating localised interactions caused by wall-locked tearing modes and bulging. Experiments with very low plasma currents ($I_p < 150\,\mathrm{kA}$) have unveiled spontaneous, rapidly rotating tearing modes. Furthermore, the vessel's proximity has influenced the very low q ohmic tokamak operations [36]. After recognising the constraints imposed by its toroidal complex [40], a significant redesign of the RFX experiment has been suggested, dubbed RFX-mod2, marking the second significant alteration since its initial concept.

The experimental outcomes and numerical simulations from RFX-mod [41] suggest that a highly conductive shell near the plasma boundary could facilitate higher plasma current thresholds for wall locking in RFP regimes. This modification is also anticipated to enhance plasma startup control, lower the saturated level of tearing modes non-linearly, and reduce plasma-wall interactions, thus improving confinement.
The primary design objectives for the RFX-mod2 upgrade have been to remove the

existing resistive vacuum vessel and to minimise the shell-plasma proximity. This change is expected to significantly reduce plasma-wall interactions, particularly those associated with the last closed magnetic flux surface distortion, and enhance self-organised helical plasma regimes [27, 37].

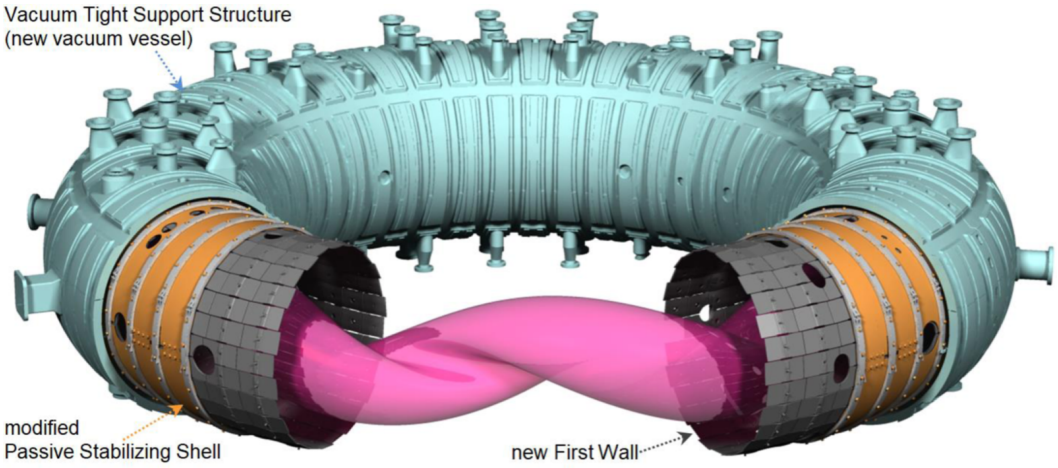


Figure 2.21: The RFX-mod2 vessel complex: the support structure is now vacuum-tight, allowing the passive stabilising shell to be closer to the plasma.

The conceptual schematic of the magnetic front-end transformation from RFX-mod to RFX-mod2 is depicted in Figure 2.20. Figure 2.21 displays the CAD model of the actual RFX-mod2 vessel complex, with the shell modified to directly support the first wall and enclosed in a newly adapted vacuum vessel from the external Support Structure.

By replacing Inconel with copper, RFX-mod2 aims to become the RFP device with the lowest resistivity first continuous conductor around the plasma, nearly half that of the aluminium shell in the MST device. This alteration is expected to facilitate the characterisation of RFP properties with optimised passive boundaries and complete the RFX mission of exploring RFP properties at high currents.

Part II

The Control and Data Acquisition System (CODAS) in Fusion Experiments

CHAPTER 3

THE CONTROL AND DATA ACQUISITION SYSTEM (CODAS) FOR FUSION EXPERIMENTS

The Control and Data Acquisition System (CODAS) stands as an indispensable cornerstone in the landscape of nuclear fusion experiments. This chapter delves into the complexities and pivotal role of CODAS, a system integral to orchestrating and surveilling the intricate fusion process. It starts by exploring the nuances of the Plasma Control System (PCS) and Fast Control in Section 3.2, emphasising their essential contributions to real-time plasma management. The chapter then transitions to Section 3.3, where it dissects the Data Acquisition process, highlighting the challenges in managing and interpreting the voluminous data generated during fusion experiments. Section 3.4 discusses the crucial aspects of Slow Control and Plant Supervision, underscoring their role in maintaining stability and safety over extended periods. The chapter then provides insights into the applications and significance of Supervisory Control And Data Acquisition (SCADA) systems in Section 3.5, reflecting on its impact across various scientific domains. Finally, in Section 3.6, it presents the notable example of ITER's CODAC as a use case, illustrating the challenges faced by the CODAS system in a complex fusion experimental setup.

3.1 Harnessing Complexity: CODAS at the Forefront of Fusion Research

The CODAS in nuclear fusion experiments is an essential, sophisticated system designed to manage and oversee the complex processes involved in creating and maintaining nuclear fusion. It is at the heart of fusion research, controlling the fusion process and acquiring vast data from the experiments. This task involves intricate management of plasma generation and containment - critical elements in nuclear fusion - and continuous monitoring and adjusting of various experimental parameters, including temperature, pressure, magnetic field strength, and plasma behaviour.

CODAS are highly customised, given the unique nature and high stakes of each nuclear fusion experiment, such as those in tokamaks or stellarators. They are engineered to provide real-time control over the fusion process and manage and process the substantial volume of data generated. These systems have advanced algorithms and control mechanisms that allow instantaneous adjustments based on the real-time data they receive. This feature is critical given plasma behaviour's dynamic and unpredictable nature in fusion reactions.

The data acquisition aspect of CODAS is equally vital. Fusion experiments are known for generating enormous volumes of data, given the plethora of sensors and diagnostic tools employed. CODAS systems are meticulously designed to efficiently handle this influx of data, ensuring it is accurately recorded and stored for later analysis. This data is essential for researchers to understand the fusion process's complexities and make informed improvements in subsequent experiments.

Integration with other systems is another crucial aspect of CODAS functionality. These systems must seamlessly connect with various other components of the fusion experiment setup, including safety systems, power supplies, cooling systems, and various diagnostic tools. This integration is critical not only for the experiment's operational success but also for ensuring the overall safety of the process.

User interfaces in CODAS are developed to be intuitive, allowing scientists and engineers to monitor and control the experiment effectively. These interfaces also ensure that the vast amounts of data collected are accessible and analysable efficiently, which is crucial for ongoing research and development.

The challenges in developing effective CODAS for fusion experiments are significant. They include managing the extreme environmental conditions within a fusion reactor, ensuring real-time response capabilities, and handling large and complex data sets. As nuclear fusion technology evolves, the demand for more advanced and capable CODAS grows, making it a critical area of focus in ongoing fusion research and development. These systems' efficiency and capabilities are pivotal in advancing nuclear fusion research, as they directly influence the feasibility and success of generating and harnessing fusion energy.

3.2 Plasma Control System and Fast Control

The PCS (also called Plasma Real-Time System) is a pivotal component in the operation of a fusion machine. It encompasses a combination of hardware and software components essential for precisely controlling the various elements within the fusion device. This control system operates based on pre-programmed references and real-time data obtained from diagnostics [42]. The PCS is the linchpin for creating a feedback loop, which processes data during the discharge phase, effectively managing and controlling the plasma and machine components.

The need for a PCS is driven by the intricate nature of achieving fusion performance targets, such as maintaining a stable flat-top duration, controlling plasma temperature, shaping the plasma, and managing other crucial parameters within the plasma and machine [43]. Given the rapid and multifaceted changes that can occur in experimental conditions during an experiment, it becomes imperative to possess a system that can swiftly evaluate the state of the plasma and machine in real time. Consequently, based on this assessment, the PCS can expediently generate feedback references for the actuators [44].

Specifically, the PCS carries out several essential functions:

- **Input and Output Management:** This involves the intricate management of signal flow, which is indispensable for interacting with the hardware components of the system, both concerning input (diagnostics) and output (actuators). In fusion devices, input signals typically include data from coils that measure magnetic fields, currents, and voltages vital for reconstructing the plasma state. These inputs are channelled to the PCS via Analog-to-Digital Converter (ADC) devices. Outputs often consist of references created for the power supplies governing the coils responsible for shaping the electromagnetic fields. These references, in turn, play a pivotal role in maintaining the plasma's position and shape. While such references were traditionally entirely analog

and output from the PCS through Digital-to-Analog Converter (DAC) devices, there is a growing trend towards direct digital communication between the PCS and the power supplies.

- **Communication:** The PCS facilitates the seamless real-time exchange of signals among its diverse components. This exchange can be achieved through shared memory within the system or, in cases involving distributed PCSs, via network-based communication.
- **Algorithm Computation:** The PCS assumes responsibility for executing complex computations based on the received inputs. These computations entail deriving quantities from the system inputs and determining the appropriate actuator references. This step is paramount in ensuring that the fusion machine operates in line with the predetermined parameters and objectives.

A PCS comprises various hardware and software components, including interfaces for component communication. Figure 3.1 illustrates the primary elements of a PCS within the context of a fusion device.

Plasma Diagnostics in a PCS are designed for real-time plasma and machine characteristics measurement. These diagnostics provide feedback signals for immediate use and data for post-operation analysis. Even though some PCS also rely on plasma density measurements obtained through interferometry[45], magnetic diagnostics are the most commonly used ones[46, 47]. They provide essential feedback during a discharge, crucial for determining plasma position and shape. Standard magnetic diagnostics include:

- Saddle loops for radial magnetic field measurement.
- Diamagnetic loops encircling the plasma poloidally for assessing magnetic field B and occasionally plasma current I_P.
- Flux loops, circling the plasma toroidally to gauge the plasma-generated flux.
- Localised coils for pinpoint measurements of the magnetic field B's radial, poloidal, or toroidal components.

These diagnostics generate voltage measurements, processed by an ADC and converted into a digital signal for the real-time system. ADCs transform continuous signals into digital samples at regular intervals, essential for capturing signals from electromagnetic diagnostics without aliasing.

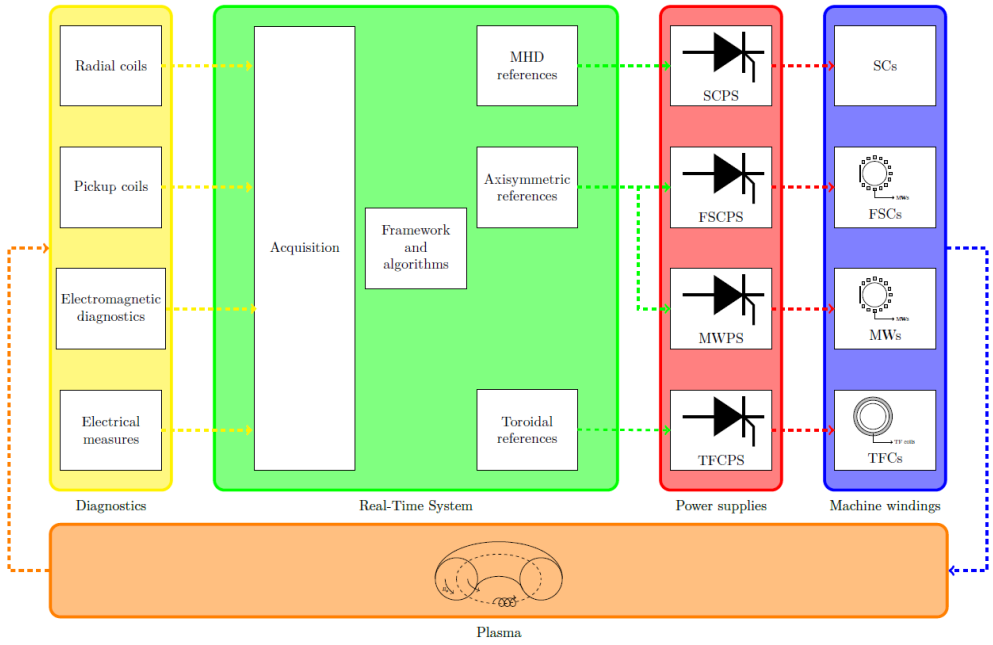


Figure 3.1: Standard setup and main components of a PCS (only magnetic diagnostics are reported for simplicity's sake). The diagnostics (yellow) acquire plasma and machine state data, which are then processed by the RTS (green) directly or through an ADC. A software framework executes algorithms to determine the necessary reactions to these inputs. These outputs are fed directly or via a DAC to the power supplies (red) that operate the device's actuators (blue).

The Real-Time System (RTS) processes inputs from acquisitions (via ADCs or direct capture) using algorithms to produce outputs. It typically consists of a software framework executing multiple custom functions. RTS outputs are instrumental in controlling plasma actuators.
Plasma Actuators in a PCS are designed to modify plasma position and shape, aiming for the desired plasma scenario. These actuators consist mainly of coils needing a power source.

3.2.1 MARTe2

The MARTe (Multi-threaded Application Real-Time executor) [48] framework is a robust tool for developing real-time control system applications. It has been widely utilised in fusion real-time control systems, particularly in the JET (Joint European Torus) tokamak.

A fundamental strength of the MARTe architecture lies in its clear separation of concerns. It neatly partitions the platform-specific implementation, environmental

details, and real-time algorithms (i.e., user code). This clear distinction simplifies development and maintenance by ensuring these different aspects can be managed independently.

The framework is not bound to a specific platform or environment. The target processor and operating system determine the platform, while the environment encapsulates the interfacing details of the deployment location. This separation allows the same code to be employed in various environments, minimising the need for extensive modifications during deployment.

Code re-usability is a significant benefit of this approach. Different systems using MARTe often share standard parameter configuration and data retrieval services. Moreover, it facilitates developing and testing user algorithms in non-real-time operating systems before deploying the same code in a real-time environment.

MARTe1 has steadily expanded its support for different environments and platforms over time. It has led to the exposure of the core code to various environment configurations, instilling confidence in its quality and robustness. Additionally, it has fostered a community of developers from diverse backgrounds, including the scientific community and industry. The development of MARTe2 focused on creating a suitable Quality Assurance (QA) strategy to integrate contributions from this broad community. This collaborative approach enhances the framework's versatility and applicability.

In summary, MARTe is a versatile and platform-independent software framework used for developing real-time control systems, particularly in the field of fusion. Its strength lies in its separation of concerns, facilitating code re-usability, and accommodating diverse environments and platforms. It has become a valuable fusion and real-time control system community resource.

3.3 Data Acquisition

Data acquisition in nuclear fusion experiments is an intricate and essential endeavour involving the gathering, processing, and interpreting of a vast array of data. Data collection is extensive in complex nuclear fusion environments such as tokamaks or inertial confinement fusion devices due to the intricate nature of plasma behaviour and the myriad parameters influencing fusion reactions.

The process begins with deploying various sensors and diagnostic tools, including magnetic probes, Thomson scattering systems, neutron cameras, and Langmuir probes. These instruments are crucial for measuring temperature, density, magnetic field strength, and plasma shape. Fusion experiments are characterised by their data generation at extremely high rates, often in the terabyte range for a single

experiment, necessitating advanced data storage and processing infrastructure.

Once collected, the raw data undergoes processing to filter out noise, calibrate measurements, and combine information from multiple sources. Advanced computational methods, including machine learning and artificial intelligence, are increasingly vital for analysing this data.

This data acquisition is not an isolated process; it is intimately connected with the control systems of the fusion device. Real-time data feedback is critical for controlling plasma conditions, maintaining stability, and optimising the reactor's performance.

Data sharing becomes a key component due to the complexity and cost associated with fusion experiments. International research teams often share data, necessitating collaborative platforms and standardised data formats for effective global research collaboration.

The primary challenges in this domain include managing large volumes of data, ensuring the integrity of the data, meeting the demands of real-time processing, and integrating diverse sets of data from various diagnostic tools. Data acquisition in nuclear fusion melds cutting-edge sensor technology, high-performance computing, and sophisticated data analysis to unravel and control the intricate process of nuclear fusion.

3.3.1 MDSplus

MDSplus (Model Driven System plus) [49] is an advanced, open-source data management system designed primarily for scientific data, especially in nuclear fusion and plasma physics research. It efficiently handles the large and complex data volumes characteristic of fusion experiments, seamlessly integrating data acquisition, analysis, and archiving. The system is structured to store data in a hierarchical tree format, optimising the organisation and retrieval of diverse, multi-dimensional data sets. This framework supports a broad spectrum of data types and formats, from time series and images to waveforms and experiment-related metadata, making it adaptable for various experimental needs.

One of the critical strengths of MDSplus is its facilitation of real-time data acquisition and analysis, a crucial aspect for fusion experiments where immediate feedback is essential for experimental adjustments. This system is integrated with diagnostic systems and instruments and compatible with numerous diagnostic devices and sensors, enhancing its versatility in fusion research. It offers tools and libraries for data analysis and visualisation, as shown in Figure 3.2, aiding researchers in deriving

rapid insights and informed decisions.

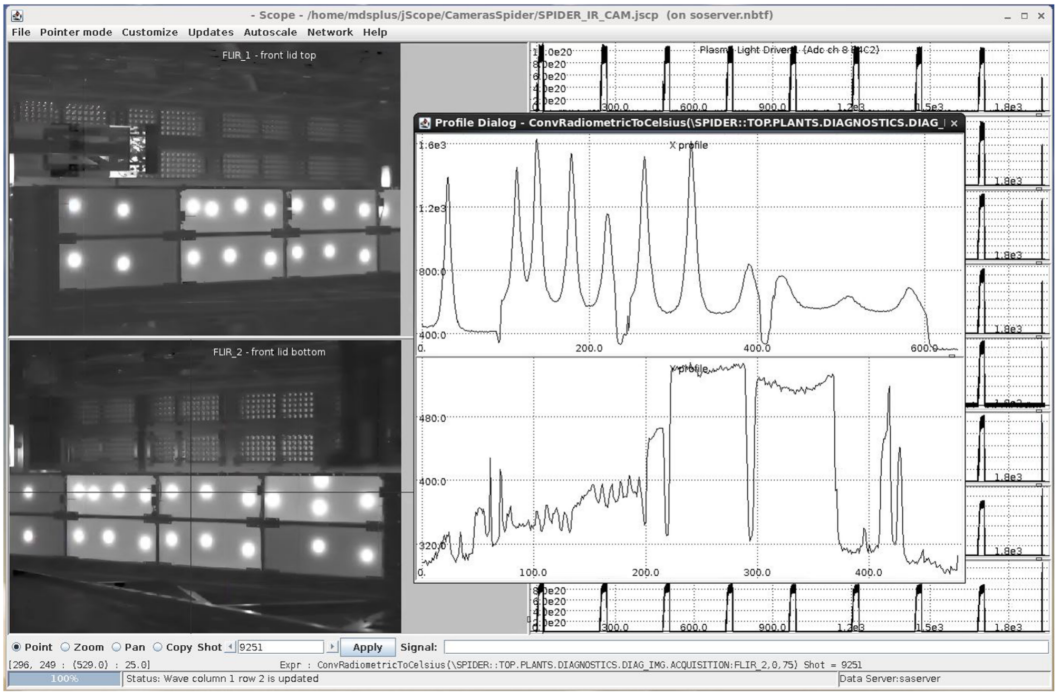


Figure 3.2: Example of complex real-time data visualisation using the jScope tool of MDSplus during a SPIDER experimental session.

The collaborative nature of MDSplus is highlighted by its easy sharing and remote access capabilities. It is essential for the globally collaborative field of fusion research, allowing scientists worldwide to access and analyse data efficiently. As an open-source project, MDSplus encourages community-based development and customisation, supported by an active user and developer community that provides resources, support, and documentation.

Continuously evolving, MDSplus is, as mentioned, a staple in fusion and plasma physics, having a track record of successful utilisation in major fusion experiments like TCV, EAST, RFX-mod2, the NBTF and DTT.

3.4 Slow Control and Plant Supervision

Slow control systems in fusion experiments are dedicated to maintaining and regulating steady-state processes, ensuring long-term stability. They manage essential parameters, like temperature, pressure, and gas flow rates, to guarantee the experiment's success. Continuous data logging provides a historical record of conditions

and performance for analysis. Slow controllers perform active control for non-time critical components such as vacuum and cooling systems and are commonly implemented using PLCs. While these devices are real-time, their cycle time is usually higher (in the order of tens or hundreds of ms) than those of "hard real-time" devices used in fast control operations. Therefore, it is possible to use the term "soft real-time" when referring to them.

Safety is a top priority, with slow control systems implementing safety mechanisms, interlocks, and response protocols to protect equipment and personnel. These systems work closely with Instrumentation and Control (I&C) systems, with I&C systems handling plant control and rapid adjustments.

Historical data collected by slow control is analysed to identify trends and enhance experiment efficiency. Long-duration experiments, particularly in steady-state fusion devices, rely on slow control to maintain consistent conditions for meaningful results.

Plant supervision systems oversee the fusion facility, providing comprehensive oversight and safety measures. They work with slow control, I&C, and other systems to ensure the plant's safe and efficient operation. During anomalies or equipment malfunctions, plant supervision systems detect issues and trigger appropriate responses, such as system shutdowns or alarms.

Data analysis and reporting are integral to plant supervision systems, offering insights into plant operation, performance, and safety. These systems also play a critical role in coordinating emergency responses during critical incidents and prioritising personnel and facility safety.
A coordinated effort between slow control, I&C, and plant supervision systems is essential to maintaining stability, safety, and efficiency in the experimental environment.

3.5 Supervisory Control and Data Acquisition (SCADA)

A SCADA system is a complex framework utilised predominantly in industrial processes for monitoring, controlling, and acquiring data from various devices and sensors. It plays a pivotal role in industrial automation, enabling the efficient operation of large-scale processes across diverse sectors like energy, manufacturing, water treatment, and transportation.

At its core, a SCADA system integrates several components to manage industrial processes remotely and in real time. It consists of hardware and software elements. The hardware typically includes sensors, actuators, and remote terminal units (RTUs) distributed throughout the industrial environment. These RTUs collect data from sensors, which can be anything from temperature readings to flow rates and execute control actions sent from the central SCADA system. This data is then communicated to the central control system, often using a variety of communication protocols and networks, which could range from wired connections to wireless systems and even satellite communication in more remote or expansive installations.

The software component of SCADA is multifaceted. It includes a Human-Machine Interface (HMI), where operators can visualise process data, often in real-time, and make informed decisions or adjustments to the system. This interface is designed to be intuitive, displaying data graphically, such as trends, schematics, or digital dashboards. The system also encompasses advanced data processing and management capabilities, including data logging, alarm management, and reporting functionalities. These features enable operators to detect and respond to anomalies swiftly, ensuring the smooth operation of industrial processes.

A crucial aspect of modern SCADA systems is their ability to provide extensive control capabilities, including basic control tasks, complex algorithmic control, and automation of processes. Through advanced programming and integration with other industrial systems like PLCs and Distributed Control Systems (DCS), SCADA systems can automate workflows, improving efficiency and reducing manual intervention.

Regarding architecture, SCADA systems have evolved from monolithic to more distributed models. Earlier systems were often centralised, with all components tightly integrated and dependent on a single mainframe computer. Modern systems, however, leverage networked configurations, often employing cloud computing and Internet of Things (IoT) technologies. This shift allows for greater scalability, flexibility, and resilience and the ability to integrate with other business and operational systems, facilitating a more holistic approach to industrial management.

Security in SCADA systems is paramount, especially given the critical nature of the processes they control and the potential for cyber threats. Modern systems in-

corporate robust security measures, including network security, application security, and physical security protocols. Encryption, firewalls, regular security audits, and adherence to industrial cybersecurity standards are all integral to protecting these systems from unauthorised access and cyberattacks.

SCADA systems have become fundamental in large-scale scientific experiments. They are crucial in managing data acquisition, process monitoring, and control, transcending their traditional role in industrial automation due to their precision and adaptability.

For example, they are integral in environmental research, providing real-time data from sensors that track temperature, humidity, atmospheric pressure, and pollutants. This information is vital in assessing and understanding environmental changes.
In physics, large-scale experiments like those in particle accelerators, including CERN (Conseil Européen pour la Recherche Nucléaire)'s Large Hadron Collider (LHC), depend on SCADA systems for managing complex machinery and monitoring essential parameters, ensuring safe and efficient operations.
In astronomy, SCADA systems control telescopes and manage data from space missions while monitoring environmental conditions that could impact observations. They oversee bioreactors and other critical equipment in bioengineering and pharmaceutical research, maintaining optimal cell growth and drug development conditions.
Energy research, particularly in renewable sources like solar and wind, also benefits from SCADA systems. These systems aid in gathering and analysing data to enhance the efficiency and design of solar panels and wind turbines.
In nuclear fusion research, SCADA systems are pivotal in controlling and monitoring complex processes and machinery. Fusion experiments, such as those in ITER or other tokamaks, rely on SCADA systems to oversee critical parameters like plasma temperature, magnetic field strength, and ion density. These systems ensure the precision and safety required in such high-stakes environments.
The importance of SCADA systems in these large-scale facilities has risen because of their size increase and because their industrial component has become more and more relevant. As research moves towards demonstrating the feasibility of energy production, experiments must become more and more reliable and, therefore, closer to industrial plant systems.

Here are some concrete examples of commonly used SCADA systems in scientific settings:

1. EPICS: EPICS [50] is widely used in particle physics laboratories, such as the SLAC National Accelerator Laboratory, originally the Stanford Linear Accelerator Center (SLAC) and the European Spallation Source (ESS). EPICS provides real-time data monitoring and control of the intricate machinery in these large-scale physics experiments.

2. LabVIEW (Laboratory Virtual Instrument Engineering Workbench): Developed by National Instruments, LabVIEW [51] is a popular system in various scientific research domains, including engineering, bioengineering, and environmental science. Its graphical programming environment is suited for managing experiments that require real-time data analysis and control, such as monitoring environmental parameters or controlling biomedical devices.

3. Siemens SIMATIC WinCC OA: In fields like renewable energy research, particularly in solar and wind power experiments, WinCC OA [52] is used for monitoring and controlling processes. It helps collect and analyse data and optimise the performance of solar panels and wind turbines.

4. Wonderware by AVEVA: Wonderware [53] SCADA systems are employed in various scientific applications, including pharmaceutical research and biotech laboratories. They are used for monitoring bioreactors, managing production processes in pharmaceutical manufacturing, and ensuring quality control.

5. TANGO Controls: TANGO [54] is a robust and flexible open-source distributed control system widely used for managing complex experiments, particularly in synchrotron light sources and large-scale research facilities like the European Synchrotron Radiation Facility (ESRF) in France. It is designed for continuous operation and offers real-time control, data acquisition, and monitoring, essential for scientific research. TANGO's architecture supports scalability, controlling thousands of devices. Its client-server model and comprehensive device control and monitoring tools suit it for various scientific applications.

3.5.1 EPICS

The Experimental Physics and Industrial Control System (EPICS) [50] is a sophisticated framework for creating distributed control systems primarily used in scientific

environments such as particle accelerators, telescopes, and nuclear fusion experiments.

Its extensive software components and tools characterise EPICS, enabling application developers to build comprehensive control systems. The architecture of EPICS facilitates the management of complex scientific facilities involving networks of computers that range from tens to hundreds. These computers are interconnected to provide centralised or remote control and feedback for various parts of the devices used in these facilities. EPICS base, the core of the EPICS framework, supports an arbitrary number of target systems, IOCs (input/output controllers), host systems, and OPIs (operator interfaces) of various types. This extensibility is crucial for the adaptability of EPICS to different scientific applications.

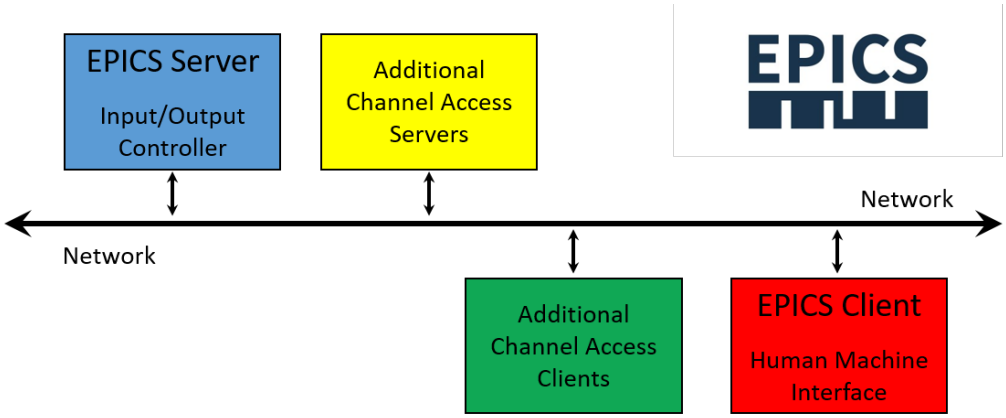


Figure 3.3: Diagram of the EPICS network architecture.

For effective communication across the interconnected computers, EPICS employs Client/Server and Publish/Subscribe methods. IOCs, often running on embedded platforms like VxWorks [55] and RTEMS [56] or as soft IOCs on the host platform, handle real-world I/O operations and local control functions. They use EPICS-specific network protocols, Channel Access (CA) [57] and pvAccess (pvA) [58], designed for high bandwidth and soft real-time networking applications, enabling EPICS to support distributed control systems in a modular and scalable way. Figure 3.3 presents a diagram of the EPICS network architecture.

EPICS's open-source nature has been instrumental in its widespread adoption and continuous innovation within the scientific community, extending its applications to various fields such as astronomy, materials science, and energy research. The system's versatility and adaptability to various scientific applications under-

score its importance in experimental physics and industrial control systems.

Some of the most important experiments and projects using EPICS include:

1. Particle Accelerators: EPICS is extensively used in particle accelerator facilities. These facilities often require precise control and monitoring of numerous parameters, such as beam intensity and alignment. Examples include the Advanced Photon Source (APS) and the SLAC laboratories.

2. Synchrotron Light Sources: Synchrotrons, a type of particle accelerator, also rely on EPICS for control and data acquisition. Facilities like the Diamond Light Source in the UK, the Advanced Light Source (ALS) in the US, and the Australian Synchrotron use EPICS to manage their complex operations.

3. Telescopes and Astronomical Observatories: EPICS is employed in some telescopes and astronomical observatories for controlling telescopes and managing data. The Australian Square Kilometre Array Pathfinder (ASKAP) is among these facilities.

4. Nuclear Fusion Experiments: In the field of nuclear fusion, experiments like KSTAR (Korean Superconducting Tokamak Advanced Research) and the ITER project utilise EPICS for controlling and monitoring various subsystems and experimental processes.

3.5.2 WinCC OA

WinCC Open Architecture (WinCC OA) [52] is a versatile and powerful software platform developed by Siemens designed for industrial automation and process control. This platform stands out for its open system architecture, scalability, and comprehensive features, making it an ideal SCADA choice for various industries, including energy, water treatment, transportation, and manufacturing.

The core features of WinCC OA are extensive and diverse. It excels in scalability and data handling, capable of managing large volumes of data. It is essential for controlling and supervising complex applications like wind farms, oil refineries, and urban mass rapid transport systems. The software supports distributed systems, enabling monitoring and control over multiple locations. Additionally, WinCC OA offers a user-friendly interface for visualisation and interaction with process data. It also supports various communication protocols and integrates well with other Siemens products and third-party systems.

Concerning reliability and redundancy, WinCC OA ensures continuous system operation and maintains data integrity with high levels of redundancy and reliability. It features robust security measures to protect against unauthorised access and cyber threats.

Another significant feature is the CTRL language for scripts. The CTRL scripting language blends the familiar syntax of C programming with advanced features suitable for industrial automation. It excels in multitasking, allowing simultaneous processing of diverse tasks. CTRL scripts are adaptable, catering to both passive and active graphical elements. Passive scripts typically run during panel initiation, focusing on display attributes, while active scripts interact dynamically with user inputs like mouse clicks. In terms of execution, CTRL offers versatility. It can spontaneously respond to changes in data point attributes or be triggered by user actions, like pressing a button. The language's precision in addressing data point attributes is a key strength, enabling detailed control and manipulation of system elements.

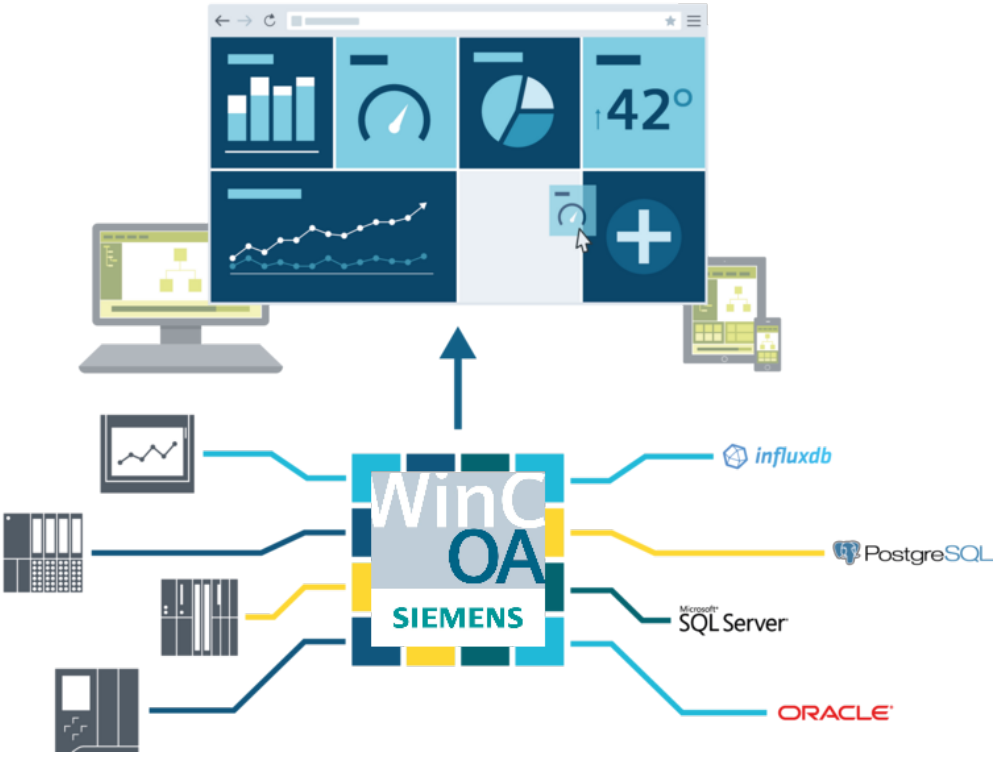


Figure 3.4: Diagram of the versatile WinCC OA Architecture, allowing for handling HMIs, PLCs, Data Acquisition and Archiving systems in a scalable and distributed fashion.

CTRL's repertoire of data types is vast, encompassing everything from basic integers and strings to more complex types like dynamic arrays and enums. This variety allows for the creation of nuanced and multifaceted scripts. The dynamic arrays, adept at managing multiple values within a single data type, are particularly noteworthy and bolstered by a suite of functions for their manipulation. This combination of user-friendly syntax, comprehensive data handling, and responsive execution makes CTRL a powerful tool in industrial process control.

WinCC OA's integration with OPC UA (Open Platform Communications Unified Architecture) [59] is notable, ensuring secure and reliable data exchange and interoperability with various industrial devices and systems. This integration supports data aggregation, filtering, historical data access, cross-platform compatibility, and secure communication. The software also excels in handling data points and configurations (configs). Datapoints in WinCC OA represent data or control points within a system. Master data points act as templates for configuring multiple devices of the same type, using *PowerConfigs* to streamline the setup process. Datapoint configs have unique properties that can be created, edited, or removed via scripting, including peripheral address, alert class, alert handling, archive settings, admitted value range, and more.

WinCC OA features native S7 drivers for connecting to Siemens S7 PLCs. It supports various functionalities like sending, polling, receiving spontaneous data from the PLCs and receiving and acknowledging S7 alarms. WinCC OA also integrates InfluxDB for archiving, offering options for the database name, backup path, and dynamic data point name filter splitting settings. It includes settings for the backup host connection string and database startup timeout, ensuring automatic backups by the InfluxDB backend.

WinCC OA is a comprehensive solution for modern industrial and process control environments. It is adept at handling complex automation tasks with its flexibility, security features, and integration capabilities with technologies like OPC UA and InfluxDB, marking it a key player in industrial automation.

3.6 An example of CODAS in Fusion Experiments: ITER's CODAC Core System

The CODAC (Control, Data Access and Communication) Core System [60] serves as the development and interface kit for plant systems I&C, leveraging the widely adopted open-source software, EPICS, and CSS (Control System Studio). It operates on various components, including Mini-CODAC, Plant System Host (PSH), and Fast Controllers, facilitating interactions with Slow Controllers (PLC) and third-party controllers. CODAC is pivotal as the central control system overseeing ITER's operations. Within ITER, diverse plant systems are integral, each empowered by local I&C, denoted as Plant System I&C. These I&C units comprise local controllers falling into three categories: standardised programmable logic controllers (PLCs) or Slow Controllers, standardised rack-mounted computers known as PCF (Plant Controller, Fast), and non-standardised controllers referred to as POC (Plant Other Controllers). Additionally, each Plant System I&C incorporates a PSH to handle standard functions without specific software requirements or signal inputs/outputs. The physical architecture of the plant system I&C is visually represented in Figure 3.5.

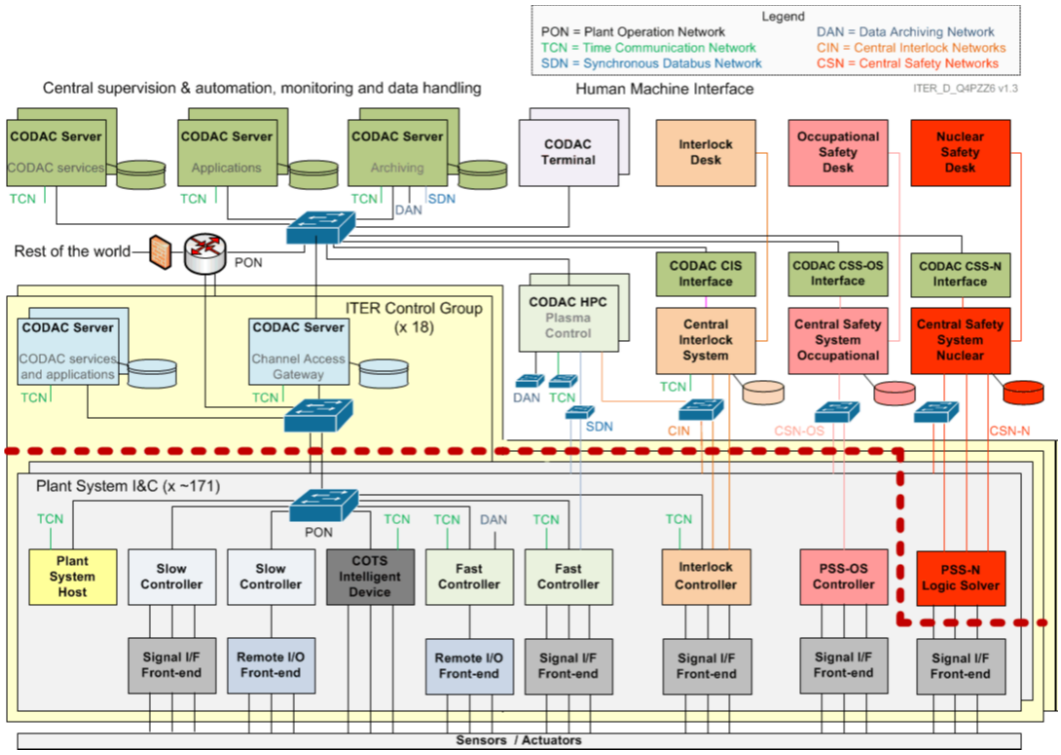


Figure 3.5: CODAC Core System Plant System I&C Physical Architecture.

During development and testing, the CODAC central infrastructure, encompassing servers and operator terminals, is substituted by a dedicated computer known as Mini-CODAC. This specialised setup efficiently fulfils a subset of the core CODAC functions, optimising the development process. The CODAC Core System, distributed by the ITER Organisation, is a comprehensive software package for developing plant system I&C. It caters to the diverse I&C computers' needs and furnishes a development and testing environment that meets ITER's stringent requirements. To ensure a seamless experience, the CODAC Core System comes bundled with the Red Hat Enterprise Linux (RHEL) Operating System, pre-configured to accommodate the specificities of the CODAC Core System.

The foundation of the CODAC Core System lies in the EPICS open-source distributed control system software platform. EPICS, widely embraced in experimental physics, features a framework and utilities developed and maintained by its user community, including ITER. RHEL stands as the standard operating system for the CODAC Core System, finding applications across mini-CODAC, PSHs, Fast Controllers, CODAC Terminals, and CODAC Servers, all running officially supported RHEL versions by Red Hat.

EPICS Base is central to the CODAC Core System's operation, integral to the CCS distribution, and essential for all system profiles. The EPICS toolkit is the basis for controlling processes executed on Fast Controllers and PSHs. Key EPICS communication protocols, CA and pvA provide the standard means to access plant system I&C data across the Plant Operation Network (PON). The latest iteration of CSS, EPICS 7, extends EPICS Base version 3.x, introducing pvA modules, which include the pvData and pvA libraries for structured data management over the network using the pvA network protocol.

Within this comprehensive ecosystem, various major EPICS components play crucial roles, such as the Sequencer for executing state machine programs, IOC logs for error and trace logging, auto-save for automated configuration variable saving and reloading, and CA Gateway, which enables CA clients to access servers on diverse subnetworks. Features like Multi-Core Utilities also optimise EPICS thread management by configuring CPU core isolation and priorities. Java implementations of CA and pvA are incorporated for Java-based CODAC clients, while Python developers benefit from Python wrappers for CA and pvA.

Furthermore, the CODAC Core System encompasses valuable tools like the IOC Monitor/System Monitor for health parameter monitoring, the Visual Database Configuration Tool for configuring EPICS databases via a graphical interface, and ASYN, a versatile driver framework for EPICS Device Support. Other modules like Stream Device enable data exchange between EPICS records and hardware, Area Detector handles image acquisition devices, and Device Support modules for Modbus and OPC UA facilitate interfacing with specific devices. Finally, pyDevSup offers a generic device support module for device integration through Python scripts. In contrast, Device Support modules tailored to the Fast and Slow Controller catalogues cater to a wide array of supported hardware. The distribution includes a PostgreSQL relational database with the following databases configured:

- SDD, which stores plant system I&C data.
- Alarm, dedicated to storing alarm server configuration and alarm history.
- Archive, designed to store archive engine configuration and archived data.
- Logbook, utilised for storing events.

This comprehensive suite empowers the ITER project with the necessary tools to ensure the seamless operation of its diverse plant systems.

CHAPTER 4

SPIDER AND MITICA CODAS - A DETAILED ANALYSIS

This chapter delves into the Control and Data Acquisition System (CODAS) of the SPIDER and MITICA experiments at the ITER NBTF (Neutral Beam Test Facility). Section 4.2 examinesSPIDER's CODAS, outlining its structural division between the central system and plant systems and discussing its data management strategies and control challenges in nuclear fusion experiments. Section 4.3 focuses on MITICA's CODAS, exploring its supervisory control, real-time control strategies, and the network infrastructure essential for effective data communication and management. The section sheds light on the unique aspects of MITICA's system design, including integrating various control and safety mechanisms. The chapter offers insights into the complexities and solutions in real-time control and data acquisition, underlining the significance of these systems in advancing nuclear fusion research. The concluding section synthesises key learnings from the SPIDER and MITICA experiments, highlighting their contributions to fusion technology development.

4.1 Orchestrating Complexity: The CODAS of SPIDER and MITICA

The CODAS of the SPIDER and MITICA projects are critical components in the ITER NBTF's quest for nuclear fusion advancements. Each system, developed for a specific purpose within the broader context of fusion research, exhibits a unique set of features and challenges pivotal to these high-stakes experiments' success.

In the SPIDER project, the CODAS is intricately designed to manage the complexities of the beam source, a key element in ITER's heating neutral beam injectors. Its architecture combines control, interlock, and safety systems, reflecting a commitment to reliability and precision. This system's design ensures effective operational management and navigates the complexities of extensive data management and network infrastructure.
On the other hand, MITICA's CODAS showcases a unique approach tailored to meet its specific experimental requirements. It encompasses comprehensive supervisory and real-time control strategies, essential for maintaining the experiment's integrity and success. The project's network infrastructure further exemplifies MITICA's forward-thinking approach, ensuring robust data communication and effective management.

The subsequent sections provide a detailed exploration of these CODAS systems, shedding light on their structural components, functional intricacies, and the innovative solutions employed to overcome the challenges inherent in fusion research. Through this analysis, it is possible to gain a deeper understanding of these sophisticated systems' roles in advancing the nuclear fusion field.

4.2 SPIDER CODAS

SPIDER (Source for Production of Ion of Deuterium Extracted from Rf plasma) is the integral beam source for ITER's heating neutral beam injectors [17]. Its primary goal is to enhance the functionality and efficiency of the source, making it a critical component of the two major experiments at the ITER NBTF in Padova, Italy [61, 16]. This facility began its operations in June 2018, achieving more than 5000 pulses since starting their start [62, 63].

The control system of SPIDER is designed to ensure reliable operation, safeguard investments, and maintain personnel safety. These objectives are achieved through three distinct Instrumentation and Control (I&C) systems: the SPIDER CODAS [64, 65], the Central Interlock System (CIS) [66], and the Central Safety System (CSS) [67, 68]. These systems, developed over a period spanning from 2014 to 2020, were based on the specifications for ITER's HNBs [69], the design of the SPIDER plant systems, and insights from the construction and operation of similar systems in the field of fusion [70, 71, 72].

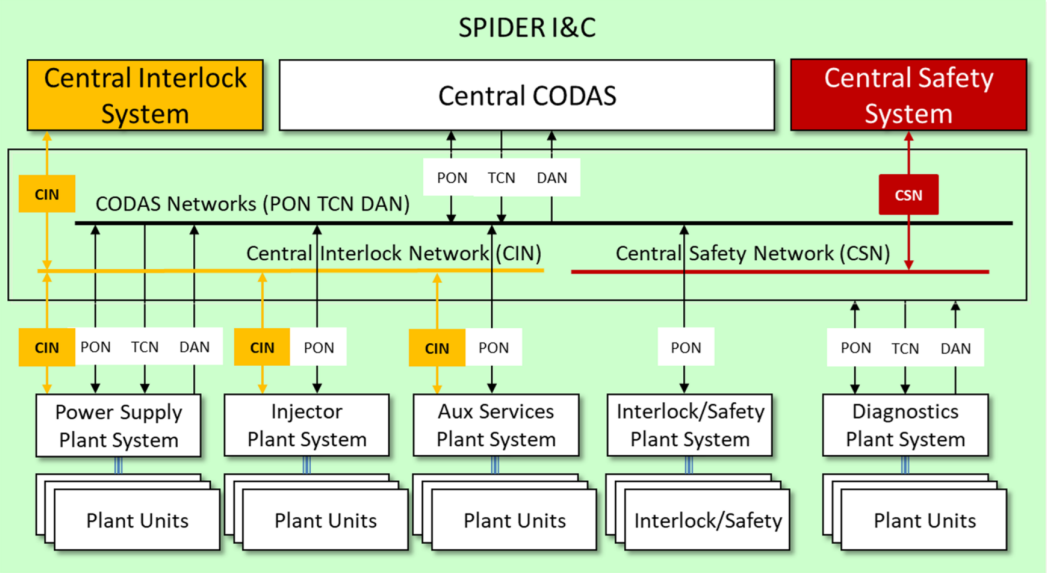


Figure 4.1: The SPIDER I&C architecture, comprising the three vertical tiers and the three horizontal layers.

Aligning with the highest standards in systems engineering and the ITER methodology, SPIDER's control, interlock, and safety functions are managed by three autonomous systems. The design philosophy aimed to reduce system complexity: the control system is the most complex, followed by interlock and safety. On the other hand, reliability follows the inverse trend, increasing from the control system to the interlock and the safety ones. The SPIDER I&C architecture is organised into three vertical segments - control, interlock, and safety - and three horizontal layers - the central system, plant systems, and plant units - illustrated in Figure 4.1. Each plant unit, typically commissioned under a single contract, is responsible for either a singular function or a group of related functions. For instance, the SPIDER gas and vacuum system, which handles vacuum pumping and gas injections, is one such unit. Similarly, the power supply system for the ion source and extraction is an-

other unit responsible for various tasks, including H_2/D_2 radiofrequency ionisation, negative ion extraction, and plasma grid magnetic filtering. These units form the larger plant systems, collectively supporting common objectives like power supply, injector components, auxiliary services, interlock/safety interfaces, and diagnostics.

4.2.1 CODAS Structure

The SPIDER CODAS system is divided into two main sections: Central CODAS and Plant System CODAS, linked via the SPIDER network infrastructure. The Plant System CODAS manages interactions with plant systems and units, ensuring seamless integration. On the other hand, Central CODAS is responsible for broader, global functions that include managing data storage and access, ensuring timely communication and synchronisation across the system, and overseeing high-level monitoring and supervision. It also handles advanced Human-Machine Interface operations and data display. In its role, the Plant System CODAS is specifically focused on the supervision and monitoring of plant operations, along with managing real-time control and acquiring data.

4.2.2 Plant System CODAS Overview

The Plant System CODAS constitutes the segment of CODAS that is in charge of directly overseeing the SPIDER plant systems and units. The plant's primary components directly linked to SPIDER CODAS are detailed in Table 4.1. In compliance with ITER guidelines defined in the Plant Control Design Handbook (PCDH), the choice of CODAS technologies adhered as much as possible to the ones chosen by ITER CODAC, focusing on those that are well-established and proven [73]. This approach was particularly pertinent to selecting control and data acquisition hardware, including Siemens PLC slow controllers, PC-based fast controllers, and National Instruments PXI-based data acquisition modules. Since the ITER CODAC specifications for the Time Communication Network (TCN) and the Data Archiving Network (DAN) high-performance networks were yet not available, SPIDER-specific adaptations were developed and implemented, as detailed in Section 4.2.4.

4.2.3 Central CODAS Overview

Central CODAS executes general system supervision based on state machine logic, manages an advanced human-machine interface, and handles short-term and long-term data archiving. It also includes top-tier infrastructure such as the SPIDER control room and the NBTF data centre. The data archiving function distinguishes

Plant System	Plant Unit ID	Plant Unit Description
Power supply	ISEPS	Ion source and extractor power supply
	AGPS	Acceleration grid power supply
	MV	22 kV power distribution board
	TPU	Thermal protection unit: grid, vessel, transmission line and beam dump thermal sensors
Injector	CS	Caesium ovens
	PL	Plasma light
Auxiliary services	GVS	Gas and vacuum system
	CP	Cooling system
Interlock/safety	CIS	Central interlock system
	CSS	Central safety system
Diagnostics	DS	Spectroscopy
		Imaging
		Tomography
		Neutrons
		Electrostatic probes
		Cavity ring-down spectroscopy
		Calorimetry
		Instrumented calorimeter (STRIKE)

Table 4.1: List of SPIDER main components.

between short-term and long-term storage, with variations in data access speed and storage capacity. For immediate access, short-term data storage employs solid-state drives with a capacity of 2 TB, deemed adequate at the planning stage for holding up to two days' worth of SPIDER operational data. In contrast, the long-term data archive has been designed with a capacity of 100 TB, accommodating the anticipated volume of data from SPIDER operations, the average operational time per year, and a projected operational span of up to four years. From this data, it could be possible to deduce that SPIDER operates only 25 days per year; however, the 2 TB capacity constitutes an upper bound in case of heavy-duty experimental sessions: the typical shot dimension data is reported in Section 4.2.7.

4.2.4 Network Infrastructure

SPIDER's network infrastructure facilitates communication between its central and plant systems through several specialised networks.
The Plant Operation Network (PON) transmits operational data and handles low-bandwidth data acquisition for SPIDER. Concurrently, the DAN manages the high-bandwidth data flows crucial for massive data acquisition tasks. Both PON and DAN are built on standard Ethernet technology [74]. PON primarily operates on the EPICS (Experimental Physics and Industrial Control System) protocol for Channel Access (CA) over TCP-IP multicast [50]. In contrast, DAN utilises mdsip on TCP-IP, aligning with the MDSplus data transfer protocol [49]. This choice does not align with ITER's CODAC EPICS standard since they were not defined at the time of its implementation.

The TCN plays a vital role in synchronising data acquisition. It relies on a GPS-synchronized grand-master clock [75], supplemented by industrial timing modules for transmitting and receiving synchronised clock signals [76] and a PXI system for converting absolute time to relative time [77]. This absolute time, generated by the grand master clock, is distributed using the PTP (Precision Time Protocol) [78, 79]. Relative time begins from a predefined start trigger, marking the zero time of a beam pulse, and is disseminated through the plant using a 1 MHz synchronous clock signal transmitted via optical fibres.

The Central Interlock Network (CIN) and the Central Safety Network (CSN) also form the backbone of SPIDER's safety infrastructure. The CIN is based on the Profinet standard. The CSN utilises Profibus, specifically chosen for its compliance with functional safety applications up to SIL (Safety Integrity Level) 3 as per the IEC 61508 technical standard. Profinet, also compliant up to SIL 3, was not qualified at the time of the safety system's initial design phase.

4.2.5 Software Components

SPIDER CODAS's software environment is distinguished by its cohesive integration of three major collaborative open-source frameworks. These include the CODAC Core System [60], which is used for supervisory control, monitoring, and facilitating the human-machine interface; MDSplus [49], which is essential for data management; MARTe2 [48], designed for rapid real-time control.
The CODAC Core System operates on the Red Hat Enterprise Linux (RHEL) platform (with Linux MRG-R as an option) [80]. It incorporates a variety of software

frameworks, such as EPICS for slower control processes, Control System Studio for the HMI, and other standard software technologies like MAVEN, Eclipse, and PostgreSQL.

Significant efforts have been made to integrate seamlessly EPICS, MDSplus, and MARTe2, ensuring clear and efficient communication between these systems [81, 82]. This integration allows for multiple functionalities. Both EPICS and MARTe2 can access MDSplus for reading and writing data. Within EPICS, data from MDSplus can be directly accessed in the EPICS Databases through specific records or via a customised version of the EPICS Data Archiver, which stores signal trends in MDSplus pulse files [83]. Meanwhile, real-time signals generated by MARTe2 can be archived in MDSplus using a specialised MARTe2 DataSource component. This setup allows for real-time operations to be decoupled from data access processes. MARTe2 typically accesses parameters stored in MDSplus pulse files before the system enters its real-time operational mode.

4.2.6 Real-time Breakdowns Management

Effective real-time control is essential in neutral beam injectors to address breakdowns, commonly flashovers between grids or between grids and grounded components. These are frequent due to regular high-voltage operations and limited insulation space. An illustrative breakdown sequence on a SPIDER grid (specifically the extraction grid) is depicted in Figure 4.2. Detecting these breakdowns occurs within the plant's power supply units. It is rapidly communicated to the CIS. It is important to note that a breakdown does not indicate a fault but rather a regular operational occurrence. Upon detecting a breakdown, the CIS informs the fast controller of the power supply, which then adjusts the waveform references to manage the power supply system effectively.

As demonstrated in Figure 4.2, at the initial moment (t_0), a negative voltage is gradually applied to the extraction grid. When a breakdown is detected at time t_s, the voltage reference for the extraction grid's power supply is immediately reset to zero. Following a brief time interval (usually about 20 ms), the fast controller initiates the recovery of the power supply (at time t_e), employing a preset waveform to modulate the voltage of the extraction grid, thus reducing the chances of subsequent breakdowns. This breakdown management protocol applies to the acceleration grid and radiofrequency power supplies. Moreover, the fast controller sends information about the breakdown across the TCN, activating a high-frequency, event-based data

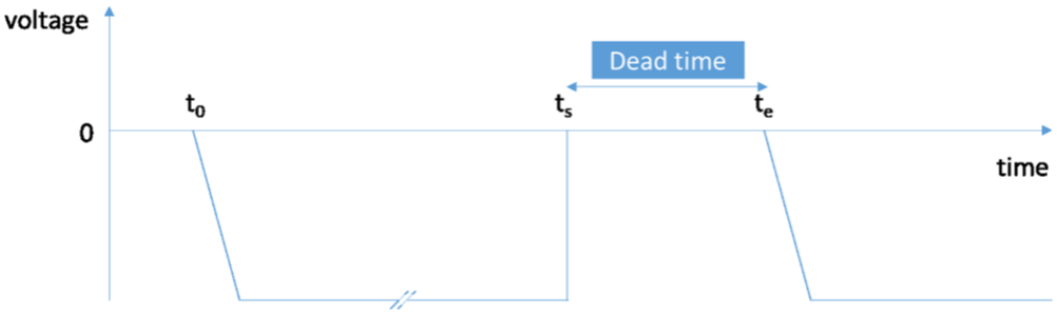


Figure 4.2: Example of a breakdown event on the extraction grid. At time t_0, a negative voltage is progressively applied to the extraction grid. At time t_s, a breakdown occurs.

acquisition mode. This procedure ensures data is gathered at an elevated sampling rate during a specific time window centred around the breakdown event. The interlock system guarantees swift response to breakdowns and serves as a support system for the fast controller's operations. Fast control is implemented through the MARTe2 real-time framework.

4.2.7 Data Acquisition System

SPIDER boasts an extensive array of diagnostic tools designed to explore and analyse various phenomena within the beam source, such as plasma formation, distribution of Caesium, beam divergence, and the electron/ion ratio. In addition, it is essential to gather data concerning electrical events in the power supply systems and circuits, along with visual and infrared (IR) imagery. For data acquisition, SPIDER utilises the MDSplus system operating on Linux servers. This setup allows for the collection of analog signals via analog I/O modules [84, 85]. Concurrently, image capture from Basler and Forward-Looking Infrared cameras is integrated into CODAS using the GenICam software [86], which supports the Gigabit Ethernet Vision (GigE Vision) protocol [87].

The range of sampling frequencies in SPIDER varies widely, from a few Hz for most temperature readings to several MHz for rapid phenomena. Advanced applications like Cavity Ring-down Spectroscopy demand even higher frequencies, reaching hundreds of MHz. Image capture is versatile, with frames ranging from 1 to 25 per second, resolutions spanning from 600×420 pixels for IR to 1920×1200 for visible images, and pixel depths of 8-14 bits. On average, a SPIDER beam pulse results in data accumulation of around 130 GB, with a maximum capacity of 630 GB, contributing to total data storage of 33 TB at present. This total storage amount is lower than initially anticipated due to various factors, including a short duty cycle in

the early operational period, gradual integration of diagnostic tools, frequent downtime for engineering updates, and reduced or halted activity during the COVID-19 pandemic. Data generated during beam pulses are initially stored in a short-term storage system and transferred to long-term storage and a backup system at the end of each operating day.

Since SPIDER is a long-term experiment with beam-on times reaching up to 3600 s, managing data acquisition and access requires careful planning to avoid overwhelming data volumes and computational demands. Strategies for managing long-duration experiments include optimising data storage, balancing real-time data reading and storage, tuning resources, and monitoring system parameters effectively. These approaches are elaborated in [88].

Concerning data visualisation, SPIDER employs the MDSplus GUI tool, jScope, primarily for post-pulse data analysis, which is unsuitable for real-time display. SPIDER uses the Grafana tool for live data visualisation [89], which integrates various data sources, including those exported through selected EPICS process variables and subsampled data from ADC devices during SPIDER pulses. This data is streamed to the system through MDSplus events, with a specially developed Grafana plugin updating the displayed graphs accordingly.

4.3 MITICA CODAS

As explained in Section 2.3.3.2, MITICA's [14, 15, 16] plant breakdown structure presents several differences compared to ITER's HNB [17]. For example, it integrates central I&C systems, including the CODAS, CIS, and CSS, which are distinct and independent in ITER.
Figure 4.3 elucidates the structure of MITICA's plant systems. The HNB (Heating Neutral Beam Injector) plant system in MITICA will encompass all elements found in the standard HNB setup, but MITICA will also feature unique service and diagnostics systems. While the HNB plant system in ITER will adhere to the guidelines and standardisation outlined in the ITER PCDH [73], the central I&C, along with the service and diagnostic systems in MITICA, may not strictly follow these standards.

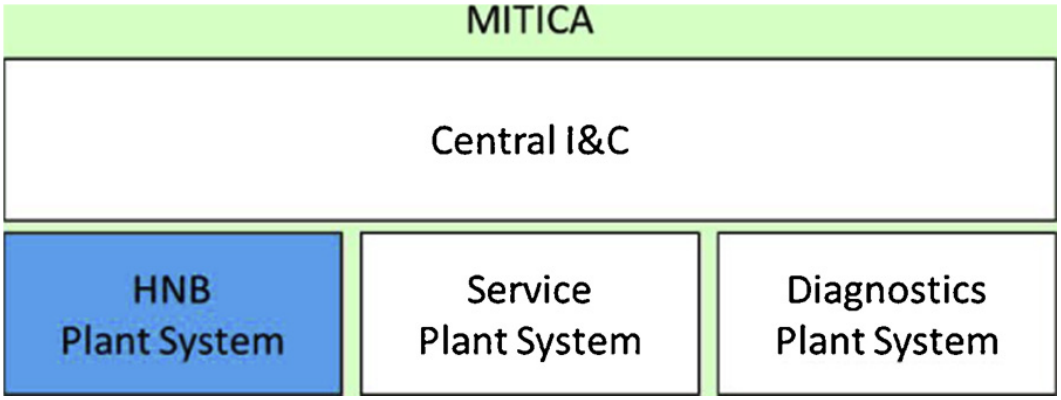


Figure 4.3: Structure of MITICA plant systems.

4.3.1 Control and Data Acquisition System

The CODAS facilitates standard control operations, including supervisory and real-time control, interfacing with the plant system, and managing data through acquisition, storage, and retrieval. Additionally, CODAS will handle all aspects of communication within the system.

4.3.2 Supervisory Control

In the context of MITICA, supervisory control is responsible for coordinating the plant system to ensure the smooth operation of MITICA. This control function operates without needing real-time response and is not critical for reliability. Similar to the approach in ITER, supervisory control in MITICA is structured around various operational states: Global Operating States (GOS), Common Operating States (COS), and MITICA-specific Plant System Operating States (PSOS). The GOS for MITICA, closely mirroring those in ITER, include states such as Long Term Maintenance (LTM), Short Term Maintenance (STM), Conditioning State (TCS), and Beam Operation State (BOS), with Table 4.2 detailing these states and their corresponding sub-states.
MITICA's operations are outlined in terms of PSOS, with a precise mapping to the COS. A mechanism for managing additional GOS sub-states by activating or deactivating plant units will be implemented to enhance flexibility. However, this will only be available during the commissioning phase. These GOS and sub-states play a crucial role in control processes, machine protection, and ensuring personnel safety. Consequently, they will be integrated into the CSS. They will be communicated to the CIS and CODAS.

GOS	Sub	Description
LTM	-	-
STM	Standby	Plant systems are kept in a low energy consumption regime
TCS	Filament-only pulse	Only filaments are tested
	Gas-only pulse	Only gas injection into the ion source and/or neutraliser is tested
	RF pulse	Only RF is tested
	HV without gas	Only high voltage (HV) is tested in high vacuum
	HV with gas	High voltage (HV) is tested along with gas injection
	RID only pulse	Only the residual ion dump is tested
BOS	H beam	Beam operation in H
	D beam	Beam operation in D

Table 4.2: MITICA Global Operating States and sub-states.

4.3.3 Real-time Control

Real-time control in MITICA is focused on executing essential yet non-critical functions within specific time constraints to ensure consistent quality. Essential real-time functions include initiating plasma, matching perveance, managing breakdowns, modulating power, and overseeing power supply systems.
The process of starting plasma involves synchronising the heating of filaments and adjusting biasing in line with the injection of RF power while monitoring the plasma light signals. If plasma formation does not occur, the RF generators must be deactivated quickly, typically within tens of milliseconds. Perveance matching, a crucial step in optimising beam optics, involves adjusting the accelerator voltage based on the ion beam current. This task requires the control cycle to align with the 1 kHz bandwidth of the AGPS (Acceleration Grids Power Supply) power supply.
Breakdown management deals with short discharges between electrostatic acceleration grids, often caused by high electrostatic fields. This phenomenon is typically identified by an overcurrent in a grid power supply. In such instances, all high-voltage power supplies are temporarily shut down for a brief period, usually tens of milliseconds, before the voltage is reapplied in a controlled manner to manage the breakdown.

4.3.4 Plant System Interface

The interface for the plant unit in MITICA will primarily utilise the Ethernet and TCP-IP-based PON. Digital and analog signals will be essential in certain situations, such as enabling command signals, generating reference waveforms for power supplies, and acquiring data signals. Wherever practical, fibre optics will be employed for transmitting data. For converting analog to digital signals and vice versa, commercially available optical link modules will be used for analog signals, and differential and balanced signal channels will be applied for connections within the same room.
The installation of CODAS and CIS hardware on the high-voltage decks of MITICA will be deliberately avoided to ensure swift access to electronics during operations. Adhering to the best Electromagnetic Compatibility (EMC) practices will be a priority to safeguard CODAS and CIS equipment from irradiated and conducted disturbances. The I&C cubicles will be designed per EMC standards, incorporating power EMI (Electromagnetic-Interference) filters, effective shielding and grounding for cable shields, noise reduction, and overvoltage protection for analog ports.

4.3.5 Data Management

Managing data in MITICA encompasses various activities, from initial data gathering to long-term storage, accessibility, and analysis. The anticipated data acquisition rate for MITICA is set to mirror that of SPIDER, around 200 MB/s.
A diverse array of transducers in MITICA generates signals that necessitate data capture at relatively low sampling rates, typically up to tens of Hz. Sensors measuring temperature, pressure, and mass flow fall into this group. Meanwhile, electrical signals from the power supply system and circuits demand a much higher sampling rate, reaching up to several MHz (in case of a breakdown occurrence).
Under normal, steady-state operating conditions, these rapid electrical signals usually exhibit low harmonic content. However, the harmonic content significantly increases during specific occurrences, like breakdowns between acceleration grids or beam-off events. To efficiently handle these variations, MITICA employs an event-driven data acquisition approach [90]. This method enhances the sampling frequency around the time of such events, utilising a high-frequency circular buffer that precisely timestamps each sample. Only filtered, low-frequency signals are extracted from this buffer during stable operational phases. In contrast, the system captures high-frequency samples during events, allowing for a more detailed analysis of these critical moments.

4.3.6 Communication

MITICA is set to integrate ITER's communication technologies [91]. The system will use EPICS CA [57] as a middleware solution for disseminating plant information via the PON.

To enhance performance, MITICA will establish High-Performance Network (HPN) infrastructures. The timing and communication protocols will adhere to the ITER TCN, utilising the precision of the IEEE 1588 protocol (PTP) [78, 79, 92]. For real-time data transmission, MITICA will employ Ethernet-UDP, aligning with the ITER Synchronous Data Network (SDN) model. The SDN, a future standard interface, will facilitate connection with the ITER plasma control system and the HNB's fast controller.

MITICA's network is also designed to handle the distribution of asynchronous events. Data signals, especially those requiring high bandwidth, will be routed to the central database through either mdsip (MDSplus's native data transmission protocol) or the ITER DAN, based on EPICS. Data acquisition drivers will be configured using MARTe2 to support both systems, giving users flexibility in setting up their preferred communication method. Moreover, MITICA's central data archive, based on MDSplus technology, will incorporate a DAN server to receive and store data. This configuration leverages MDSplus's standard technology and optimal performance while ensuring a seamless transition to the data communication protocols used at ITER's HNB.

Figure 4.4 illustrates the block diagram of the MITICA HNB plant system, outlining the integration of a slow controller for supervising the system and a fast controller for executing real-time HNB applications.

4.3.7 Hardware Design

The HNB plant system will implement components from the ITER catalogue, including the integration of slow controllers, such as Siemens PLCs, and fast controllers, like PICMG1.3, PCIe, PXIe, and cRIO. Additionally, various network technologies will be employed following the ITER CODAC catalogue standards, including the PON with 1 Gb/s Ethernet, the TCN using PXIe, the SDN, and the DAN with 10 Gb/s Ethernet.

The services and diagnostics systems of the plant will also follow the ITER catalogue guidelines as closely as possible. However, there will be some deviations in specific diagnostic systems, such as those used for cavity ring-down spectroscopy and visible tomography.

For central control and data acquisition, the Central CODAS will be based on

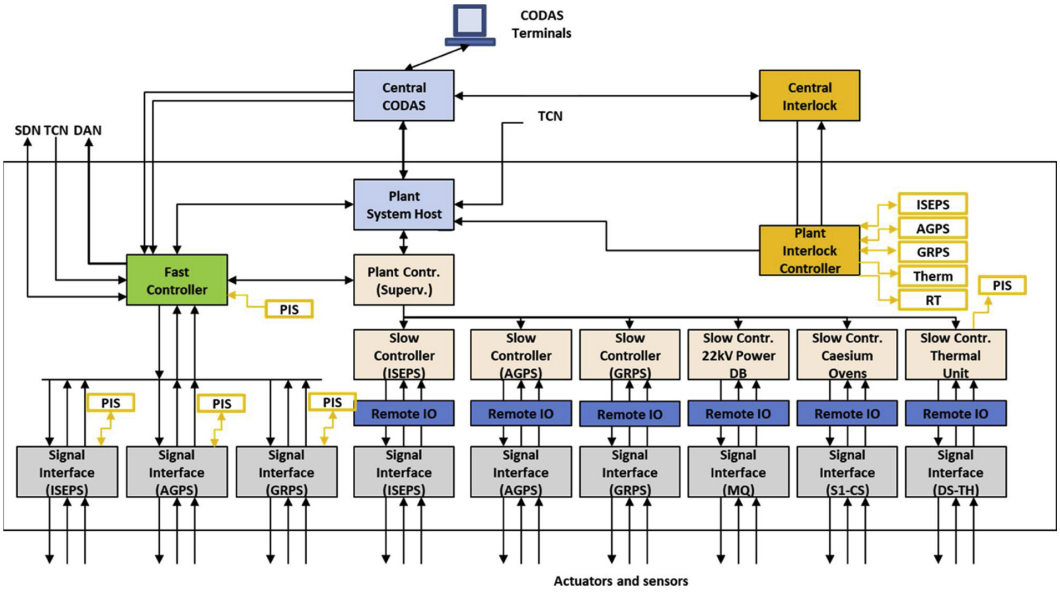


Figure 4.4: Block diagram of the I&C of the MITICA HNB plant system.

standard Linux server technology, employing Ethernet networks ranging from 1 to 10 Gb/s. This approach ensures a consistent and reliable framework for managing the HNB plant system's operations.

4.3.8 Software Design

The CODAS software will predominantly utilise established software frameworks. The ITER CODAC Core System version 6 or later will be deployed for various functions, including supervisory control, plant system hosting, human-machine interface, and alarm management.

Like the approach in SPIDER, the data management aspect will be handled using the MDSplus [49] data framework, distributed under the open-source BSD License [93]. MDSplus will be responsible for maintaining the data archive. Additionally, EPICS Process Variables will be archived using the EPICS CA, built on the MDSplus framework [83].

For the SDN and the TCN, the ITER Application Programming Interface (API) will be employed. Diagnostic data will be stored using mdsip, the communication protocol of MDSplus. Meanwhile, the HNB plant system will be equipped to eventually utilise the DAN API, which is expected to undergo further development.

MITICA will utilise the MARTe2 [48] framework for fast control operations, distributed under the EUPL (European Union Public Licence) [94]. This choice is based on MARTe2's superior real-time performance and greater determinism than the EPICS IOCs typically required by ITER.

4.4 Conclusions

The SPIDER at the ITER NBTF marks a significant stride in neutral beam injection technology since its operational commencement in 2018. Its sophisticated control system, encompassing the SPIDER CODAS, CIS, and CSS, epitomises the fusion of high-end technology with safety and reliability. The system's architecture, aligned with rigorous engineering standards, is a testament to its innovative design and functional efficacy.
Key to SPIDER's success is its robust network infrastructure and the integration of advanced software frameworks, ensuring effective communication, control, and data management. The project's proficiency in handling extensive data and its dynamic approach to real-time control highlight its capacity to manage complex operational scenarios.

At the same time, the development of the CODAS and the CIS for MITICA is currently in progress. This design process adheres to the life cycle requirements of Plant System I&C as outlined in the ITER PCDH for the HNB plant system. MITICA's operational control model will be structured around a series of operating states: global (GOS), common (COS), and specific to the plant system (PSOS). MITICA will utilise ITER's established technologies for the PON and HPNs for data communication. These state concepts' implementation and integration with the ITER network are designed to smooth the transition from MITICA's I&C systems to the HNB ones.

CHAPTER 5

CODAS UPGRADE IN RFX-MOD2

In the realm of fusion research, the RFX-mod2 project represents a significant leap forward from its predecessor, RFX-mod, particularly in control and data acquisition capabilities. This chapter delves into the comprehensive upgrade of the CODAS in RFX-mod2, addressing the challenges and advancements in managing an increased volume of electromagnetic signals and integrating cutting-edge technologies. Section 5.1 provides an overview of the new system's architecture and objectives, focusing on integrating more electromagnetic probes and transitioning to modern hardware and software solutions. The subsequent sections detail the intricacies of the CODAS architecture (Section 5.1.1), the handling of electromagnetic signals (Section 5.2), and the innovative FPGA-based framework (Section 5.2.1). The chapter also explores the refurbishment of data acquisition systems (Section 5.3), advancements in plasma control (Section 5.4), network infrastructure updates (Section 5.5), the new approaches in plant control and supervision (Section 5.6) and the upgrades to the SCADA system (Section 5.7). Each section aims to underscore how these enhancements collectively contribute to the RFX-mod2's potential for superior plasma performance and experimental capabilities.

5.1 Overview of the new RFX-mod2 CODAS

RFX-mod2 represents an advancement over RFX-mod, featuring a redesigned shell and mechanical structure to enhance the proximity between the plasma and the

shell, thereby improving plasma control [95, 96]. A key objective of RFX-mod2 is to achieve superior plasma performance, mainly through the use of an increased number of electromagnetic (EM) probes - 1500, a significant rise from the 800 used in RFX-mod. This increase in signal volume for acquisition and potential use in real-time plasma control notably affects the requirements of the Control and Data Acquisition System (CODAS), necessitating further expansions and enhancements.

Despite inheriting the overall architecture of its predecessor, the new CODAS, called SIGMA (Sistema di Gestione, Monitoraggio ed Acquisizione Dati) in RFX-mod2, needs to address the obsolescence of various hardware and software components. Fortunately, many architectural decisions made nearly two decades ago for RFX-mod, both in hardware and software, remain relevant. The new system will thus preserve these elements. The use of Linux, for instance, has become even more prevalent since the original CODAS was developed. Software frameworks like MDSplus [49, 1] and MARTe2 [48, 97], integral to data acquisition and real-time control, are still actively used and maintained in numerous fusion experiments.

In terms of hardware, the new system will continue using the CompactPCI (CPCI) technology, which forms a significant part of the data acquisition system and remains broadly utilised. However, some outdated systems still employing CAMAC (Computer Automated Measurement and Control) will be upgraded to more contemporary hardware.
Similar updates are planned for plant control and supervision. While the Siemens S7 400 PLCs, utilised in some plant systems, will be retained, the older S5 PLCs used in other subsystems will be replaced with S7 1500 ones. The new PLC software for these devices presents notable advancements while maintaining compatibility with the previous version still running on older devices.

The transition to WinCC OA (WinCC Open Architecture) [52] as the new SCADA system in the RFX-mod2 project represents a pivotal enhancement, substantially elevating the system's efficiency and streamlining its architecture. This strategic upgrade propels the RFX-mod2 project to the forefront of current technological standards, aligning it with the most advanced practices in the field. The development of a custom backend allowing direct WinCC OA-MDSplus communication not only simplifies the system's structure but also delivers marked improvements in operational efficiency.

5.1.1 CODAS Architecture Details

Figure 5.1 presents the comprehensive CODAS architecture forseen for RFX-mod2. The central data server houses the MDSplus pulse file. This file is made accessible remotely through the mdsip protocol, facilitating data access to control room computers and industrial PCs used in data acquisition.

These industrial PCs have replaced the previous CPCI CPUs and are connected to the CPCI and VME crates using bus extenders. The CPU on the EM data acquisition boards is responsible for both receiving configuration details and storing data in local RAM. These boards are also tasked with streaming the sampled data in real-time for plasma control, under the oversight of MARTe2 on the real-time server. This server employs PCIe to dispatch reference waveforms to actuators for plasma control.
WinCC OA, the new SCADA system, will perform plant supervision, controlling the new hierarchy of CODAS PLCs. Additionally, plant data acquired from slow controllers is archived in MDSplus, utilising a custom MDSplus data backend for WinCC OA. WinCC OA will access the MDSplus pulse file's configuration parameters via OPC UA or a dedicated C++ API.

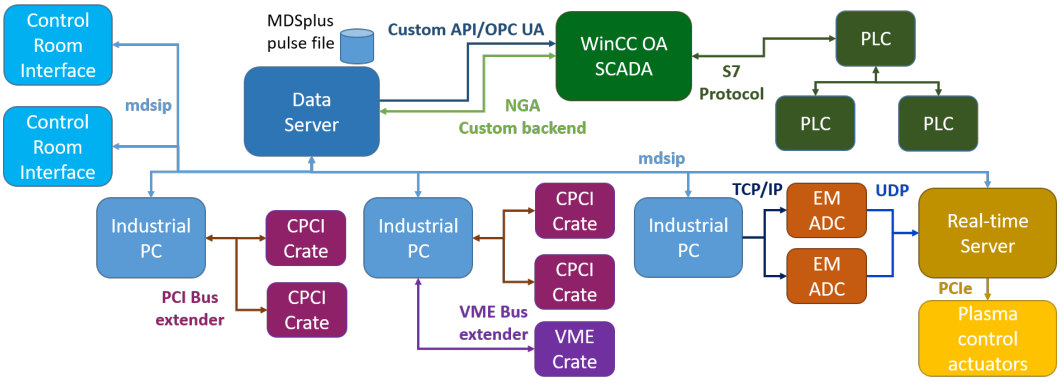


Figure 5.1: The new CODAS architecture of RFX-mod2.

5.2 Electromagnetic Signals

In fusion research, signal acquisition from electromagnetic (EM) probes is a crucial aspect of the CODAS function. This process is essential for reconstructing plasma equilibrium profiles, which are fundamental in analysing plasma performance and controlling the plasma itself. Typically, the analysis of plasma performance is conducted offline using a signal database created by CODAS, commonly referred to as

a pulse file in Fusion research terminology. In contrast, plasma control necessitates real-time data collection and computational control to comply with the stringent timing demands of the Plasma Control System (PCS), requiring a system response time of less than a millisecond.

In the RFX-mod system, two separate systems were employed for these tasks: a CPCI-based solution for non-real-time data collection and a VME (Versa Module Eurocard) for real-time data collection. This approach led to expensive hardware duplication, making managing data acquisition and hardware maintenance more complex. Additionally, most signals from EM probes in RFX-mod need to be time-integrated to ascertain the measured magnetic field, a task performed by specialised hardware.

Furthermore, Plasma Control often requires the time derivative of these input signals, for instance, for PID control. These objectives can be achieved in two ways: (i) by digitally differentiating the time-integrated original signal, or (ii) by acquiring both the integrated and original EM signals separately. In the past, the latter approach was generally preferred for control purposes as it avoids potential errors from digital differentiation. However, this method necessitates the duplication of acquisition channels.

5.2.1 FPGA-based Architecture

The innovative FPGA (Field-Programmable Gate Array)-based framework being developed for RFX-mod2 offers a comprehensive solution:

1. The ADC board equipped with the FPGA will serve two purposes: firstly, for high-speed data collection (reaching up to 1 MHz) using DMA for transferring data to local memory, and secondly, for low-speed (10 kHz) data sampling and streaming, essential for plasma control.

2. This same FPGA will handle digital integration, supplying real-time integrated signals and eliminating the need for a separate analog integration front end.

For extended experiments, continuous streaming is necessary for data collection. However, given the expected pulse duration in RFX-mod (up to 1 s), the local ADC board memory can be utilised in a Transient Recorder mode. In this mode, signals are stored in local memory during the pulse and then read post-pulse.
Furthermore, due to the brief duration of plasma discharges, precise integration can be achieved by correcting offsets before discharge. This process bypasses the need

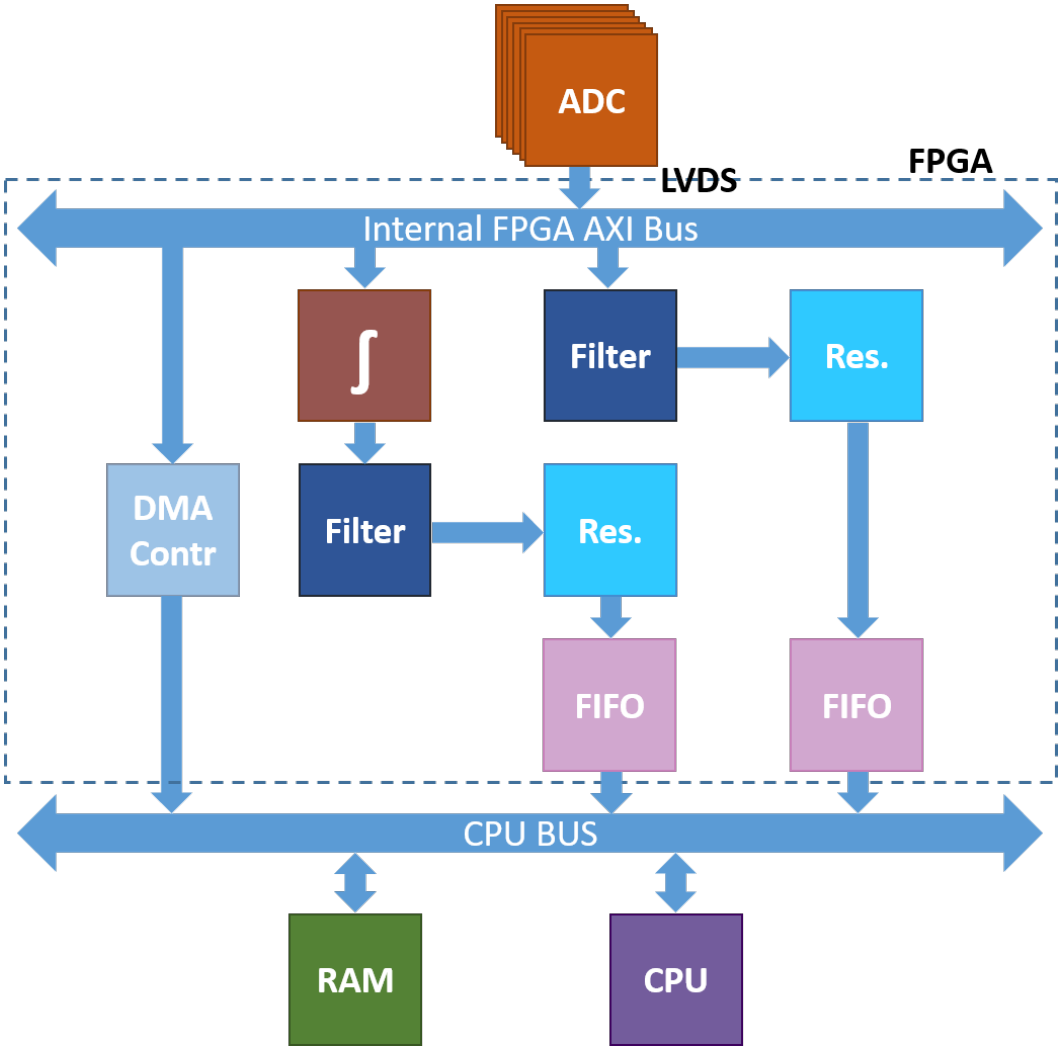


Figure 5.2: The FPGA architecture. DMA (Direct Memory Access) Controllers and FIFO interfaces are implemented by XILINX IPs AXI-DMA and AXI4-FIFO, respectively.

for more complex techniques like chopping, as used in ITER.

The presence of FPGA for real-time calculations enables the numeric integration of EM signals for real-time control, thus bypassing the need for analog integration and the duplication of ADC channels. Each channel is designed to rapidly capture a direct signal (typically a magnetic field derivative) and stream a sub-sampled, integrated signal for plasma control.
While the data acquisition of integrated signals at high frequency is feasible via the FPGA, it is unnecessary. Indeed, these signals can be computed offline, which helps avoid unnecessary duplication of DMA channels and memory usage.

5.2.1.1 Architecture Details

An initial assessment of numerical integration's viability revealed that the performance of the ADC stage, particularly its noise spectrum, was crucial. The noise spectrum's 1/f characteristic could lead to unacceptable integration drift, as cited in [98]. This issue rendered unsuitable the initially considered ADC stage, previously utilised in other experiments [99].

A new ADC stage is being developed to address this, featuring the ADS8900B chip, which boasts an improved SNR. Another key feature of this ADC board is the galvanic isolation of input signals after digitisation. This isolation is vital in EM probe acquisition due to strong electromagnetic fields' potential induction of over-voltages exceeding 1 kV. These insulated ADC stages are connected to a SOM Board PicoZed 7020 [100]. Each board will handle 12 ADCs, responsible for temporary storage, filtering, sub-sampling, integration, and streaming.

Figure 5.2 illustrates the hardware components incorporated into the FPGA. It shows how 20-bit channels, acquired at 1 MHz from the 12 ADCs, are fed into a unified internal AXI (Advanced eXtensible Interface) bus and distributed to other components. The first data pathway directs data to RAM through a DMA controller, while the second internal AXI bus takes in integrated data. This data, after undergoing low pass CIC filtering and being downsampled to 10 kHz, is forwarded to the CPU via a FIFO interface.

The Linux IP stack on the ARM processor embedded in the Zynq FPGA provides the software interface between the board and the network. The MARTe2 framework, installed in the processor [46, 101], will oversee all data flow, encompassing offline data acquisition and real-time streaming. A prototype of this board has been developed and is currently undergoing testing.

5.3 CODAS Refurbishment

Aside from EM signal acquisition, RFX-mod2's data acquisition system will employ a distributed architecture similar to RFX-mod, utilising Linux machines and CPCI. The embedded Linux CPUs within the CPCI units, functional since 2004, are now outdated and no longer viable. Replacing these CPUs would incur significant costs, leading to the choice of a more economical alternative. The planned strategy involves using inexpensive industrial PCs linked to the CPCI crates via bus extenders,

with each PC capable of supporting up to two CPCI crates.

Software-wise, the adaptability of the MDSplus data acquisition system simplifies the transition from embedded systems to bus extenders. This shift only necessitates reassigning tasks among processes without requiring user or system software modifications. The MDSplus experimental database, which includes data acquisition task configurations, retains this information.

The MDSplus Dispatcher tool [102] facilitates the activation and coordination of distributed software elements for data acquisition, relying on the existing data within the experiment database. Consequently, alterations in the data acquisition task structure, such as relocating tasks between computers, merely require the experiment database content updates. These modifications can be performed using MDSplus's graphical tools, jTraverser and jScope, designed for navigating the experiment database.

This methodology is also applied in replacing certain VME-based diagnostic control systems that utilise embedded CPUs operating on VxWorks. It involves introducing VME bus extenders and adapting the VxWorks-specific code for Linux compatibility. Other older systems built on CAMAC will need custom solutions using RedPitaya devices with custom FPGA configurations.

5.4 Plasma Control

At RFX-mod, the MARTe framework was utilised for plasma control, operating on a multicore server for real-time tasks and integrated with MDSplus for configuration retrieval and signal storage [46]. For RFX-mod2, the plasma control system will continue to employ MARTe but with several notable changes. It will operate on a multicore server running real-time MRG Linux.

The upgrade from MARTe to MARTe2 is a key difference. MARTe2 represents an advanced version of the original MARTe framework, developed following stringent software quality guidelines and incorporating new, advanced concepts for system abstraction [101]. Like its predecessor, MARTe2 operates based on a system configuration description detailing component interactions. This configuration, typically a file parsed at system start-up, can be complex due to the involvement of multiple components. Consequently, even minor modifications in the system configuration

previously necessitated significant efforts in updating, often leading to errors and difficult-to-diagnose crashes during operation.

MARTe2 eases this issue slightly by implementing a more rigorous set of consistency checks at start-up. Nevertheless, the complexity of the configuration description remains a challenge, typically manageable only by skilled developers. Addressing this, RFX-mod2 has adopted a new approach by improving the integration of MARTe2 with MDSplus. This enhancement utilises MDSplus for a high-level system description, which is more straightforward and user-friendly than the low-level MARTe2 configuration file. Consequently, the MARTe2 configuration is generated automatically from this high-level description in the pulse file through a series of device objects. This method significantly reduces the risk of configuration errors. The trade-off for utilising the high-level description from MDSplus devices is reduced flexibility in MARTe2 configurations. However, this high-level model abstraction satisfactorily fulfils all practical requirements for plasma control.

Regarding interfacing, the plasma control system will connect with data acquisition components via a 10GB insulated ethernet segment using UDP. RFX-mod was among the pioneers in adopting UDP for plasma control component communication [103], a protocol also chosen for the ITER Synchronous Data Network (SDN) for plasma control. The direct Peripheral Component Interconnect Express (PCIe) link between the multicore server and DAC devices for reference waveform generation, a feature introduced in RFX-mod in 2012, will be continued in RFX-mod2.

5.5 Network

The RFX-mod2 network infrastructure, continuing to rely on Ethernet technology as the de facto standard, has seen significant updates, including introducing new communication technologies and replacing obsolete components. While maintaining its star architecture, the network now includes a modular chassis Ethernet switch/router and Ethernet edge switches distributed across all experimental areas. At the same time, introducing IEEE 802.1Q VLAN (Virtual Local Area Network)s allows optimising and separating traffic flows on a functional basis, reducing the number of Ethernet edge switches.

Significant topological changes have expanded the network to new experimental areas, enabling data acquisition from proximity sensors and fast diagnostic systems, particularly in areas with diagnostic cameras and EM probes requiring real-time

acquisition. The upgrade to redundant 10 Gbit/s fibre optic technologies for backbone connections enhances bandwidth for massive data transfers and reduces latency, crucial for real-time data acquisition. This upgrade includes using 50/125 µm multimode optical fibres in class OM3 or OM4, supported by 10GBASE-SR IEEE 802.3ae Ethernet technology, and cat. 7 type S/FTP network cables for 10GBaseT IEEE 802.3an-2006 technology. Peripheral uplinks have been increased to 1 Gbit/s, primarily using optical fibre, to mitigate electromagnetic disturbances and looping risks.

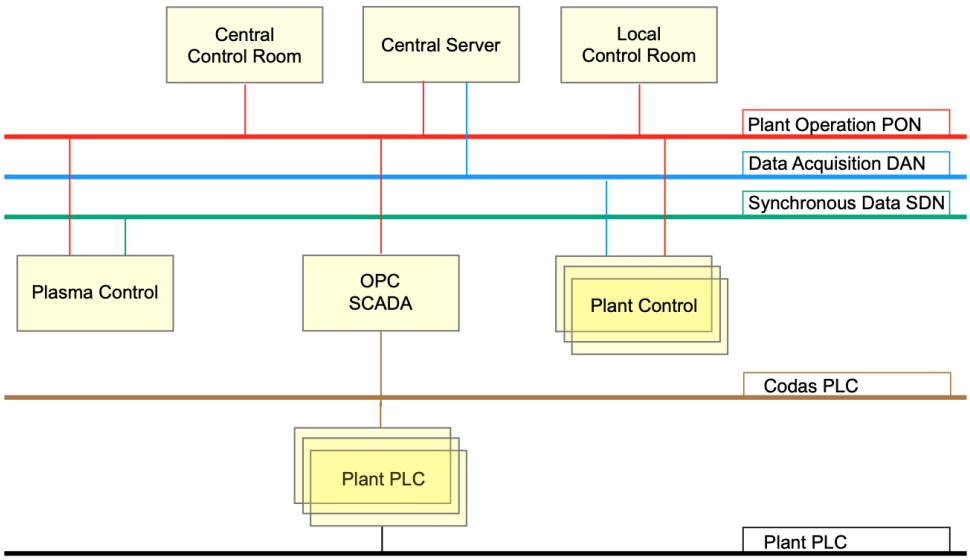


Figure 5.3: The RFX-mod2 network architecture.

As mentioned, integrating IEEE 802.1Q VLAN technology enables the functional separation of different traffic flows, such as plant operation, PLC plant, and real-time SDN traffic. This network segregation is depicted in Figure 5.3, which shows the VLANs defined for RFX-mod2 CODAS: the Plant Operation Network (PON) VLAN, connecting control and acquisition systems, including local and central control room consoles; the Data Archiving Network (DAN) VLAN for high-bandwidth systems; and the SDN VLAN for real-time plasma control communication. Additionally, the PLC VLAN facilitates communication between SCADA systems and PLC devices.

Physical and logical segmentation, aligned with modern concepts of system functional safety, ensures regulated and authorised access for remote assistance and management activities. The Plant PLC network, linking plant and CODAS PLCs, is an

independent LAN without routing mechanisms for enhanced safety.

Network Monitoring Systems employing Simple Network Management Protocol (SNMP) have been introduced for monitoring the status of individual uplinks and switch ports, allowing for prompt response to anomalies. These systems and replacing obsolete Ethernet switches support implementing a comprehensive fault monitoring system and load analysis on individual sections. All these network management activities utilise a separate management VLAN.

Furthermore, the network's resilience and load distribution are bolstered by introducing bandwidth aggregation technology for each uplink. The Link Aggregation Control Protocol (LACP), defined by the IEEE 802.1AX standard, is envisaged to provide load distribution and redundancy, significantly enhancing the network's overall performance and reliability.

5.6 Plant Control and Supervision

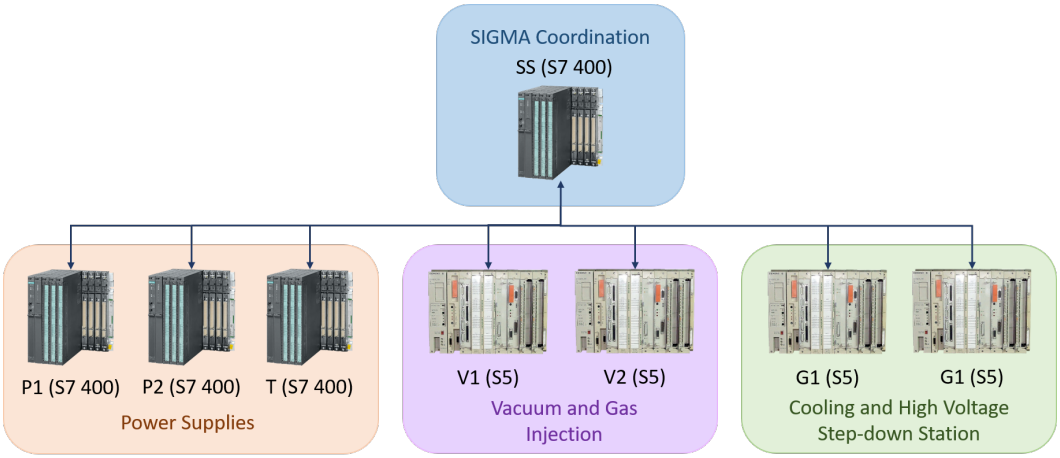
The Plant Control for RFX-mod2 will continue to use the hierarchical distributed structure established in RFX-mod. The strategy is to keep existing hardware and software at the plant and supervisory levels wherever possible, aiming to minimise costs and shorten development time. Nevertheless, updates are required due to technological advancements and new operational needs.

Several plant systems, unchanged since the earlier RFX experiment, present age-related maintenance issues and are, therefore, due for an overhaul. Among these critical systems, which can no longer be maintained with their current setup, are the Vacuum and Gas Injection System, the Cooling System, and the High Voltage Step-Down Substation. This revamping includes their plant control systems, often based on outdated Siemens SIMATIC S5 PLCs technology. They will undergo a complete hardware and software redevelopment to meet new experimental requirements and solve the oldness issues. Concerning plant control, all the legacy S5 PLCs will be replaced by the more contemporary SIMATIC S7-1500 ones.
Conversely, other plant systems, such as the Power Supply ones for poloidal and toroidal coils, were already upgraded for RFX-mod. Despite not being at the state of the art, these systems are still functional, so upgrading them would be resource-intensive.
Concerning Plant Control, these systems operate on SIMATIC Siemens S7 400 PLCs, which will remain in function.

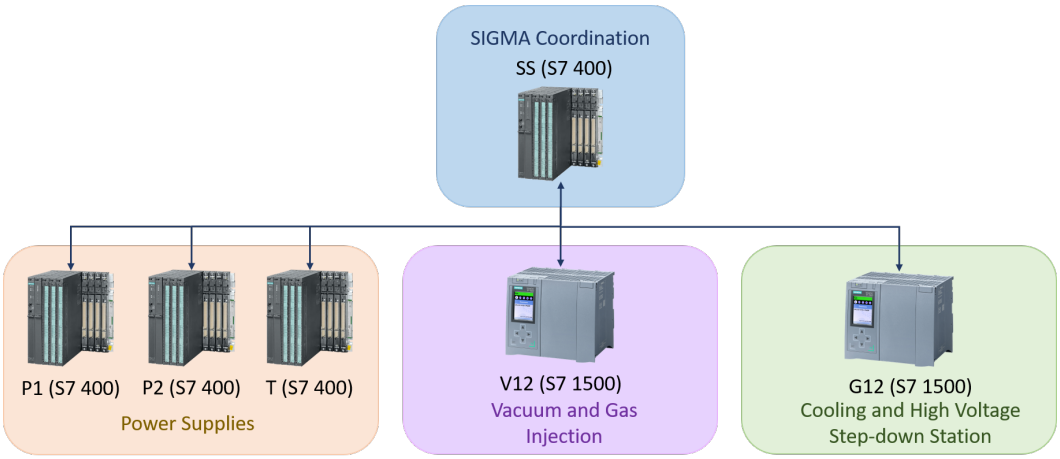
In terms of Plant Control coordination, similar principles apply. While specific components will be replaced, the overarching SIGMA architecture, characterised by a hierarchical state machine synchronising all subsystems, will remain unchanged. In terms of hardware, all the old S5 PLCs will be substituted with S7 1500 models. When the system design permits, a single S7 1500 PLC will take over the functions of multiple S5 units, leading to cost savings and a more streamlined architecture.
As in the case of Plant Control, SIGMA S7 400 PLCs will remain operational when possible. However, the top-level SIGMA coordinating PLC, called SS, constitutes an exception. While it is an S7 400 PLC, its hardware must be replaced with a more recent S7 400 series one. The reason leading to this intervention will be made clear in Section 5.6.1.
The changes in the hierarchical structure of SIGMA PLCs are illustrated in Figure 5.4.

The Vacuum and Gas Injection System and the High Voltage Step-Down Substation and Cooling systems exemplify the architectural simplification regarding the SIGMA PLC hierarchy.
Consider the Vacuum and Gas Injection System first; until RFX-mod, its plant control consisted of several S5 PLCs. VI was responsible for controlling the Gas Injection subsystem, VV was tasked with controlling the Vacuum one, and VD was in charge of controlling gates to the vacuum chamber. All these obsolete devices will be replaced with S7 1500 ones. A single S7 PLC, named VV/VI, will perform the tasks of its two predecessors, VV and VI; VD will be replaced by a separate S7 1500 PLC, and a new S7 1500, aptly called GDC, will be in charge of controlling the Glow Discharge Cleaning subsystem. The coordination of these plant PLCs consisted of two SIGMA S5 PLCs, namely V1 and V2. These two PLCs reported directly to the supervisor SS PLC. For RFX-mod2, it is foreseen to replace V1 and V2 with a single S7 1500 PLC called V12, which will perform all of the tasks of his predecessors.
The new control architecture for the Vacuum and Gas Injection System of RFX-mod2 is illustrated in Figure 5.5.

The approach used for plant coordination of the High Voltage Step-Down Substation and Cooling plant systems is similar. In RFX-mod, the plant coordination hierarchy consisted of two S5 PLCs, namely G1 and G2, reporting to SS. While G1 supervised the plant PLC of the High Voltage Step-Down Substation, called GS, G2 was responsible for coordinating the S5 plant PLC of the Cooling System, called GR. Going more into detail, G1 and GS were implemented by a single S5 PLC,



(a) The RFX-mod SIGMA PLC hierarchy.



(b) The RFX-mod2 SIGMA PLC hierarchy.

Figure 5.4: Foreseen changes to the SIGMA PLC hierarchy from RFX-mod to RFX-mod2.

called G1/GS. Since the High Voltage Step-Down Substation and the Cooling plant system will undergo a complete overhaul, and the S5 devices are now legacy, their plant coordination control structure for RFX-mod2 will be the following.
The Cooling System and the High Voltage Step-Down Substation will have dedicated new plant control S7 1500 PLCs, consistently named GR and GS, respectively. As for the Vacuum and Gas Injection System, the SIGMA coordinator PLCs G1 and G2 will be replaced by a single modern S7 1500 one named G12.

Substituting these devices while simplifying the overall architecture introduces several compatibility challenges, thoroughly detailed in Section 5.6.1.

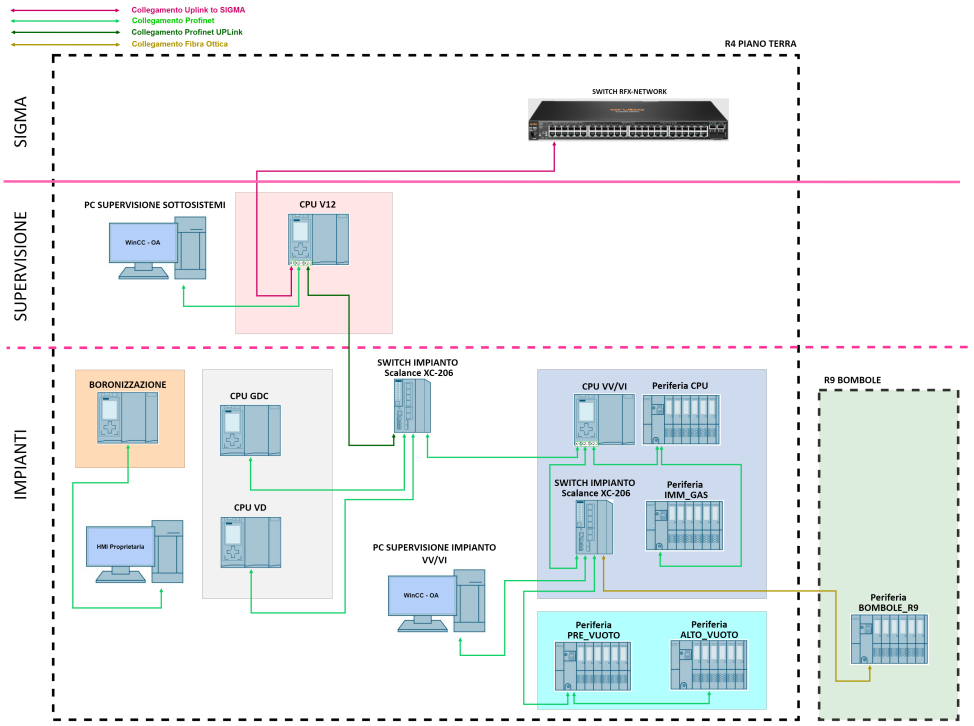


Figure 5.5: The RFX-mod2 detailed Plant Control architecture of the Vacuum and Gas Injection system.

5.6.1 PLC Software

While inevitable, the decision to replace several SIGMA PLCs presents some critical aspects that must be considered.
For instance, starting the development of a new version of the software running on these PLCs, called *Scheduler*, was necessary. This updated Scheduler version is completely written in SCL, avoiding the legacy STL used to write all of the S5 and S7 400 Scheduler code.
The new Scheduler version includes a hierarchical state machine, enhanced alarm management, and improved communication features while retaining several of the characteristics of its previous version to avoid compatibility issues with preexisting devices.

One notable example is the message exchange mechanism used by SIGMA PLCs to communicate with each other: in the previous Scheduler versions, information exchange was implemented by sending and receiving custom messages, relying on proprietary Siemens S7 Protocol routines (AGLSEND, AGLRECV). Each message is structured as a series of 16 consecutive bytes [104], containing information on

the message source, destination, class, type, and data to be sent, as illustrated in Table 5.1.

SOURCE		Word 0
DESTINATION		Word 1
CLASS	TYPE	Word 2
DATA 0	DATA 1	Word 3
DATA 2	DATA 3	Word 4
DATA 4	DATA 5	Word 5
DATA 6	DATA 7	Word 6
DATA 8	DATA 9	Word 7
DATA 10	DATA 11	Word 8

Table 5.1: The SIGMA Scheduler message structure.

The mechanism to write, send, receive and read messages is rather complex. Though thoroughly documented in [104], its software implementation is not always straightforward and often not adequately separated from other software logic.
Maintaining this message system in the new Scheduler version is paramount to exchanging information with the older SIGMA PLCs. The only other alternative would imply significantly modifying the previous Scheduler software running on these PLCs. The adopted solution consisted of replicating its logic while completely rewriting its routine sequence, making it possible to save resources and avoid introducing errors in a widely developed and often undocumented code.
However, new issues arose when replicating the message exchange mechanism on S7 1500 PLCs. As mentioned, the previous Scheduler software relied on proprietary Siemens S7 Protocol routines (AGLSEND, AGLRECV) for communication between PLCs. Unfortunately, these routines are unsupported in the S7 1500 PLCs. Instead, the TCP-IP-based Open User Communication (OUC) has become the standard for non-real-time communication with SIMATIC S7 CPUs, which translates to using the SCL TCONN, TDISCON, TSEND and TRECV communication routines.

Therefore, the older Scheduler software requires minor adjustments to implement OUC communication with the new S7 1500 PLCs. Note that these modifications only regard the communication routines sending the data: no change is required to the message framework.
Unfortunately, the current CPU (model 416-2 DP 6ES7 416-2XK01-0AB0) and the CP (model 443-1GX11-0XE0) of the SS Supervisor PLC proved incompatible with

OUC: their current framework does not support OUC, and upgrading their framework to the required version is unsupported. Consequently, it is necessary to substitute the SS PLC hardware with a more recent S7 400 version.
It is important to understand the reason for replacing it with an S7 400 series instead of a more modern S7 1500 one: introducing new S7 1500 hardware for the SS PLC would imply running the new Scheduler software. Therefore, this strategy would only shift the OUC compatibility problem to the communication between the SS PLC and all other S7 400 PLCs, worsening it by far instead of solving it.

While a new Siemens S7 400 series CP has been purchased for this purpose (model 443-1 6GK7443-1EX30-0XE0), the order of the new CPU (model 416-3 PN/DP 6ES7416-3ES07-0AB0) has been pending since December 2022.
In the meantime, OUC has been tested between an S7 1500 PLC and an S7 400 one by using the new S7 400 CP together with an S7 400 CPU (model 416-3 PN/DP 6ES7 416-3ER05-0AB0) temporarily borrowed from another experiment. However, the new S7 400 series CPU is fundamental for further software development and testing.

In addition to the message framework, more previous Scheduler features had to be retained to facilitate software compatibility.
Among these is the MOP, which defines what the experiment is doing at a high level. Examples of RFX MOP are RFP impulse, Tokamak impulse, GDC (Glow Discharge Cleaning), PDC (Pulse Discharge Cleaning), Baking, Pumping and Variation of Temperature.
Setting the experimental MOP has to be done before starting the experimental activity so that all SIGMA devices must be informed about the selected MOP and synchronise accordingly.

Similarly, the RFX-mod state machine, shown in Figure 5.6, was maintained and implemented in the new Scheduler, meaning the Finite State Machine (FSM) state request/transition mechanism was also retained. Each device has an internal copy of the RFX FSM, which must always be synchronised with the FSM of its partner devices. State transition requests are propagated hierarchically from the SS PLC to the other PLCs, which must reply accordingly. Each FSM state has three internal sub-states, namely *PREP*, *ENTER* and *NEXTEN* and must inform its parent device when each of these is reached.

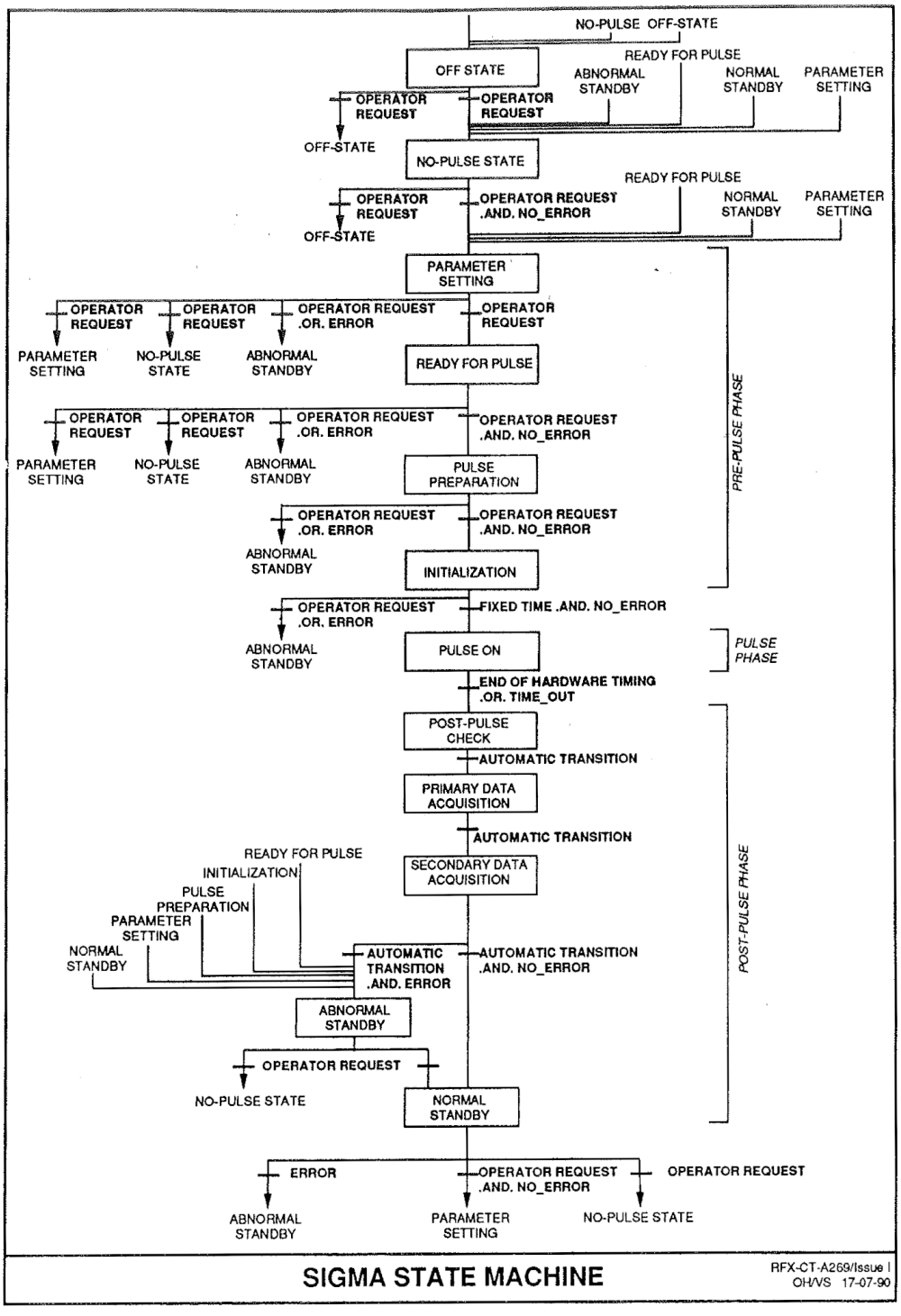


Figure 5.6: The RFX-mod SIGMA State Machine.

Error handling has also been aligned. Two main error alarms can occur and are propagated upwards hierarchically: the *ABORT*, which freezes any experimen-

tal operation and forces each device to reach a safe FSM state called *ABNORMAL STANDBY*, and the *PULSE INHIBIT (PI)*, which prevents the FSM from advancing further.
Two other features which have been retained are the check-in and watchdog protocols, which are used to check devices' presence and responsiveness: the first is used at start-up, and the second periodically to verify communication status.

Despite bearing all these similarities, the new Scheduler version also presents fundamental changes compared to its previous version. As previously mentioned, two of the most evident ones are choosing SCL over STL and basing communication on OUC.
Another fundamental improvement is code reusability and modularity, as seen in several code design choices.
The new Scheduler version is designed so that a single physical PLC can run multiple devices. To achieve this, it is simply necessary to (i) instantiate a copy of a specifically developed Function Block (FB) (which can be seen as a "macro-function block" and will be referred to as *Device-FB* from now on) and run it in the *Main* Organization Block (OB) (OB1), (ii) set some device parameters in the associated Data Block (DB).
An example of the TIA Portal Scheduler software structure for a SIGMA PLC is shown in Figure 5.7.

The Device-FB, which completely defines the device's parameters, its communication partners and its FSM, is based on a modular structure which heavily relies on nested UDTs and FB. Thus, its concept closely resembles the one of a *class* in object-oriented programming: it contains not only data fields but also all the data-based routines (which are FBs in PLC logic).
The main parameters to be configured are the device ID and its network parameters; the same should be done for its communication partners, specifying their hierarchical level (i.e. if the partner is the parent or a child in relation to the device).
Even if it is not compulsory to instantiate a device (default values are always set), it is also possible to easily customise the FSM parameters, such as state names, timeouts, etc.

The FSM is instantiated as a field of the Device-FB. Its structure is based on the reception of a state transition and on advancing the three sub-states mechanism, thus making it independent of the specific state as much as possible.
The number of communication partners is also easily scalable (up to a maximum of

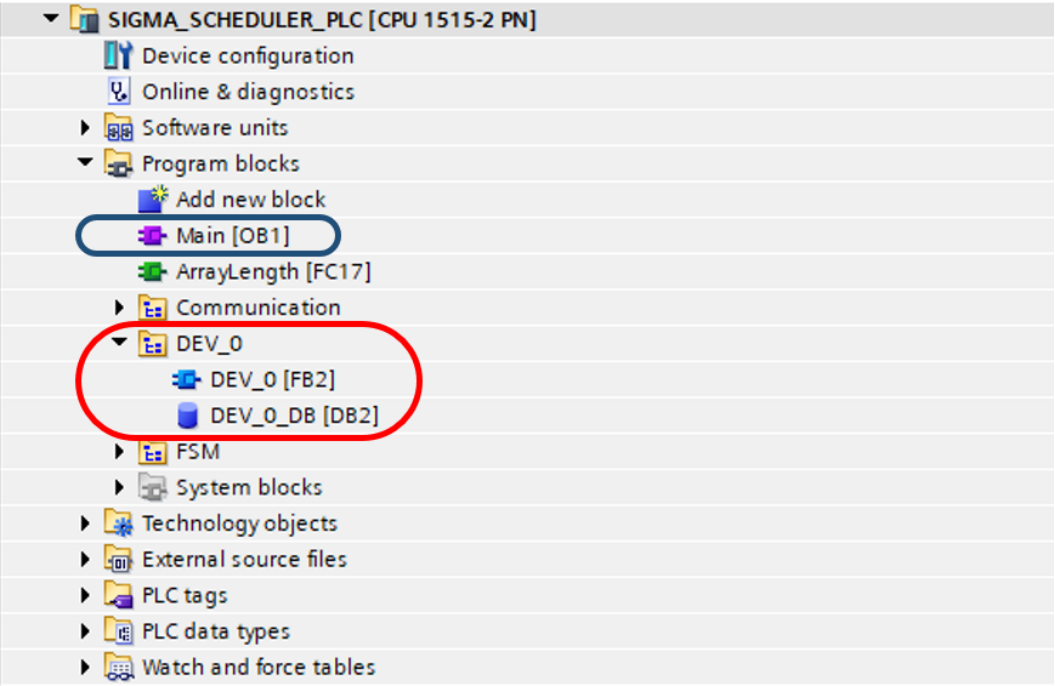


Figure 5.7: The SIGMA Scheduler Scheduler software structure. The only necessary steps are configuring parameters in the Device-FB (circled in red) related to the instantiated SIGMA device and calling it in the Main OB1 (circled in blue).

ten per device): it is sufficient to set the parameters only for the desired partners in the specific area of the Device-FB. All the OUC communication FBs are automatically instantiated based on the provided partners' IP addresses. Note that this communication paradigm is highly flexible since it avoids the creation and manual configuration of dedicated proprietary "System Blocks", allowing the definition of partners without explicitly declaring the network architecture in TIA Portal.
The unexploded structure of the Device-FB is illustrated in Figure 5.8.

In summary, the code structure of the new Scheduler, consisting of nested FBs and FCs, achieves notable simplicity and robustness thanks to its data-oriented and code reusability approach. It is mainly based on the Main OB, where all the necessary Device-FBs are called. Each Device-FB handles two main tasks: communication with partner devices (parent and children) and running its FSM. The communication function is based on standard OUC FBs, nested as sub-FBs in the Device-FB structure. Moreover, all secondary communication routines are based on FCs, allowing them to be shared among the devices running on the same hardware and enhancing code reusability. The FSM FB is nested into the Device-FB and based on the three state sub-steps (*PREP*, *ENTER* and *ENABLE*) to make the code not

DEV_0

	Name	Data type	Default value	Retain
1	Input			
2	<Add new>			
3	Output			
4	<Add new>			
5	InOut			
6	<Add new>			
7	Static			
8	Dev_ConfigOK	Bool	false	Non-retain
9	FSM	"FSM"		
10	PartnerDataArr	Array[0..10] of "Partner_Data"		Set in IDB
11	NumOfEnabledPartners	UDInt	0	Non-retain
12	MOP_Table	"MOP_Table"		Non-retain
13	PulseData	"PulseData"		Non-retain
14	PartnerCommunication	"PartnerCommunication"		
15	HMI_Communication	"HMI_Communication"		
16	Temp			
17	i	Int		
18	retDateTime	Int		
19	Constant			
20	<Add new>			

Figure 5.8: Unexploded structure of the Scheduler Device-FB.

dependent on the current state. As in the case of the communication routines, all secondary functions are implemented through FCs to maximise code reusability. A detailed explanation of the new Scheduler software is provided in Appendix A to avoid overly burdening the discussion in this chapter.

All these considerations exemplify how deploying and configuring a SIGMA PLC running the new Scheduler software becomes straightforward, significantly reducing its complexity and the risk of error introduction.
Further development and testing in real-case scenarios are, however, still needed. One promising feature considered for development consists of a tool to automatically generate a device's configuration file (for example, using JSON syntax). This file could then be used to automatically generate the related Device-FB in TIA Portal, thus sparing the necessity to manually configure the parameters at DB level.

5.7 SCADA System

The SCADA system used in RFX-mod is FactoryLink. FactoryLink was a real-time industrial automation and process control software application developed by

USDATA Corporation. It was widely used for SCADA applications. The key features of FactoryLink included its ability to connect with a wide variety of industrial devices and systems, real-time data processing, graphical user interface (GUI) for monitoring and control, and robust data logging and reporting capabilities.
Siemens acquired FactoryLink and other related products from USDATA Corporation around 2001. Subsequently, Siemens integrated many of FactoryLink's features into its industrial software offerings. FactoryLink is a legacy product, and Siemens' newer SCADA solutions have largely replaced it in the marketplace. These modern systems offer more advanced features, such as better integration with other Siemens products, and are more suited to the evolving needs of industrial automation and digitalisation.

This rationale is behind replacing FactoryLink with the top-tier SCADA from Siemens: WinCC OA. This upgrade will also facilitate native S7 communication between the SCADA and the plant system PLCs, streamlining the setting and reading of process variables. In RFX-mod, FactoryLink ran on a dedicated Windows server called NTSCADA. Similarly, a dedicated Windows OPC server, NTOPC, needed to act as an intermediary between FactoryLink and Siemens PLCs. As illustrated in Figure 5.9, WinCC OA allows direct access to Siemens PLCs DBs: the only needed action consists of configuring a built-in driver at the SCADA level.
Therefore, introducing WinCC OA removes the need for this second server to access PLC tags.

However, the NTOPC server performed another function in RFX-mod: it enabled communication towards the MDSplus pulse file on the Linux operation server, ROServer. This connection is essential to store data acquired from slow controls to MDSplus and to set SCADA and PLC parameters from MDSplus devices. As a matter of fact, towards the end of RFX-mod experimental sessions, when FactoryLink was already being phased out, several of its GUIs (such as those used to set the MOP) had already been transitioned to MDSplus devices.
The introduction of WinCC OA greatly facilitates communication with MDSplus, which is still essential for data acquisition, configuration, and control.

Concerning the necessity to set WinCC OA data points (the equivalent of EPICS PVs) from MDSplus devices, current efforts are focused on determining the most efficient method. One possibility is to achieve this through an OPC UA (Open Platform Communications Unified Architecture) server integrated into WinCC OA, thus replacing the dedicated NTOPC server. This strategy requires minor modification

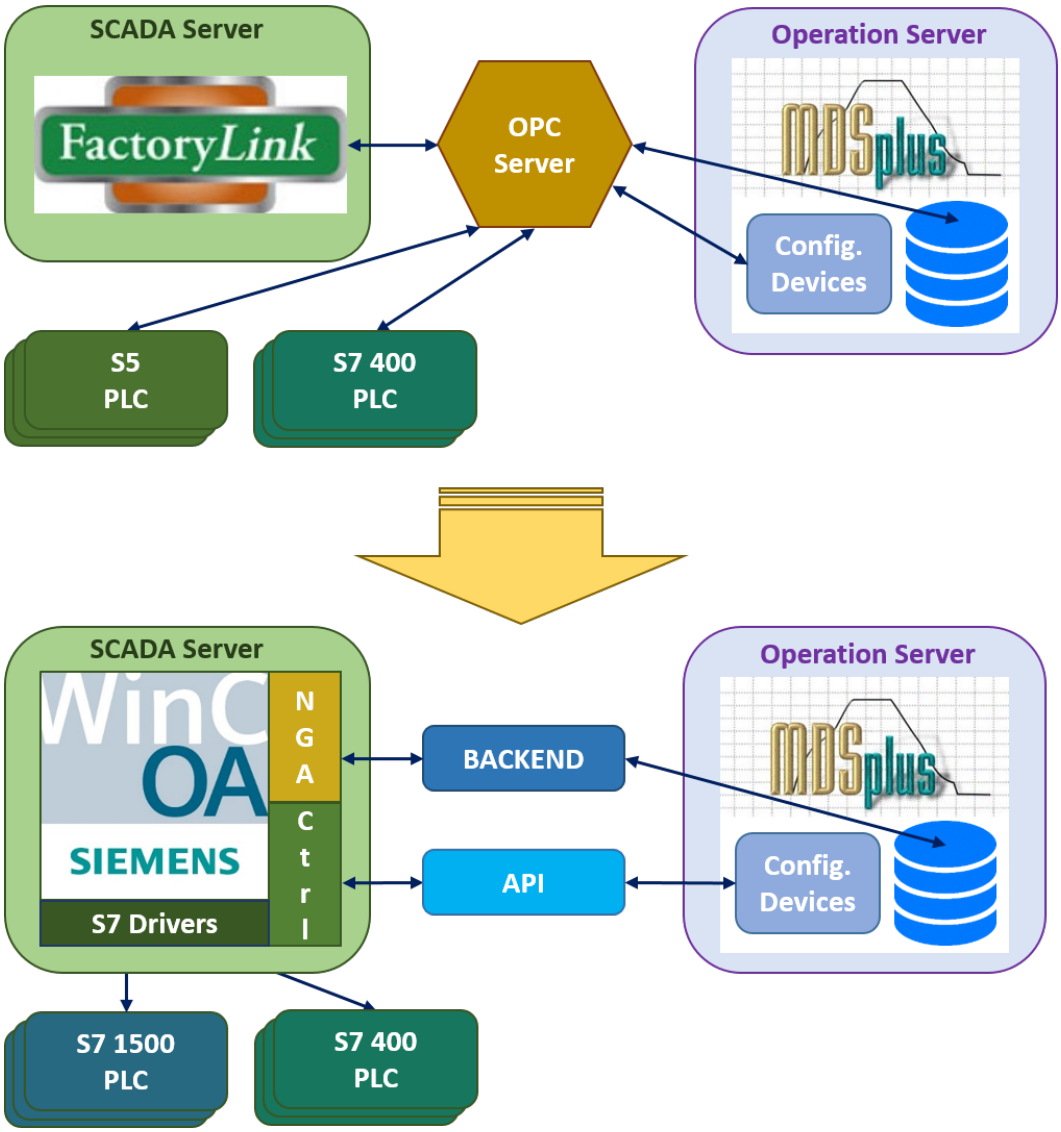


Figure 5.9: The evolution of the Plant Supervision architecture from RFX-mod to RFX-mod2.

on the MDSplus side for interfacing with OPC UA.

The second possibility involves developing a custom C++ API. This API would be accessed by WinCC OA CTRL scripts, which could be executed whenever needed. If the API approach proves more effective, it will provide direct access to MDSplus pulse files for configuration purposes.

Regardless of the chosen solution, the need for the NTOPC server would cease to exist, thus removing an intermediate layer and simplifying the overall SCADA architecture.

Before being able to proceed, it is, however, paramount to address the storage of data points to the MDSplus pulse file. To this purpose, a new C++ database backend is being developed for WinCC OA to record the time evolution of process variables monitored by the SCADA system in MDSplus.
The backend is a layer of software which allows MDSplus to be seen as an archiving database by WinCC OA. The NextGenArchiver (NGA) feature of WinCC OA manages the first half of the communication, sharing data between the SCADA system and the backend. The backend, in turn, has been developed to have native access to the MDSplus pulse files. The backend architecture is described in more detail in Section 5.7.1.

In addition to developing the backend and the API, significant work is still underway to re-map all the necessary data points from NTSCADA/NTOPC to WinCC OA and to develop state-of-the-art GUIs to monitor and control them.
Defining a strategy that allows this process to be automated at least partially is critical to saving resources. The native WinCC OA ASCII manager, which allows massive data point import and export from CSV files, constitutes a valid starting point.

5.7.1 The WinCC OA - MDSplus Backend Architecture

As mentioned in Section 5.7, significant effort has been put into developing the custom backend, which allows direct communication between WinCC OA and MDSplus.
The development did not start from scratch: ETM (the Siemens subsidiary company which originally developed WinCC OA) provided an initial standard version of the C++ software, already able to communicate with the NGA, receiving its request and sending back the related responses.
The necessary steps to interface it with WinCC OA consist of:(i) compiling the code, (ii) copying the compiled file to a specific WinCC OA installation directory, (iii) importing in WinCC OA a set of provided data points through the integrated ASCII Manager, (iv) activating a licence supporting custom database backends, (v) activating the custom backend in WinCC OA's database settings panel, (vi) creating a first example of data point archiving group associated with the custom backend and (vii) setting the archive group's parameters from the database settings panel.
In this way, the NGA can see the stub version of the backend as a custom data point archiving database.
However, at this point, there still was no interfacing to MDSplus pulse files, so the

backend acted as a sort of empty shell, sending back empty replies to the NGA's requests.

Before addressing the interfacing to MDSplus, spending a few words on WinCC OA's data point archive group concept is necessary. WinCC OA mainly works on asynchronous data point value change, calling them *Events*. Therefore, it handles data point archiving by sorting them into archiving groups called *Event Groups*. Usually, each Event Group is associated with an archiving database. However, it is also possible to associate one group with more than one database or several separate groups with the same one. This concept has been taken advantage of by imagining that different data point groups could be conveniently archived in separate MDSplus trees, depending on their origin and scope. For example, data points related to the Power Supplies System could belong to a specific MDSplus tree, while others related to the Vacuum and Gas Injection System to their own.
Each of these MDSplus trees used to archive data point samples il called a *TrendTree*. Therefore, while WinCC OA is oblivious to MDSplus's internal tree structure, simply seeing it as a database, MDSplus architecture possibly consists of several TrendTrees to archive data points.
While presenting advantages, this strategy introduces the need to identify the Event Group related to each data point sample to archive it in the correct TrendTree. The top-level tree *MasterTree* is introduced to achieve this. This tree is not used to archive data points but to map each Event Group to its related TrendTree. This architecture is illustrated in Figure 5.10.

The structure of the MasterTree is quite simple, as shown in Figure 5.11. It contains a node called "TREE_MAP", which is used to map all the known TrendTrees. Next comes a structure node called "Trees", where the model trees of all available TrendTrees are linked as sub-trees for convenience. Only the model shot of the MDSplus MasterTree is used.

Figure 5.12 illustrates the structure of a TrendTree. The model shot and subsequent shots are used for a TrendTree, but their scope differs. The model shot is needed to organise all the data point information stored in this tree and save the information on how it should be stored. Successive shots are instead used to archive data point samples, each mirroring the model tree's structure.
Most of its nodes are set in the model tree and merely referred to in successive shots. Its first node, called "DP_MAP", contains pairs of data point "WinCC OA names"

Figure 5.10: Overview of the WinCC OA - MDSplus software backend.

and "indexing names" for each data point belonging to the TrendTree. Data point "indexing names" are then used to order the data point structures in the "DPES" node.

Each of these data point structures contains, in turn, several sub-nodes. The most important ones used in the model tree are the "NAME" node, containing the WinCC OA data point name; the "ID" node, containing the WinCC OA data point ID; the "VARIABLE_TYP" node, containing the WinCC OA data point variable type; the "TYPE_NAME" node, containing the WinCC OA data point type name; and the "CREATION_TIM" node, containing the data point absolute creation time in the TrendTree. The other nodes are optional and store data related to data point aliases, descriptions and units. Each time a new data point is created, modified or deleted, the "DP_MAP" node and the related data point node structure are updated accordingly.

Going back in the node hierarchy of a couple of levels, the "ARCHIVE_DATA" node can be found. It aims to save parameters regarding the related WinCC OA Event Group, sent to the backend upon creation.

The "TIME_MAP" node is instead of paramount importance. It stores information

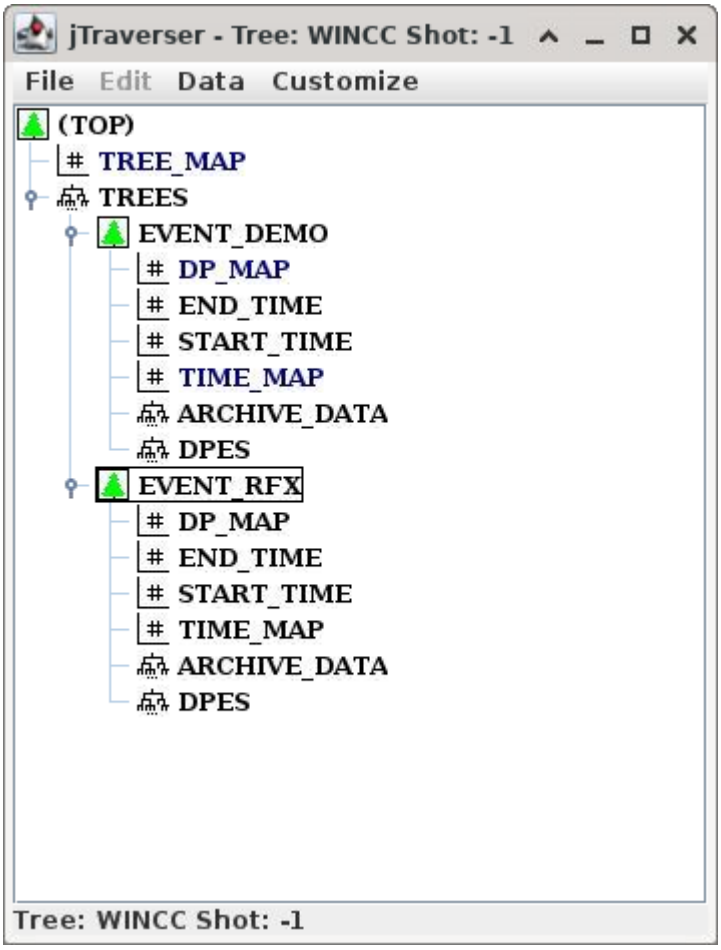


Figure 5.11: Overview of the MDSplus MasterTree (in this case named WinCC).

on all successive TrendTree shots used to store data point samples. For each successive shot, three parameters are saved: the shot number, start time, and end time. By default, each shot lasts one day. Each time a new shot is created, this field is updated accordingly.

The successive TrendTree shots store shot information and data point samples using the remaining nodes. The "START_TIME" and "END_TIME" nodes' purpose is to store the shot start and end times, respectively. The duration of a TrendTree shot is currently fixed to 24 hrs (if necessary, this will become a configurable parameter). A new TrendTree shot is not created by default after the current one expires its end time, but every time a received data point sample's timestamp exceeds the current TrendTree shot's end time.

Finally, returning to the data point structure node, the "VALUES" node is probably the most important: the MDSplus signal-type node where data point samples (values and absolute time) are stored.

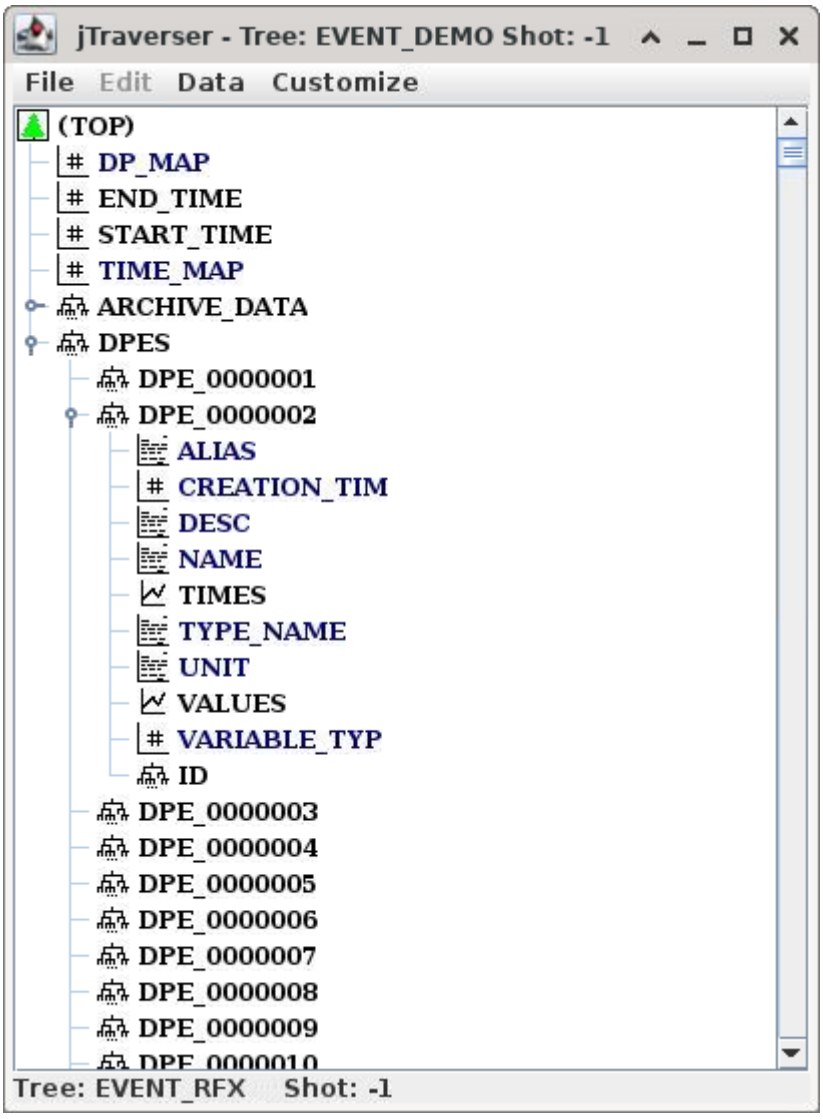


Figure 5.12: Overview of a MDSplus TrendTree (in this case named Event_RFX).

Having discussed the MasterTree and the TrendTree structure thoroughly, it is possible to explain the overall backend behaviour when prompted by the frontend (WinCC OA):

- **Event Group creation**: When a new MDSplus related Event Group is created, the NGA informs the backend, which creates a new TrenTree and links it to the MasterTree.
- **Archived data point creation/modification/deletion**: When a new data point is set to be archived in an MDSplus related Event Group, the backend reacts by selecting the appropriate TrendTree. It then updates its "DP_MAP"

node in the model tree and creates a new data point structure accordingly. Similar considerations hold for data point modification and deletion.

- **Archiving a data point sample**: When a new data point sample is ready to be archived, the backend first finds the corresponding TrendTree (or the corresponding ones, since a sample can be related to more than one TrendTree). It then looks at the sample time value, deciding whether it should be archived in the latest TrendTree (also known as *current tree*) or if a new TrendTree shot must be created (in case its timestamp exceeds the current shot end time). Having selected or created the appropriate TrendTree shot, the backend selects its "DP_MAP" node to find the corresponding data point structure, then stores the sample in the corresponding "VALUES" node. The process is similar if the backend receives a request to archive several data point samples.

- **Reading back a data point sample**: The last relevant backend operation happens when the frontend requests the values of a data point (or more data points) in a specific time frame. The backend selects the TrendTree related to the data point. It then uses the "TIME_MAP" node to select all TrendTree shots related to the requested time frame. For each TrendTree shot, the backend then uses the "DP_MAP" node to access the "VALUES" node related to the requested data point and retrieve its samples. All the requested samples, possibly coming from different TrendTree shots, are joined together and sent back to the frontend as a whole signal.

Observing Figure 5.10, it is possible to see that one crucial backend feature still has not been explained: *ShotTrees*.
The purpose of the ShotTree arose to match how experimental data is saved in RFX MDSplus pulse files. Data related to each RFX experimental pulse is saved in its own MDSplus shot file, meaning a new shot file is created each time. These experimental shots do not use an absolute time reference: each shot uses a relative time reference, its origin coinciding with the pulse starting instant. Consequently, experimental data samples use this relative time reference as well.
Some data points related to signals coming slow control and plant supervision generate values also when no experimental shot is taking place. Archiving these samples is therefore necessary not only during experimental shots, but in a continuous manner. The TrendTree archiving mechanism fully covers this necessity.
However, when an experimental shot takes place, archiving data point samples using the relative time reference used by the shot makes correlating them to experimental

data much easier. In principle, it is possible to extract shot related samples archived in a TrendTree and to convert their time reference on the fly during data visualisation to correlate them with other experimental signals. Nevertheless, this process makes the display process more complex, leading users to pledge for more immediate integration in mixed visualisation of trend data point samples and experimental signals.
The ShotTree serves this purpose: archived data points are stored in its "VALUES" nodes during an experimental shot, using the relative experimental shot time reference. Creating this kind of pulse file duplication implies negligible resource overhead but greatly improves the usability of such signals when correlating them with experimental data.
Consequently, each TrendTree has a twin ShotTree associated with it. The ShotTree structure is constantly synchronised with the one of its corresponding TrendTree. Archiving data point samples to a ShotTree is similar to storing them in TrendTrees, but there are a few key differences.
A new ShotTree MDSplus tree shot is generated in correspondence with every new experimental shot, reflecting the shot creation mechanism used for experimental RFX pulses. In addition to storing datapoint samples using the relative shot time, this strategy also permits linking ShotTrees as subtrees directly in the main experimental RFX MDSplus tree files.
Secondly, data point samples archived in ShotTrees are never returned to the frontend: other tools, such as jScope, are used to analyse them, together with other experimental data.
The backend must be informed of the start of a new experimental shot to archive data point samples according to the relative shot time reference: knowing the shot start time makes it possible to scale data point timestamps from absolute to relative shot time. This information is passed to each ShotTree thanks to MDSplus event listeners: each ShotTree has its own. Therefore, each ShotTree is always informed about shot start and end times and can scale timestamps accordingly. The "TIMES" node belonging to the data point structure node is an auxiliary node for this time reference conversion.
Overall, the WinCC OA-MDSplus backend allows data points to be archived as a continuous trend with an absolute time reference or in correspondence with an experimental pulse using relative pulse time. Such an approach ensures that all experimental information is consolidated within a single data acquisition system.

The backend code, fully developed in object-oriented C++, is organised in sep-

arate namespaces and classes to improve the overall structure and functionality. The first two, already provided in their initial version by ETM, respectively named *DemoBackend* and *DataReceivedHandler*, handled communication with WinCC OA and had to be customised to account for MDSplus function calls. These function calls are contained in the *MDSplusHandler* namespace, developed from scratch to handle MDSplus communication. Moreover, dedicated *MasterTree*, *TrendTree* and *ShotTree* classes exist to support respectively the MasterTree, TrendTree and ShotTree concepts.
Appendix B comprehensively analyses the backend code to avoid making the discussion in this chapter overly detailed.

The backend is in its final development, reaching the stress test phase. Having proven efficient in archiving and retrieving small batches of data points, it is now paramount to see how it performs when dealing with thousands of them at once and at sub-second archiving frequencies (10-100 Hz) to understand its current limitation in performance terms.
Figure 5.13a reports a jScope window with an example of TrendTree data point archiving: it shows the archived values of a single data point chosen from a test performed with 2000 data points simultaneously archived with a frequency of 10 Hz. All the data points involved in this test were of float type. Their samples were generated using a trigonometric function using a WinCC OA CTRL script.
Thanks to the backend, observing how the data point samples are correctly archived into the MDSplus tree file is possible.
Similarly, Figure 5.13b shows a WinCC OA panel with a trend reporting the same data point values: this time they have been successfully read back from the TrendTree to WinCC OA thanks to the backend.
This example shows that when prompted, the backend can successfully store data to MDSplus and retrieve data to WinCC OA. This result will guarantee a deep integration between PLC slow control and the scientific data management system.

5.8 Conclusions

The updated RFX-mod2 CODAS must conform to new demands while facing various limitations. Essential requirements involve handling more electromagnetic (EM) signals and phasing out components based on outdated technologies.
A notable limitation of this project is the restricted budget. As a result, developing the new CODAS has required multiple compromises, with innovative solutions

incorporated only when strictly necessary.
Situations necessitating new developments included instances where the existing system could not accommodate the advanced needs related to EM signals or when it was no longer feasible to maintain outdated components like CAMAC, S5 PLCs, and FactoryLink.
However, architectural choices made nearly two decades ago have proven durable and greatly eased the upgrade of CODAS. Key design decisions that have been particularly effective in RFX-mod include:

1. **Adopting MDSplus and MARTe2 Frameworks:** The use of MDSplus and MARTe2 frameworks, common in various labs and actively maintained, has protected the system against obsolescence.

2. **Selecting CPCI Technology for Hardware:** Choosing CPCI technology for major hardware components has been beneficial. The widespread use of CPCI and the market availability of CPCI bridges have eased the transition from older embedded systems.

3. **Deploying Linux MRG (Linux Messaging, Realtime and Grid) and Multicore Servers:** Upgrading in 2012 to Linux MRG and multicore servers for running the Plasma Control System has been a forward-looking choice for real-time control.

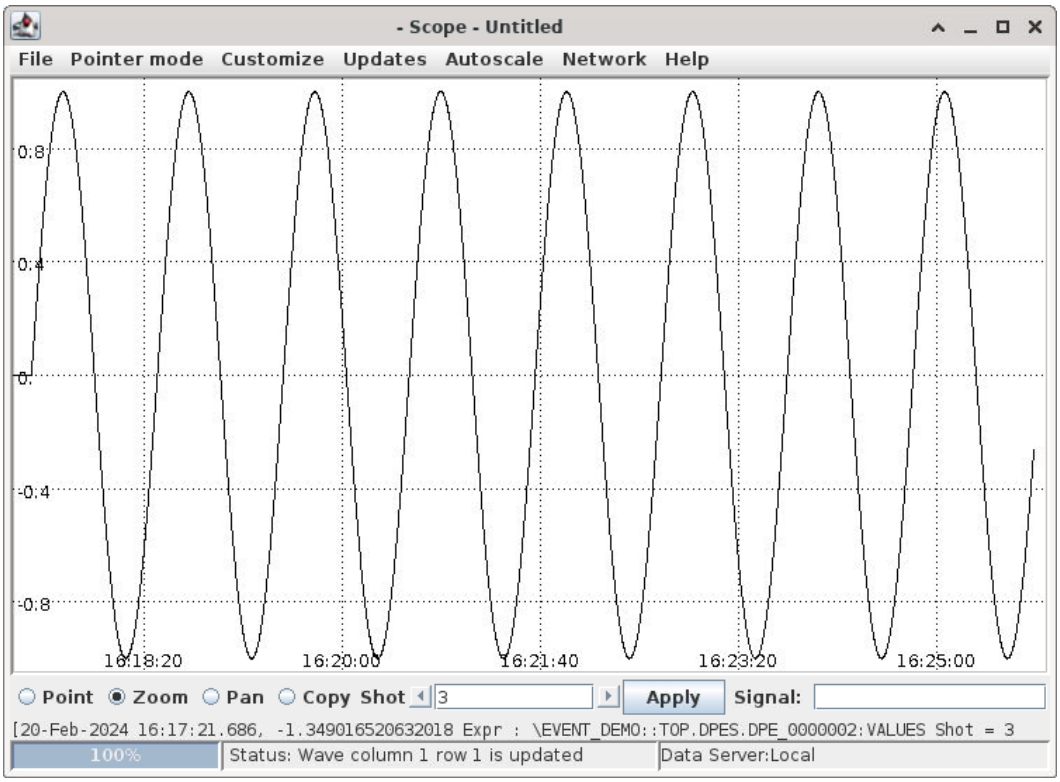
4. **Employing UDP for Real-Time Plasma Control Communication:** Using User Datagram Protocol (UDP) for real-time plasma control communication has proven to be a resilient decision. Despite some experiments using different technologies like reflective memories, Ethernet's mainstream status ensures its longevity and continuous performance enhancement. This strategy is consistent with the technology used in ITER, underscoring its effectiveness.

5. **Maintaining a Hierarchical and Distributed Plant Control Architecture:** Modularity in replacing obsolete hardware and software has enabled resource conservation while keeping several components intact.

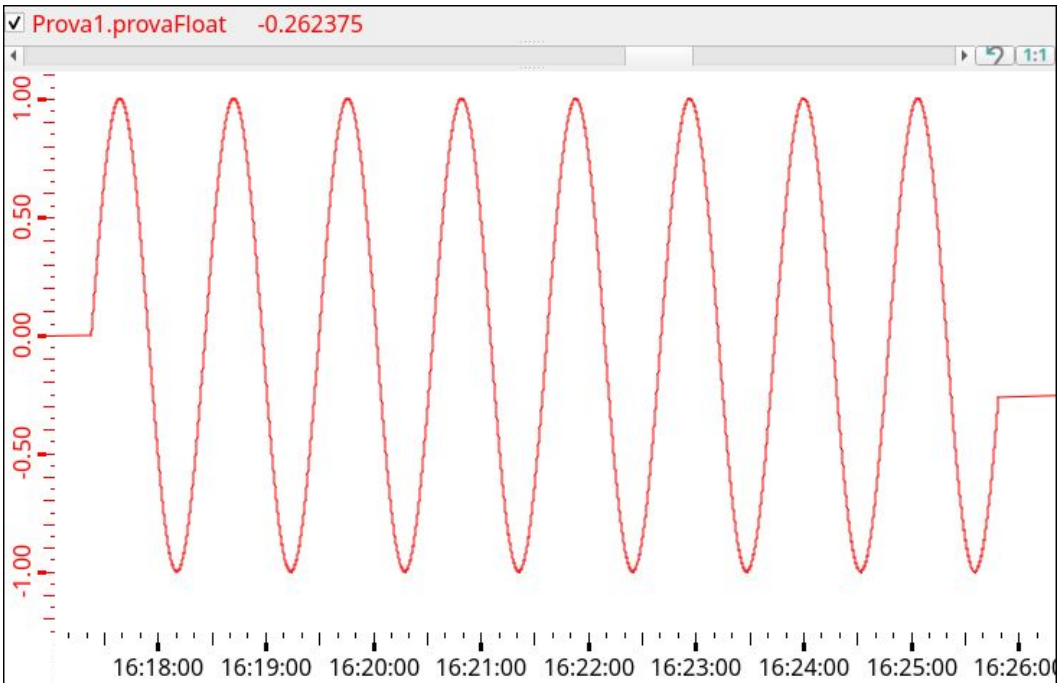
The comprehensive upgrade of the RFX-mod2 CODAS represents a significant stride in fusion research technology. This chapter has outlined the multifaceted aspects of the upgrade, from adopting innovative FPGA-based architectures for signal processing to transitioning to modern SCADA systems like WinCC OA. Each aspect of the upgrade, whether the network infrastructure improvements, the refinement in plasma control, or the advancements in plant control and supervision, contributes

to a more robust and capable system.
Key challenges addressed in this upgrade include the integration of a larger volume of electromagnetic signals, the replacement of outdated components, and the need to maintain compatibility with existing systems. The strategies employed to overcome these challenges, such as developing the WinCC OA-MDSplus backend, illustrate the project's commitment to innovation while respecting budgetary constraints.

Reflecting on the journey of CODAS from RFX-mod to RFX-mod2, it is evident that the strategic decisions made in the past have been instrumental in facilitating this transition. The use of enduring frameworks like MDSplus and MARTe2, the selection of reliable hardware technologies, and the adoption of scalable communication protocols have all played a critical role in ensuring the system's adaptability and longevity.
In conclusion, the RFX-mod2 CODAS upgrade is more than a technical enhancement; it is a testament to the power of foresight, adaptability, and innovation in pursuing advanced fusion research. As the project moves forward, it sets a benchmark in system upgrades, balancing legacy strengths with new technological advancements and paving the way for future developments in the field of fusion energy.



(a) Archived values of a float data point in the test "EVENT_DEMO" TrendTree, seen through jScope.



(b) The same data point values as seen by a WinCC OA trend.

Figure 5.13: WinCC OA-MDSplus backend data point archiving example.

CHAPTER 6

DISCUSSION: A COMPARATIVE OVERVIEW OF CODAS STRATEGIES FOR SPIDER, MITICA AND RFX-MOD2

This chapter presents a comparative analysis of the CODAS strategies of the SPIDER, MITICA, and RFX-mod2 projects. It begins by comparing the main characteristics of their network architectures (Section 6.2), assessing their impact on system flexibility and scalability. The exploration then shifts to the diverse data acquisition strategies employed (Section 6.3), followed by an in-depth look at the real-time control mechanisms (Section 6.4). The nuances of plant control and supervision are scrutinised next (Section 6.5) before diving into the complexities and choices around SCADA systems (Section 6.6). The chapter concludes with a reflection on the collective impact of these strategies on the future of fusion research CODAS (Section 6.7). This analysis aims to highlight the tailored approaches essential in advancing fusion experiments.

6.1 Introduction: Comparing the Different Approaches

The evolution of Control and Data Acquisition System (CODAS) in fusion research is critical in advancing the field. The CODAS architectures of SPIDER, MITICA and RFX-mod2, presented in Chapter 4 and Chapter 5 respectively, offer a valid starting point for an in-depth comparison, highlighting their unique contributions and challenges and assessing the optimal strategies for current and future CODAS development in large experimental fusion facilities. These projects, integral to the progress in fusion technology, utilise different approaches in their CODAS implementations, reflecting their diverse experimental requirements and constraints.

The discussion will focus on critical aspects such as Supervisory Control And Data Acquisition (SCADA) systems, network architecture, plant control, real-time control, and data acquisition. These components are essential for the successful operation and data management of fusion experiments. By comparing the strategies of SPIDER and MITICA with RFX-mod2, the analysis aims to understand how different approaches address the balancing of experimental requirements and industrial robustness of each project while contributing to the overall field of fusion research. The insights drawn from this analysis will showcase the technological advancements in each project and shed light on potential directions for future fusion experiments.

6.2 Network Architecture: Adapting to Project Specifics

The approaches to network infrastructure in SPIDER/MITICA and RFX-mod2 reveal a nuanced blend of shared foundations and distinct strategies, each tailored to their specific experimental demands.
All three experiments have adopted Ethernet technology as the backbone for their network infrastructures, reflecting its reliability and widespread support. This common ground extends to their adoption of specialised network segments, delineating the varied functionalities within these complex systems, from data acquisition to safety mechanisms, thereby enhancing efficiency and reliability.
Significant focus is also dedicated to security, with the adoption of dedicated networks catering for different experimental needs, either physically independent or segregated through VLAN technology.

However, the divergence in their network designs is notable. SPIDER/MITICA, for example, employ network architectures that align with their open-source SCADA systems, adopting specific network protocols like EPICS (Experimental Physics and

Industrial Control System) for Channel Access (CA) over TCP-IP multicast to tailor their network architecture to meet ITER's requirements. This approach is further exemplified in its use of Profinet and Profibus for safety and interlock systems, underscoring a commitment to compliance with rigorous safety standards.
For the transmission of real-time data, they will utilise Ethernet-UDP, following the ITER SDN model. Simultaneously, the DAN will handle the essential high-bandwidth data flows for extensive data acquisition tasks.
A notable difference lies in SPIDER's and MITICA's DAN implementation. SPIDER employs mdsip over TCP-IP to ensure compatibility with the MDSplus data transfer protocol for its DAN. This choice reflects the unavailability of ITER's CODAC specifications at the time of its implementation. MITICA is instead set to follow a different path, implementing both mdsipMDSplus and EPICS for data acquisition. Thus, it will be able to harness MDSplus performance and vast use case experience and conform with ITER's CODAC guidelines. This double configuration is granted by using MARTe2 as an intermediate layer on data acquisition drivers, which allows for switching the acquisition channel.
Another fundamental characteristic is represented by the Time Communication Network (TCN), which synchronises data acquisition. Implementing PTP and relying on a GPS-synchronized grand-master clock, this network represents the state of the art of time synchronisation. SPIDER's TCN only has one central GPS-synchronized National Instruments device implementing PTP. This device generates a trigger distributed using optical fibre to trigger all other devices connected on the TCN.
On the contrary, MITICA follows ITER's CODAC guidelines, fully implementing PTP on it TCN: all modules generate a synchronous trigger at the desired instant.

In contrast, RFX-mod2's network infrastructure, marked by its recent upgrade to a 10 Gbit/s fibre optic network, signifies a strategic push towards accommodating higher data volumes and faster processing capabilities. This forward-looking approach is complemented by implementing VLANs as defined by IEEE 802.1Q standards, optimising traffic flow and functional separation.
Similarly, this VLAN network architecture is split into separate networks, each responsible for its specific area: the PON for plant operation and control; the DAN relies on MDSplus for high-bandwidth data acquisition; and the SDN for real-time plasma control communication via multicast UDP. This publisher-subscriber approach is very flexible when a message has to be communicated to a non-fixed number of devices while maintaining the real-time performances of UDP. It is the same strategy adopted for ITER's SDN.

Additionally, communication between the SCADA system and PLCs, based on the SIMENS S7 protocol and OPC UA, is handled by a dedicated VLAN. This feature speaks to RFX-mod2's adaptability and scalability and aligns it closely with current industry best practices.
Considering time synchronisation, RFX-mod2 currently adopts a timing system based on optical fibre distribution. This system is not as advanced as the PTP one of SPIDER and MITICA, but it is fully functioning after it transitioned from CAMAC to CPCI technology. It will, therefore, be maintained due to its satisfactory performance and current resource limitations. Moreover, while PTP technology is state-of-the-art, to this moment, it has never been fully deployed on an operating nuclear fusion experimental facility.

Overall, the CODAS network implementations of SPIDER, MITICA, and RFX-mod2 present common choices but differ in several aspects, emphasising the importance of modern network standards for high reliability, future scalability and data handling needs.
All these factors contribute to underlining how the growing complexity of modern fusion experiments calls for introducing increasingly industrial reliable solutions.

6.3 Data Acquisition Strategies: Addressing Volume and Complexity

The data acquisition systems of SPIDER & MITICA and RFX-mod2 present notable contrasts and similarities shaped by their specific operational requirements and technological choices.

As mentioned in Section 6.2, SPIDER and MITICA adopt MDSplus for data acquisition, despite being integral to the ITER project. In SPIDER's case, ITER's specifications were still unavailable, leading to employing MDSplus for its widespread and tested usage in several real case scenarios. Concerning MITICA, this choice also stems from the necessity to handle the voluminous and complex data typically generated in nuclear fusion experiments. Conforming to ITER's CODAC standards for the DAN, EPICS will be used for MITICA's data acquisition system, in parallel with MDSplus. However, experience shows how it might need support in managing such extensive experimental data. This distinction is crucial, as EPICS is more adept at managing plant signals than the large-scale experimental data inherent in fusion research. EPICS is based on an elegant distributed client/server approach.

However, it may lead to excessively high numbers of PV subscriptions in large facilities where millions of PVs are deployed across subsystems, which could cause the network to be overloaded by PV event messages. On the other hand, WinCC OA exhibits a more hierarchical approach: it can be distributed across several systems, each having layers employing dedicated drivers and event managers acting as a filter to reduce the overall network payload. Consequently, integrating MDSplus becomes essential in effectively managing these experiments' complex data landscape.
The SPIDER and MITICA data acquisition systems introduce an innovative aspect that lies in the ability to shift adaptively between different sampling rates based on operational conditions. While normally operating at lower sampling frequencies for routine monitoring, this system is designed to automatically increase its sampling rate to several megahertz during critical events like grid breakdowns. This event-driven approach represents a significant advancement in fusion research data acquisition, as it captures detailed, high-frequency data precisely when it is most needed, enabling a more nuanced analysis of these pivotal moments.

In contrast, RFX-mod2 data acquisition system is fully based on MDSplus, given its state-of-the-art performance and widespread support.
Moreover, it introduces significant hardware innovations for this system, enhancing its capability to handle large data volumes and reflects a progressive step in experimental fusion research.
Integrating Field-Programmable Gate Array (FPGA) technology represents a significant technological advancement in real-time control and data acquisition. As delineated in Section 5.2.1, the FPGA approach innovatively addresses the complex demands of electromagnetic signal collection and processing. This framework allows for dual functionality: high-speed data collection for extensive analysis and low-speed data sampling crucial for real-time plasma control. Moreover, the FPGA's ability to handle digital integration obviates the need for a separate analog integration front end, streamlining the system and reducing hardware redundancy. The FPGA architecture's efficacy in numerical integration, coupled with advanced ADC stages, ensures precise data capture and integration, pivotal for accurate plasma control and analysis. This approach adopts numerous design elements from the ITER EM acquisition system [105] yet achieves substantial simplification and cost reduction for two primary reasons: (i) the brief duration of pulse operations (lasting up to 1-2 seconds) obviates the need for advanced techniques like chopping to manage integration drifts, and (ii) there is no necessity for high-speed data streaming during data acquisition, allowing for the signals to be stored locally and accessed

post-pulse.
This leap in data acquisition technology not only enhances the efficiency and accuracy of the RFX-mod2 experiments but also sets a new standard for future fusion research endeavours. Another pivotal development in RFX-mod2 is the seamless integration of MDSplus with the WinCC OA system through a custom-developed backend. This integration represents a significant step forward in data acquisition and management. It enables more efficient analysis and handling of experimental data by storing slow control data directly in MDSplus, thus simplifying the overall architecture by removing the intermediate layer of a dedicated OPC server. Moreover, thanks to the backend, it is possible to archive slow control data directly correlating it with experimental data using the relative pulse time reference.

Despite their different approaches, both systems are designed to face the challenges of managing large and intricate experimental data sets. SPIDER and MITICA's reliance on MDSplus is a direct response to the limitations of EPICS in handling extensive experimental data. This specialisation ensures efficient management and storage of the vast experimental data generated by these experiments.
On the other hand, RFX-mod2's strategy of integrating advanced hardware solutions with MDSplus and WinCC OA offers a more versatile and flexible data handling capability, facilitating efficient data processing and analysis.
Such strategic choices in data acquisition systems underscore each experimental setup's unique requirements and challenges, reflecting fusion research's nuanced and evolving landscape.

6.4 Real-Time Control: Open Source Framework Advancements

Real-time control is a critical component in fusion experiments, dictating the precision and safety of operations.
The control systems employed in SPIDER, MITICA, and RFX-mod2 represent sophisticated approaches to managing complex operational requirements. Each of these projects leverages the MARTe2 framework, albeit in ways that highlight their unique experimental needs and technological advancements.

Through their open-source approach, SPIDER and MITICA have implemented systems that provide considerable adaptability. Though integral to the ITER project, relying on EPICS for real-time control, especially in handling large data sets, could

pose limitations due to its inherent design constraints. As a matter of fact, EPICS has been successfully used for plant control in several applications but never for handling a vast number of PVs such as the one expected in ITER [106].
While this will not likely represent a serious limit in SPIDER, where plant control is limited to handling thousands but not millions of signals, it may become a bottleneck in large-scale fusion devices. Therefore, the MARTe2 framework has been adopted for real-time control in SPIDER and MITICA.
For real-time control, SPIDER and MITICA utilise the MARTe2 framework to handle real-time control tasks crucial for high-voltage operations and plasma initiation processes. EPICS has been proved not to be fit for real-time applications [107, 108], and indeed even in ITER a diffent system is used in the PCS.
In SPIDER, a significant emphasis is placed on managing breakdowns during high-voltage operations. The real-time control system rapidly addresses these breakdowns through effective communication with the power supply units. MITICA extends the use of MARTe2 to a broader range of real-time control functions, maintaining a balance between operational complexity and experimental integrity.

RFX-mod2, on the other hand, takes the integration of MARTe2 a step further. In addition to relying on the MARTe2 framework for advanced plasma control, it introduces the possibility of configuring the framework itself through MDSplus devices. This novel approach allows for a more dynamic and flexible configuration process. By leveraging MDSplus devices, the configuration of the MARTe2 framework can be adapted and fine-tuned more efficiently, catering to the specific needs of RFX-mod2's experiments. This integration signifies a progressive step in fusion control systems, showcasing the potential of combining real-time control with adaptable configuration strategies.
The use of MDSplus in conjunction with MARTe2 in RFX-mod2 highlights a key advantage: the ability to seamlessly integrate the configuration of real-time control systems with the experimental data management system. This integration simplifies the control system configuration process and ensures that the real-time control is in synergy with the experiment's overall data acquisition and management strategy.

In contrast, while SPIDER and MITICA effectively employ MARTe2 for their respective real-time control requirements, they do not utilise MDSplus for real-time control configuration like RFX-mod2. Their approach, more traditional, still provides robust and responsive control systems, yet without the added flexibility offered by the integration with MDSplus seen in RFX-mod2.

In conclusion, while SPIDER and MITICA demonstrate the effectiveness of MARTe2 in handling their experiments' specific real-time control needs, RFX-mod2 stands out for its innovative integration of MARTe2 with MDSplus. This integration enhances the flexibility and adaptability of the real-time control system and aligns it closely with experimental data management, setting a new benchmark in fusion experiment control systems.

6.5 Plant Control: Customisation and Efficiency

The Plant Control and Supervision approach in SPIDER & MITICA versus RFX-mod2 reveals distinct methodologies, each with unique challenges and solutions.
SPIDER and MITICA have adopted a multi-tiered control system that distinctly segments control, interlock, and safety. This division allows each segment to be finely tuned and optimised for its specific role, enhancing reliability and performance. In compliance with ITER guidelines defined in the Plant Control Design Handbook (PCDH), the choice of CODAS technologies adhered as much as possible to the ones chosen by ITER CODAC, focusing on those that are well-established and proven [73]. This approach was particularly pertinent to selecting control and data acquisition hardware, including Siemens PLC slow controllers, PC-based fast controllers, and National Instruments PXI-based data acquisition modules. Using Siemens SIMATIC S7 PLCs is central to their strategy, which aligns with the established industrial standards and reflects a commitment to robust and reliable control mechanisms.

On the other hand, RFX-mod2 presents a contrasting picture. It navigates the challenges of integrating new technologies with pre-existing hardware, a task necessitated by limited resources.
The brilliance of RFX-mod2's strategy lies in its ability to solve compatibility issues innovatively, seamlessly bridging the gap between traditional industrial control systems and open-source software capabilities. This fusion not only maintains the functionality of the existing hardware but also enhances it, setting a new benchmark for adaptability in complex experimental setups. Paramount to this achievement is the modularity of its plant control architecture, preserved since its conception, which has allowed targeted replacement of legacy hardware while respecting budgetary constraints.
Moreover, planning the control architecture on this modular approach constitutes an advantage for future hardware and software replacement. Given the average 30-year

lifespan of a fusion experiment, device substitution due to obsolescence will always be a fundamental issue to address also in other experimental facilities such as ITER, DTT and DEMO.

Another standout feature of RFX-mod2's approach is developing new PLC Scheduler software for the SIMATIC S7-1500 PLCs. This innovative software solution exemplifies a forward-thinking approach to plant control, significantly improving the flexibility and efficiency of the system. Its most prominent features are its ease in configuring a new SIGMA device in terms of connection and FSM parameters. The possibility to set a custom number of PLC communication partners only by setting their network parameters stands out as deeply innovative in an environment where fixed network configurations are customary. Similarly, introducing a customisable hierarchical FSM to synchronize plant coordination PLC devices showcases great versatility.
This innovative software and hardware strategy addresses the immediate needs of plant control and paves the way for future enhancements and upgrades, demonstrating a sustainable approach to managing limited resources.

In conclusion, while SPIDER and MITICA have taken a path of specialised segmentation in their control systems, RFX-mod2 has charted a course of innovative integration, brilliantly overcoming compatibility challenges and setting a precedent for future advancements in plant control systems within the fusion research domain. It is worth mentioning that the adoption of industrial devices for plant control, such as the Siemens PLCs, has become paramount to handling the complex architecture of these facilities, consisting of several plant systems.

6.6 SCADA Systems: Balancing Open-Source and Industrial Solutions

Choosing the best SCADA solution for large-scale fusion experiments like SPIDER, MITICA, and RFX-mod2 is challenging, requiring a delicate balance between technical capabilities, system integration, and scalability. The decision-making process often starts with a comprehensive evaluation of available SCADA options.

The choice of SPIDER's SCADA system exemplifies this process, as detailed in [109]. After testing various commercial and open-source SCADA packages, it concluded that the final decision would be made between WinCC OA, a commercial

SCADA, and EPICS, and open source one, based not only on the technical test results but also considering the project's time constraints and operational schedule. Ultimately, EPICS was chosen as the primary SCADA framework for the SPIDER and MITICA experiments. This choice was mandated to align with the ITER's CODAC (Control, Data Access and Communication) system, setting a standard that facilitated integration with broader ITER systems. EPICS's open-source nature offered considerable customisation and adaptability, which was crucial for the unique requirements of these projects. Furthermore, SPIDER and MITICA integrated MDSplus for data acquisition and management, ensuring a comprehensive control and data handling framework.

In contrast, RFX-mod2 adopted Siemens' WinCC OA for its SCADA system. This pivotal decision marked a strategic shift towards a blend of open-source and industrial solutions, capitalising on WinCC OA's flexibility and industrial robustness. WinCC OA provided a robust and adaptable platform capable of handling RFX-mod2's complex requirements, ensuring reliability and scalability. Additionally, RFX-mod2 introduced an innovative backend integration with MDSplus. This advancement facilitated efficient data acquisition and management, addressing a critical challenge in the project.

Comparing EPICS/CODAC with WinCC OA reveals distinct approaches to SCADA implementation. EPICS, while offering high customisation and alignment with ITER's broader systems, shows limitations in scalability, especially when dealing with complex interactions between multiple Process Variables. Linking PV records together in EPICS to reproduce simple behaviours is not straightforward, even when using the graphical tool. The SDD Editor of the CODAC Core System introduces significant improvements over the standard EPICS interface for programming PV interactions. However, it still faces challenges in implementing complex PV logic involving several variables. PV interaction is achieved by separately configuring their respective panels: an overall view is still missing. Similarly, the SNL used for developing EPICS PV state machine code for the CODAC Core System is less powerful and flexible than WinCC OA's CTRL language for script generation. While it allows reading and setting the values of multiple PVs and performing PV logic operations, it is limited to the creation of state machines.
On the other hand, WinCC OA's CTRL language offers more robust scripting capabilities, allowing for more intricate data point logic implementation, a crucial factor for large-scale facilities like RFX-mod2. This superiority in handling complex control

logic and script development is a significant advantage of WinCC OA over EPICS. Not to be overlooked is WinCC OA's NGA feature. The NGA made the direct integration with MDSplus possible for slow control data acquisition through the development of the custom backend.
WinCC OA's native ability to communicate directly with Siemens PLCs through the S7 protocol and the possibility to instantiate integrated OPC UA servers are two other salient points which must be taken into consideration when choosing the best SCADA solution.
Moreover, WinCC OA has a proven history of success in vast industrial facilities, translating into a vast collection of diverse and complex use cases and dedicated support from expert software developers.

In summary, choosing SCADA systems in these fusion experiments underscores the need for a tailored approach. While SPIDER and MITICA benefit from EPICS's flexibility and ITER alignment, RFX-mod2's choice of WinCC OA represents an optimal balance between open-source adaptability and industrial-grade reliability, enhanced further by the novel backend integration with MDSplus.

6.7 The SPIDER, MITICA and RFX-mod2: A Foundation for Future Fusion Experiments CODAS

The CODAS implementations in SPIDER and MITICA lay a foundation for future fusion research CODAS developments. While their reliance on open-source solutions like EPICS provides a high degree of flexibility, the challenges encountered in handling large-scale data acquisition and complex real-time control necessitated reevaluating these systems, as reflected by the introduction of MARTe2 for real-time control needs.
The experiences from SPIDER and MITICA have been instrumental in shaping the approach adopted in RFX-mod2, providing valuable lessons in balancing flexibility with robustness.

The shift towards a hybrid approach in RFX-mod2, integrating the adaptability of open-source systems such as MDSplus with the robustness of industrial solutions such as WinCC OA, offers a viable model for the next generation of large-scale fusion experiments, such as the DTT (Divertor Tokamak Test facility) and DEMO (DEMOnstration Power Plant). This approach addresses the critical challenges of scalability, data integrity, and system reliability, which are crucial for the success of

complex and large-scale fusion projects. The insights gained from RFX-mod2's CODAS strategy could significantly influence future fusion research, guiding the design and implementation of CODAS.

Despite these significant achievements, the road ahead is still challenging: the development of MITICA's and RFX-mod2 CODAS is still underway.
MITICA's CODAS is undergoing implementation. The next step will consist of configuring the MARTe2 devices for the data acquisition system to allow the switching between MDSplus and the EPICS based DAN. Moreover, its CODAC servers are being updated to the newly released version 7.
Concerning RFX-mod2, the hardware upgrade to control plant systems and coordination is constrained by PNRR (Piano Nazionale di Ripresa e Resilienza) funding and, therefore, often delayed. However, the innovative Scheduler software development has reached initial testing, while the WinCC OA-MDSplus backend is undergoing performance stress tests to push to the limit its capabilities.
These advancements and careful upgrade planning will surely overcome contingent limitations and lead to the successful deployment of both state-of-the-art CODAS architectures.

In summary, the comparative analysis of CODAS strategies in SPIDER, MITICA, and RFX-mod2 reflects the dynamic evolution of CODAS in fusion research. The transition from a predominantly open-source approach to a more balanced integration of open-source and industrial solutions, as seen in RFX-mod2, offers a comprehensive model for addressing modern fusion experiments' diverse and escalating demands. This evolution in CODAS implementation provides critical insights and a roadmap for future projects in fusion research, emphasising the need for an integrated approach that combines flexibility, precision, and robustness.

Part III

Epilogue

CHAPTER 7

CONCLUSIONS

The evolution of Control and Data Acquisition System (CODAS) in these fusion projects illustrates a remarkable technological adaptation and innovation journey. SPIDER and MITICA, initially relying on predominantly open-source solutions, transitioned to a balanced integration of open-source and industrial solutions in RFX-mod2. This shift signifies a crucial step toward enhancing system robustness and scalability, catering to the growing complexities of fusion experiments.

Regarding network architecture and data management, all three projects rely on Ethernet technology and specialized network segments, emphasizing the importance of reliable and scalable infrastructures. The adoption of MDSplus in SPIDER/MITICA for efficient data handling and the innovative integration of FPGA technology in RFX-mod2 highlight the necessity to manage extensive and complex experimental data effectively.
Implementing the MARTe2 framework across these projects underscores its effectiveness in real-time control tasks crucial to fusion experiments. RFX-mod2's innovative approach to integrating MARTe2 with MDSplus for dynamic configuration sets a new standard in real-time control system design.

Plant Control and Supervision strategies also reflect significant advancements. The multi-tiered control system in SPIDER and MITICA, featuring Siemens SIMATIC S7 PLCs, aligns with industrial standards, ensuring system reliability and precision. In contrast, RFX-mod2's approach to integrating new technologies with existing hardware showcases a commitment to technological adaptability and resource effi-

ciency. This strategy has also been followed while developing the new SIGMA PLC Scheduler software, which builds on compatibility to introduce innovative aspects in scalability and ease of configuration.

The selection of SCADA systems further reveals the balance between technical capabilities and system integration. EPICS was chosen in SPIDER and MITICA for its flexibility and integration with ITER's CODAC system. At the same time, RFX-mod2 adopted Siemens' WinCC OA, highlighting a strategic move towards blending open-source adaptability with industrial robustness. The custom backend integration of WinCC OA with MDSplus in RFX-mod2 exemplifies a strategic approach to harmonizing diverse system requirements.

As new challenges arise from facilities' size and complexity, modern fusion experiments should aim to holistically integrate data acquisition, real-time control, and plant supervision systems. CODAS strategies must evolve, incorporating advanced technologies and frameworks to meet these challenges. The balance between open-source flexibility and industrial-grade reliability remains critical; however, as industrial aspects become more predominant in large-scale experimental nuclear fusion facilities, relying on standard commercial CODAS frameworks may become strategic for complex system integration. In this sense, SPIDER, MITICA, and RFX-mod2 insights provide valuable lessons for upcoming large-scale experiments like the DTT (Divertor Tokamak Test facility) and DEMO (DEMOnstration Power Plant).
In conclusion, the comprehensive analysis of CODAS strategies in these three projects offers a glimpse into the evolving landscape of fusion research technology. The experiences and lessons learned from these projects are invaluable, guiding the development of efficient, reliable, and scalable systems essential for future fusion energy research success.

Appendices

APPENDIX A

RFX-MOD2 SIGMA PLC SCHEDULER SOFTWARE ARCHITECTURE

This appendix explains the main features of the software architecture of the WinCC OA-MDSplus backend. As mentioned in Section 5.6.1, the new version of the SIGMA PLC Scheduler software is designed to maintain compatibility with the precious version while introducing significant conceptual advancements.
The most important is relying heavily on UDTs and nested FB to achieve code reusability, leading to the so-called *Device-FB*. The appendix illustrates the Scheduler code's salient characteristics, starting from its Main OB1 in Section A.1. Then, the is thoroughly analysed in Section A.2.

A.1 Main (OB1)

Starting from the Main (OB1), reported in Listing A.1, the *Device-FB* concept makes running more than one software device on a single hardware one immediate: the only necessary step is instantiating a new *Device-FB* in the Main and configuring its parameters at FB level.

```
1 //MAIN
2
3 //call first device
```

```
4  "DEV_0_DB"();
5
6  //call second device
7  "DEV_1_DB"();
8
9  //call third device
10 "DEV_2_DB"();
11
12 //CALL OTHER DEVICES
```

List of Listings A.1: The Main (OB1) of the new SIGMA Scheduler

A.2 Device-FB

The unexploded structure of the *Device-FB* has been shown in Figure 5.8. Going into further detail, Figure A.1 and Figure A.2 show the sections of the associated DB which must be configured.

Figure A.1 shows the parameters of the current device and its partners FSM settings.
For the current device, the most significant ones are:

1. **ID**: sets the current device's name (example: SS).
2. **Enabled**: sets if the current device is enabled.
3. **IsEssential**: sets if the current device is essential.
4. **IsMaster**: sets if the current device is the master device in the hierarchical FSM architecture.

For a partner device, the most significant ones are:

1. **ID**: sets the partner's name (example: G1).
2. **Enabled**: states if the partner is enabled/disabled.
3. **IsParent**: sets if this partner is the current device's parent.
4. **IsChild**: sets if this partner is a current device's child.

Figure A.2 shows the parameters to configure the communication with partner devices. The most significant ones are:

1. **ID**: sets the partner's communication ID.
2. **ConnectionType**: this parameter cannot be configured. It is set to 16#0*C* because ISO-on-TCP is used.
3. **ActiveEstablished**: sets if the current device or the partner is active in setting the communication.
4. **ADDR**: sets the partner's IP address.
5. **TSel**: these fields are used to set the local and remote *TSelectors*, necessary for ISO-on-TCP communication.

Analysing the related FB code, reported in Listing A.2 is also relevant to the analysis. Its code is efficiently divided into three sections to enhance clarity. These sections are explained in Section A.2. It is worth noting that all called FBs are parameter instance FBs, which means they are nested inside the *Device-FB* static variables field. This characteristic increases modularity and code organisation.

1. **HMI_Communication**: this FB handles HMI communication in case it is foreseen.
2. **PartnerCommunication**: this FB handles the communication with all other SIGMA Scheduler devices, which are either the parent device or children ones.
3. **HMI_Communication**: this FB runs the FSM of the device.

```
1  //DEV_0
2
3  //read system date and time
4  #retDateTime := RD_SYS_T(#FSM.Data.DateAndTime);
5
6  //validate device configuration
7  #Dev_ConfigOK := "ValidateConfiguration"(FSM_Params:=#FSM.Params,
8                                          PartnerDataArr := #
       PartnerDataArr,
9                                          NumOfEnabledPartners => #
       NumOfEnabledPartners);
10
11 //check if device is configured correctly
12 IF #Dev_ConfigOK THEN
13
14     //run communication with HMI
15     #HMI_Communication(Curr_ID := #FSM.Params.ID,
16                        FSM_Data := #FSM.Data,
```

```
17                          Parent := #PartnerDataArr[0],
18                          MOP_Table := #MOP_Table);
19
20      //run communication with partners
21      #PartnerCommunication(Curr_ID := #FSM.Params.ID,
22                            PartnerDataArr := #PartnerDataArr,
23                            FSM_Data := #FSM.Data,
24                            MOP_Table := #MOP_Table,
25                            PulseData := #PulseData);
26
27      //run finite state machine
28      #FSM(PartnerDataArr := #PartnerDataArr,
29           MOP_Table := #MOP_Table,
30           PulseData := #PulseData);
31
32  END_IF;
```

List of Listings A.2: The Scheduler Device-FB of the new SIGMA Scheduler

A.3 Scheduler FSM

Finally, it is relevant to analyse the code of the FSM.
Worth mentioning is how all the called routines are simple Function (FC)s: all the FSM instances related to the different devices running in parallel can utilise them without instantiating their own FB, thus reducing the total amount program blocks and related memory.
Secondly, it can be seen how the FSM structure is based on the three sub-steps (plus intermediate ones) necessary for state transition: *PREP*, *ENTER* and *NEXTEN*. This structure makes it independent from the current state, leaving state-dependent actions to inner secondary functions.
All the data necessary for the FSM, which might change from cycle to cycle, is passed to this FB by reference, thus avoiding direct external FB data access.

```
1  //FSM
2
3  //get number of partners
4  #NumOfPartners := "ArrayLength"(ArrayInput := #PartnerDataArr);
5
6  //get number of children
7  #NumOfChildren := "ArrayLength"(ArrayInput := #Children);
8
9  //copy partner data to parent
10 #Parent := #PartnerDataArr[0];
11
```

```
12 //check NumOfPartners for safety
13 IF ((#NumOfPartners > 0) AND (#NumOfPartners -1 <= #NumOfChildren))
       THEN
14
15     //copy partner data to children array
16     FOR #i := 1 TO #NumOfPartners - 1 DO
17         #Children[#i - 1] := #PartnerDataArr[#i];
18
19         //set number of enabled children
20         IF #Children[#i - 1].FSM_Params.Enabled THEN
21             #NumOfEnabledChildren_Temp := #NumOfEnabledChildren_Temp
       + 1;
22         END_IF;
23
24     END_FOR;
25
26     #Data.NumOfEnabledChildren := #NumOfEnabledChildren_Temp; //set
       number of enabled children
27
28 END_IF;
29
30 //initialize FSM
31 "FSM_Initialization"(InitializationStep := #Data.FSM_Steps[0],
32                      Step := #Data.Step,
33                      PrevStepValue := #Data.Step.PrevValue,
34                      NextStepValue := #Data.Step.NextValue,
35                      InitializationCompleted := #Data.
       InitializationCompleted);
36
37 //check if first cycle for this state
38 "CheckFirstCycleOfState"(CurrState := #Data.FSM_State.CurrState,
39                          StateOfPrevCycle := #Data.FSM_State.
       PrevCycleState,
40                          NumOfPerfCycles:=#Data.Test_NumOfPerfCycles,
41                          State_NPS := #Data.StatesData.NameValues.NPS
       );
42
43 //call MOP management
44 "MOP_Management"(FirstCycleOfState := #Data.FSM_State.CurrState.
       FirstCycleOfState,
45                  State_OFF := #Data.StatesData.NameValues.OFF,
46                  FSM_Params:=#Params,
47                  CurrState:=#Data.FSM_State.CurrState,
48                  MOP:=#Data.MOP,
49                  Alarms:=#Data.Alarms,
50                  MOP_Table := #MOP_Table,
```

```
51                  Children:=#Children);
52
53 //call pulse number and time to shot management
54 "PulseNumTimeToShot_Management"(FirstCycleOfState := #Data.FSM_State.
       CurrState.FirstCycleOfState,
55                                  NumOfChildren := #NumOfChildren,
56                                  CurrState := #Data.FSM_State.
       CurrState,
57                                  State_PPR := #Data.StatesData.
       NameValues.PPR,
58                                  State_INI := #Data.StatesData.
       NameValues.INI,
59                                  PulseData := #PulseData);
60
61 //call watchdog management
62 "Watchdog_Management"(State_OFF := #Data.StatesData.NameValues.OFF,
63                       State_NPS := #Data.StatesData.NameValues.NPS,
64                       FSM_Params:=#Params,
65                       CurrState:=#Data.FSM_State.CurrState,
66                       Alarms:=#Data.Alarms,
67                       Parent := #Parent,
68                       Children := #Children);
69
70 //run FSM
71 //check if alarms are present
72 IF (NOT #Data.FSM_Alarms.InvalidStep) THEN
73
74     //TEST for slow step transition
75     IF "TEST_SlowStepTrans" THEN
76         #Data.Step.TimingData.Countdown := T#1S; //overwrite step
       countdown
77     END_IF;
78
79     //start timer for step transition
80     #Data.Timer_StepTrans.TON(IN := TRUE,
81                               PT := #Data.Step.TimingData.Countdown,
82                               Q => #Data.Step.TimingData.Timeout);
83
84     //start timer for state timeout, if countdown greater than zero
85     #Data.Timer_StateMaxTime.TON(IN := #Data.FSM_State.CurrState.
       TimingData.Countdown > T#0s,
86                                  PT := #Data.FSM_State.CurrState.
       TimingData.Countdown,
87                                  ET => #Data.FSM_State.CurrState.
       TimingData.ElapsedTime,
```

```
88                                  Q => #Data.FSM_State.CurrState.
   TimingData.Timeout);
89
90    CASE #Data.Step.Value OF
91
92        #Step_DISCONNECT: //0
93
94            //call step main routine
95            "DISCONNECT"(retDisconnect := #Data.Step.Completed,
96                         Step_Flags:=#Data.Step.Flags,
97                         FSM_ReqState := #Data.FSM_State.ReqState,
98                         FSM_CurrState := #Data.FSM_State.CurrState,
99                         State_UKN := #Data.StatesData.NameValues.UKN
   ,
100                        FSM_Alarms := #Data.FSM_Alarms,
101                        FSM_StatesData := #Data.StatesData);
102
103           //set next step request
104           #Data.Step.NextValue := #Step_STARTING;
105
106       #Step_STARTING: //1
107
108           //call step main routine
109           "STARTING"(retStarting := #Data.Step.Completed,
110                      FSM_Params:=#Params,
111                      Step_AlreadyEntered := #Data.Step.
   AlreadyEntered,
112                      FSM_CurrState := #Data.FSM_State.CurrState,
113                      NumOfCheckedInChildren:=#Data.
   NumOfCheckedInChildren,
114                      Parent := #Parent,
115                      Children := #Children,
116                      State_UKN := #Data.StatesData.NameValues.UKN,
117                      State_OFF := #Data.StatesData.NameValues.OFF,
118                      State_OFF_Name := #Data.StatesData.Names[#Data
   .StatesData.NameValues.OFF],
119                      State_OFF_NextMask := #Data.StatesData.
   NextMask[#Data.StatesData.NameValues.OFF]);
120
121           //set next step request
122           #Data.Step.NextValue := #Step_REQ;
123
124       #Step_REQ: //2
125
126           //call step main routine
127           "REQ"(retReq := #Data.Step.Completed,
```

```
128                 Step_AlreadyEntered := #Data.Step.AlreadyEntered,
129                 State_ASB := #Data.StatesData.NameValues.ASB,
130                 Alarms := #Data.Alarms,
131                 IsMaster := #Params.IsMaster,
132                 Parent_ReqState := #Parent.FSM_Data.ReqState,
133                 ReqState := #Data.FSM_State.ReqState,
134                 CurrState_Idx := #Data.FSM_State.CurrState.Idx,
135                 Step_Flags := #Data.Step.Flags,
136                 FSM_Alarms := #Data.FSM_Alarms,
137                 FSM_StatesData := #Data.StatesData,
138                 Children := #Children);
139
140         //set next step request
141         #Data.Step.NextValue := #Step_PREP_WAIT;
142
143     #Step_PREP_WAIT: //3
144
145         //call step main routine
146         "PREP_WAIT"(retPrepWait := #Data.Step.Completed,
147                     Children := #Children);
148
149         //set next step request
150         #Data.Step.NextValue := #Step_PREP;
151
152     #Step_PREP: //4
153
154         //call step main routine
155         "PREP"(retPrep := #Data.Step.Completed,
156                Alarms := #Data.Alarms,
157                State_UKN := #Data.StatesData.NameValues.UKN,
158                MOP_Status := #MOP_Table.MOP_Status,
159                MOP_Status_SetCompleted := #MOP_Table.
    MOP_StatusValues.SetCompleted,
160                PrepFlag := #Data.Step.Flags.Prep,
161                Children := #Children);
162
163         //set next step request
164         #Data.Step.NextValue := #Step_ENTER_WAIT;
165
166     #Step_ENTER_WAIT: //5
167
168         //call step main routine
169         "ENTER_WAIT"(retEnterWait := #Data.Step.Completed,
170                      Children := #Children);
171
172         //set next step request
```

```
173             #Data.Step.NextValue := #Step_ENTER;
174
175         #Step_ENTER: //6
176
177             //call step main routine
178             "ENTER"(retEnter := #Data.Step.Completed,
179                     EnterFlag := #Data.Step.Flags.Enter);
180
181             //set next step request
182             #Data.Step.NextValue := #Step_NEXTEN_WAIT;
183             //check if transition to next step is possible
184
185         #Step_NEXTEN_WAIT: //7
186
187             //call step main routine
188             "NEXTEN_WAIT"(retNextenWait := #Data.Step.Completed,
189                           Alarms := #Data.Alarms,
190                           ReqStateIdx := #Data.FSM_State.ReqState.Idx
    ,
191                           Children := #Children,
192                           AlarmStateMismatch := #Data.FSM_Alarms.
    StateMismatch,
193                           Timer_AlarmStateMismatch := #Data.
    Timer_AlarmStateMismatch);
194
195             //set next step request
196             #Data.Step.NextValue := #Step_NEXTEN;
197
198         #Step_NEXTEN: //8
199
200             //call step main routine
201             "NEXTEN"(retNexten := #Data.Step.Completed,
202                      Step_AlreadyEntered := #Data.Step.AlreadyEntered
    ,
203                      FSM_State := #Data.FSM_State,
204                      FSM_StatesData := #Data.StatesData,
205                      FSM_Alarms := #Data.FSM_Alarms,
206                      Step_Flags := #Data.Step.Flags,
207                      Alarms := #Data.Alarms,
208                      Timer_StateMaxTime := #Data.Timer_StateMaxTime);
209
210             //check if restart FSM request
211             IF #Parent.FSM_Data.Commands.RestartFSM THEN
212                 //set next step request
213                 #Data.Step.NextValue := #Step_DISCONNECT;
214             ELSE
```

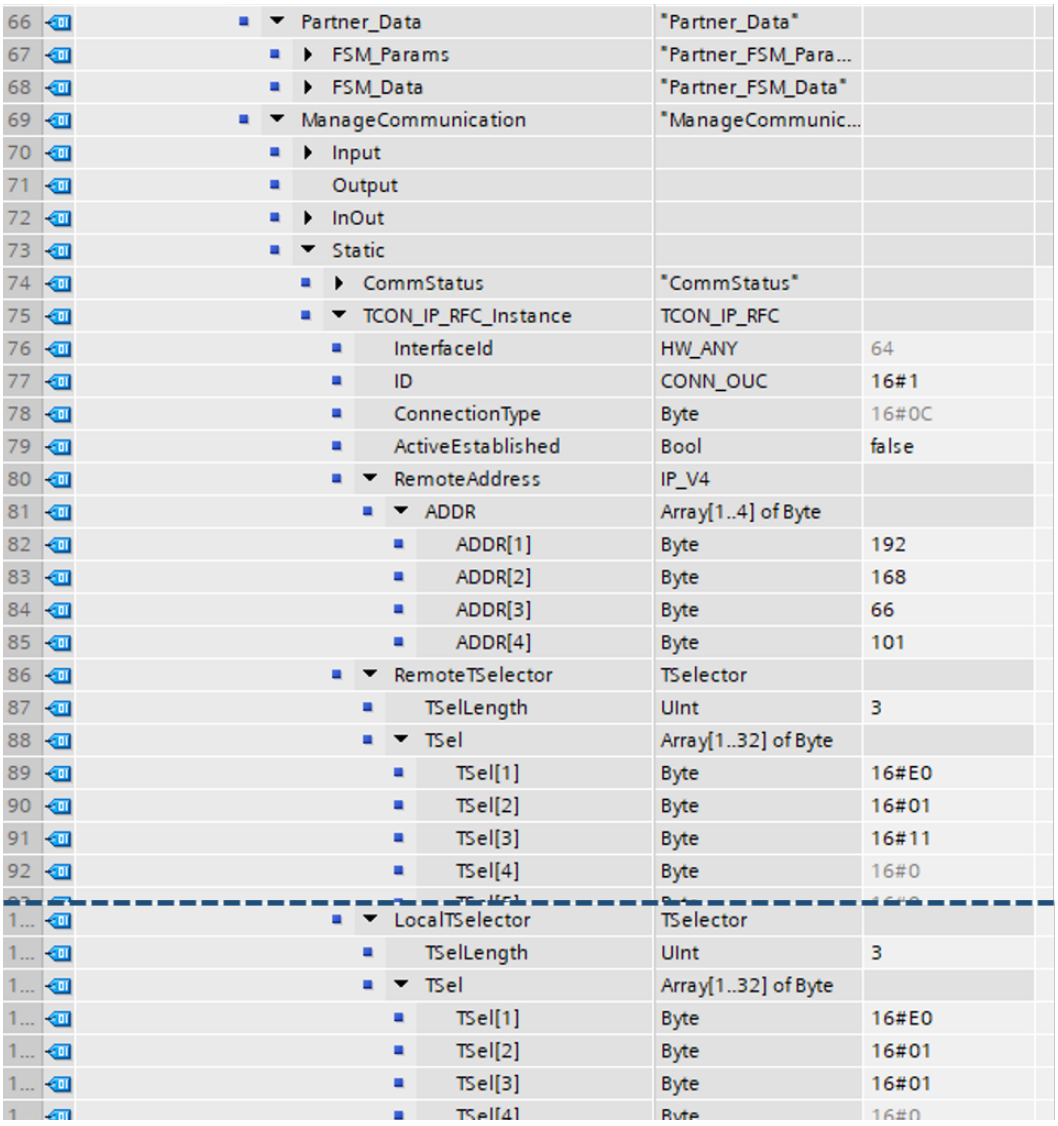
```
215                 //set next step request
216                 #Data.Step.NextValue := #Step_REQ;
217             END_IF;
218 
219         ELSE
220             #Data.FSM_Alarms.InvalidStep := TRUE; //Error! This step
    value is not valid
221     END_CASE;
222 
223     //check if state timeout
224     IF #Data.FSM_State.CurrState.TimingData.Timeout THEN
225         #Data.FSM_Alarms.StateTimeOut := TRUE; //TODO handle this
    error! Abort??
226     END_IF;
227 
228     //check if transition to next step is possible
229     "FSM_StepTrans"(NextStepValue := #Data.Step.NextValue,
230                     FSM_Step := #Data.Step,
231                     FSM_Steps := #Data.FSM_Steps,
232                     AlarmInvalidStateTransReq := #Data.FSM_Alarms.
    InvalidStepTransReq,
233                     Timer_StepTrans := #Data.Timer_StepTrans,
234                     Children:= #Children);
235 
236 END_IF;
237 
238 //call alarm handling
239 "Alarm_Handling"(CurrState_Idx := #Data.FSM_State.CurrState.Idx,
240                  State_OFF := #Data.StatesData.NameValues.OFF,
241                  State_NPS := #Data.StatesData.NameValues.NPS,
242                  Alarms := #Data.Alarms,
243                  FSM_Alarms := #Data.FSM_Alarms);
244 
245 
246 //copy parent to partner data array
247 #PartnerDataArr[0] := #Parent;
248 
249 //copy children to partner data array
250 FOR #i := 0 TO #NumOfChildren - 1 DO
251     #PartnerDataArr[#i+1] := #Children[#i];
252 END_FOR;
```

List of Listings A.3: The Scheduler FSM of the new SIGMA Scheduler

DEV_0_DB

	Name	Data type	Start value	R
1	Input			
2	Output			
3	InOut			
4	Static			
5	Dev_ConfigOK	Bool	false	
6	FSM	"FSM"		
7	Input			
8	Output			
9	InOut			
10	Static			
11	Params	"FSM_Params"		
12	ID	String	'SS'	
13	Enabled	Bool	true	
14	IsEssential	Bool	false	
15	IsMaster	Bool	true	
16	BypassCheckInProcedure	Bool	false	
17	BypassWatchdogProcedure	Bool	false	
18	Bypass_MOP_Procedure	Bool	false	
19	Data	"FSM_Data"		
20	Parent	"Partner_Data"		
21	Children	Array[0..9] of "Partn...		
22	PartnerDataArr	Array[0..10] of "Part...		
23	PartnerDataArr[0]	"Partner_Data"		
24	PartnerDataArr[1]	"Partner_Data"		
25	FSM_Params	"Partner_FSM_Para...		
26	ID	String	'G1'	
27	Enabled	Bool	true	
28	IsParent	Bool	false	
29	IsChild	Bool	true	
30	Standalone	Bool	false	
31	Ignored	Bool	false	
32	FSM_Data	"Partner_FSM_Data"		
33	PartnerDataArr[2]	"Partner_Data"		
34	FSM_Params	"Partner_FSM_Para...		
35	ID	String	'G2'	
36	Enabled	Bool	true	
37	IsParent	Bool	false	
38	IsChild	Bool	true	

Figure A.1: The *Device-FB* FSM configurable parameters.



66	Partner_Data	"Partner_Data"	
67	FSM_Params	"Partner_FSM_Para...	
68	FSM_Data	"Partner_FSM_Data"	
69	ManageCommunication	"ManageCommunic...	
70	Input		
71	Output		
72	InOut		
73	Static		
74	CommStatus	"CommStatus"	
75	TCON_IP_RFC_Instance	TCON_IP_RFC	
76	InterfaceId	HW_ANY	64
77	ID	CONN_OUC	16#1
78	ConnectionType	Byte	16#0C
79	ActiveEstablished	Bool	false
80	RemoteAddress	IP_V4	
81	ADDR	Array[1..4] of Byte	
82	ADDR[1]	Byte	192
83	ADDR[2]	Byte	168
84	ADDR[3]	Byte	66
85	ADDR[4]	Byte	101
86	RemoteTSelector	TSelector	
87	TSelLength	UInt	3
88	TSel	Array[1..32] of Byte	
89	TSel[1]	Byte	16#E0
90	TSel[2]	Byte	16#01
91	TSel[3]	Byte	16#11
92	TSel[4]	Byte	16#0
1...	LocalTSelector	TSelector	
1...	TSelLength	UInt	3
1...	TSel	Array[1..32] of Byte	
1...	TSel[1]	Byte	16#E0
1...	TSel[2]	Byte	16#01
1...	TSel[3]	Byte	16#01
1...	TSel[4]	Byte	16#0

Figure A.2: The *Device-FB* partner communication configurable parameters.

APPENDIX B

WINCC OA - MDSPLUS BACKEND SOFTWARE ARCHITECTURE

This appendix explains the main features of the software architecture of the WinCC OA-MDSplus backend. As mentioned in Section 5.7.1, the initial version of the backend provided by Siemens consisted of an initial version capable of communicating with WinCC OA's NGA.
On the software side, this translates several C++ classes, two of which had to be edited to implement the calls to the developed software for MDSplus communication: their names are *DemoBackend* and *DataReceivedHandler*.
These function calls refer to the first custom-developed namespace which handles MDSplus communication, aptly named *MDSplusHandler*, explained in Section B.1. The *MDSplusHandler*, in turn, calls the function of the other main backend classes, which are the *MasterTree*, the *TrendTree*, and the *ShotTree* ones, illustrated respectively in Section B.2, Section B.3 and Section B.4.

B.1 MDSplusHandler

The *MDSplusHandler* namespace, reported in Listing B.1, contains callbacks for all the main backend functionalities. Its main callbacks are:

1. **initializeHandler** handles the initialization of the whole MDSplus custom-developed section of the backend.

2. **processArchiveGroups** handles the creation of new *Event Archive Groups* in the MDSplus MasterTree.

3. **processMedatada** handles the creation, modification or deletion of data points in the respective MDSplus TrendTrees.

4. **processWriteRequest** handles writing new data point samples to their respective archive structures in MDSplus.

5. **processReadRequest** handles reading all requested data point samples related to a specific time interval from MDSplus TrendTrees and sending their values back to WinCC OA.

```
1  namespace NGA
2  {
3  namespace DemoBackend
4  {
5  namespace MDSplusHandler
6  {
7
8  //initializes handler
9  void initializeHandler(const std::function<void(const std::string &
       message)> &logDebugFunc);
10
11 //gets known archive group data from backend
12 void getKnownArchiveGroups(ArchiveGroupDeltaRequest &
       archiveGroupDeltaRequest, const std::function<void(const std::
       string &message)> &logDebugFunc);
13
14 //processes archive group data received from frontend
15 void processArchiveGroups(const ArchiveGroupDeltaRespone &
       archiveGroupDeltaResponse, const std::function<void(const std::
       string &message)> &logDebugFunc);
16
17 //gets known metadata data from backend
18 void getKnownMetadata(MetadataRequest &metadataRequest, const std::
       function<void(const std::string &message)> &logDebugFunc);
19
20 //processes medatadata data received from frontend
21 void processMetadata(const MetadataDeltaResponse &metadataResponse,
       const std::function<void(const std::string &message)> &
       logDebugFunc);
22
23 //gets most recent timestamp from backend
```

```
24 bool retrieveMaxTimestamp(google::protobuf::Timestamp &maxTimestamp,
       const std::function<void(const std::string &message)> &
       logDebugFunc);
25
26 //writes dp values received from frontend
27 bool processWriteRequest(const DpeList &values, const std::function<
       void(const std::string &message)> &logDebugFunc);
28
29 //processes read request for dp values received from frontend
30 void processReadRequest(const std::function<void(const ReadResponse&)
       >& sendReadResponseFunc, const std::function<void(const std::
       string& message)>& logDebugFunc, const FrontendRequest& request);
31
32 }
33 }
34 }
```

List of Listings B.1: The MDSplusHandler namespace

B.2 MasterTree

The *MasterTree* class, reported in Listing B.2, contains the code related to the MasterTree. Its salient characteristics are:

1. **trendTreeMap**: the map which contains pairs (trendTree name, TrendTree) mapping trend tree names to their respective trendTree objects.
2. **createTreeMap**: function which handles creating the trendTreeMap by reading it from the MasterTree file.
3. **writeTreeMap**: function which handles writing the TrendTree map to the MasterTree file.
4. **addTrendTreeNode**: adds trendTree node to TREES node of the MasterTree and links modelTree of trendTree to created node

```
1 //class for WinCC Master Tree: the tree to master them all TrendTrees
2 //handles all WinCC archive grous and related TrendTrees
3 class MasterTree: public WinCCTree
4 {
5 public:
6     static const std::string DEF_TREE_DIR; //default tree directory
7     static SafeQueue<DpeList> dpeListQueue; //thread safe queue for
      dpeList
```

```
8     static SafeQueue<MetadataDeltaResponse> metadataResponseQueue; //
      thread safe queue for metadataDeltaResponseQueue
9     //constructor
10    MasterTree();
11    //constructor
12    MasterTree(const std::string &treeName, const std::string &
      treeDir);
13    //copy constructor
14    MasterTree(const MasterTree &obj);
15    //destructor
16    ~MasterTree();
17    //getter for trendTreeMap
18    std::unordered_map<std::string, TrendTree> &getTrendTreeMap();
19    //getter for trendTreeSettingsMap
20    std::unordered_map<std::string, TrendTreeSettings> &
      getTrendTreeSettingsMap();
21    //getter for lastSavedTimestamp
22    uint64_t getLastSavedTimestamp();
23    //initializes masterTree
24    bool initialize();
25    //writes treeMap
26    bool writeTreeMap();
27    //adds trendTree node to TREES node and links modelTree of
      trendTree to created node
28    bool addTrendTreeNode(const std::string treeToAddName);
29    //adds dpeList to dpeListQueue
30    bool addDpeListToDpeListQueue(const DpeList &dpeList);
31    //adds metadataResponse to metadataResponseQueue
32    bool addMetadataResponseToMetadataResponseQueue(const
      MetadataDeltaResponse &metadataResponse);
33
34  private :
35    std::unordered_map<std::string, TrendTree> trendTreeMap;
            //map which contains pairs (trendTree name, TrendTree)
      which map trend tree names to trendTree objects
36    uint64_t lastSavedTimestamp = 0;
            //most recently saved timestamp among most recently saved
      timestamps of trendTrees
37    std::unordered_map<std::string, TrendTreeSettings>
      trendTreeSettingsMap; //map of trend tree settings from settings
      file
38    //creates new model shot MDSplus tree
39    bool createModelShot();
40    //creates treeMap for master winccTree
41    int createTreeMap();
42    //reads tree names from TREE_MAP node of model tree of master
```

```
    wincc tree
43     bool readTreeList(std::vector<std::string> &treeNamesVec);
44     //reads settings from json file
45     bool readSettings();
46 };
```

List of Listings B.2: The MasterTree class

B.3 TrendTree

The *TrendTree* class, reported in Listing B.3, contains the code related to the TrendTree. Its salient characteristics are:

1. **dpeMetadataMap**: the map for data point metadata. It contains pairs (data point name, data point metadata) that map data point names to their metadata.
2. **dpeLastTimeMap**: the map for the last saved time for data points. It contains pairs (data point name, data point last saved time) that map node names to their last saved time.
3. **shotTree**: the ShotTree related to this TrendTree, used to save data point samples using relative experimental shot reference time.
4. **readDpeMap**: function which reads dpeMetadataMap from the MDSplus TrendTree file.
5. **writeDpeMap**: function which writes dpeMetadataMap to the MDSplus TrendTree file.
6. **addDpMetadataToTree**: function which adds new data point metatada to the MDSplus TrendTree file.
7. **deleteDpMetadataFromTree**: function which deletes data point metatada from the MDSplus TrendTree file.
8. **modifyDpMetadataInTree**: function which modifies data point metatada in the MDSplus TrendTree file.
9. **writeDpDataToCurrentShot**: function which writes data point sample (value and timestamp) to the current TrendTree shot.
10. **createNewShot**: function which creates a new TrendTree shot when a data point sample's timestamp is more recent than the current shot's end time.

```
1  //class for handling WinCC archive group with MDSplus tree
2  //archived datapoints are stored in tree nodes: stored info includes
       datapoint metadata and values
3  //datapoint samples are stored with absolute timestamp
4  class TrendTree : public DatapointTree
5  {
6  public:
7      //constructor
8      TrendTree(const std::string &treeName, const std::string &treeDir
       , const std::string &shotTreeName, const std::string &shotTreeDir)
       ;
9      //copy constructor
10     TrendTree(const TrendTree &obj);
11     //destructor
12     ~TrendTree();
13     //getter for archiveGroupData
14     DpArchiveGroupData &getArchiveGroupData();
15     //getter for dpeMetadataMap
16     std::unordered_map<std::string, DpMetadata> &getDpeMetadataMap();
17     //getter for dpeLastTimeMap
18     std::unordered_map<std::string, uint64_t> &getDpeLastTimeMap();
19     //getter for lastSavedTimestamp
20     uint64_t getLastSavedTimestamp();
21     //initializes trendTree
22     bool initialize();
23     //creates new model shot MDSplus tree
24     bool createModelShot(DpArchiveGroupData &dpArchiveGroupData);
25     //adds dp to current tree shot and to model shot (to keep it
       aligned with current shot)
26     bool addDpMetadataToTree(const DpIdentifierWithMetadata &dpIdent)
       ;
27     //removes dp from current tree shot and from model shot (to keep
       it aligned with current shot)
28     bool deleteDpMetadataFromTree(const DpIdentifierWithMetadata &
       dpIdent);
29     //modifies dp in current tree shot and in model shot (to keep it
       aligned with current shot)
30     bool modifyDpMetadataInTree(const DpIdentifierWithMetadata &
       dpIdent);
31     //writes DP sample (value and timestamp) to shot
32     bool writeDpDataToCurrentShot(const DpeList_Value &dpeListVal);
33     //gets number of shots related to queried time period to vector
34     //returns number of involved shots if no error, -1 if error
35     int getShotNumbersRelatedToTimePeriod(uint64_t querySartTime,
       uint64_t queryEndTime, std::vector<int> &involvedShotNumbersVec);
36
```

```
37 private:
38     DpArchiveGroupData archiveGroupData;                          //
    archive group data
39     int largestUsedIndex;                                         //
    stores largest used index for dpe indexing
40     uint64_t lastSavedTimestamp = 0;                              //
    most recently saved timestamp
41     std::unordered_map<std::string, uint64_t> dpeLastTimeMap;    //map
     for last saved time for dpes. It contains pairs (dpeName,
    lastTime) which map node names to their last save time
42     std::unordered_map<std::string, DpMetadata> dpeMetadataMap; //map
     for metadata of dpes. It contains pairs (dpeName, dp metadata)
    which map dp names to their metadata
43     static const uint64_t DEF_SHOT_DURATION = 24 * 3600 * 1000; //
    default shot duration
44     uint64_t shotDuration = DEF_SHOT_DURATION;                    //
    duration for shots of this tree
45     std::string shotTreeName;                                     //
    name of related shotTree
46     std::string shotTreeDir;                                      //dir
     of related shotTree
47     ShotTree shotTree;                                            //
    MDSplus shotTree for saving data to MDSplus shot pulse file
48
49     //get largest used index for nodes names
50     int getDpeMapLargestUsedIndex();
51     //reads archive group data from MDSplus tree
52     bool readArchiveGroupData();
53     //reads in Apd node from MDSplus tree to map
54     bool readDpeMap();
55     //writes dpeMap to MDSplus tree
56     bool writeDpeMap(MDSplus::Tree *tree);
57     //creates dpeLastTimeMap from dpeMap
58     bool createDpeLastTimeMap();
59     //gets metadata from tree node
60     template <typename dpMetadataFieldT>
61     bool getDpMetadataFromTreeNode(std::string nodePath,
    dpMetadataFieldT &dpMetadataField);
62     //creates dpeMetadataMap from dpeMap
63     bool createDpeMetadataMap();
64     //opens current shot
65     bool openCurrentShot();
66     //creates new shot based on shotInfo, sets current tree to point
    to it and adds shot info to shot info vector
67     bool createNewShot(uint64_t newStartTime);
68     //checks if provided time value belongs to current shot: returns
```

```
    negative value if it preceeds start time, zero if it belongs to
    shot, positive value if it exceeds end time
69      int checkShotTime(uint64_t currTime);
70      //tries to make current shot time consistent with dp timestamp
71      bool makeCurrShotDpTimeConsistent(const int64_t dpTime);
72  };
```

List of Listings B.3: The TrendTree class

B.4 ShotTree

The *ShotTree* class, reported in Listing B.4, contains the code related to the ShotTree. Its salient characteristics are:

1. **MDSplus::Event**: the class inherits from the MDSplus::Event class for receiving MDSplus events. These events inform the ShotTree when the next experimental pulse starts and ends, allowing it to know when datapoint samples must be saved using the relative time reference.
2. **updateDpMaps**: function called to keep the ShotTree's maps aligned with its parent TrendTree's ones.
3. **writeDpDataToShot**: function used to write data point sample to the MDSplus ShotTree file using relative experimental shot reference time.

```
1   //class for storing archived datapoints to MDSplus tree
2   //datapoints values are stored to this tree only when received
        MDSplus event allows it
3   //datapoints are stored in correspondece of experimental shot: sample
        timestamp is not absolute but relative shot time
4   class ShotTree: public DatapointTree, public MDSplus::Event
5   {
6   public:
7       //constructor
8       ShotTree(char *treeName, const std::string &treeDir, const std::
        string &trendTreeName, const std::string &trendTreeDir);
9       //destructor
10      ~ShotTree();
11      //updates datapoint maps of shotTree
12      bool updateDpMaps(std::unordered_map<std::string, std::string> &
        dpeMap, std::unordered_map<std::string, int> &dpeTimesNidMap, std
        ::unordered_map<std::string, int> &dpeValuesNidMap);
13      //initializes shot tree: model tree pointer, dpeMap,
        dpeValuesNidMap, dpeTimesNidMap
```

```
14      bool initialize();
15      //returns if sample timestamp belongs to MDSplus shot
16      bool isSampleInShot(uint64_t dpTimeStamp);
17      //writes data to shotTree
18      bool writeDpDataToShot(const DpeList_Value &dpeListVal, int64_t
        dpTime, const std::string &dpName);
19
20  private:
21      //runs event lister, called each time an event is received
22      void run();
23      //parses data string received with event to read message data
24      int parseDataString(const std::string &dataString);
25      //reacts to received init message
26      int reactToMSG_init();
27      //reacts to received start message
28      int reactToMSG_start(const std::vector<std::string> &stringVec);
29      //reacts to received init message
30      int reactToMSG_stop(const std::vector<std::string> &stringVec);
31      //reacts to received deinit message
32      int reactToMSG_deinit();
33      //reacts to received status message
34      int reactToMSG_status();
35      //reacts to received testUpdateDpMaps message
36      int reactToMSG_testUpdateDpMaps();
37      //gets shotTree ready for writing data to it
38      bool startTreeWriteData(int newShotNum, uint64_t newShotStartTime
        );
39      //stops writing data to shotTree
40      bool stopTreeWriteData(uint64_t newShotEndTime);
41      //deinitialized shotTree
42      bool deinitialize();
43      //gets shotTree status string
44      bool getStatusString(std::string &statusString);
45      //sets MDSplus event for shotTree status
46      bool setStatusMDSEvent(const std::string &statusString);
47      //creates nodeIdMap from dpeNameMap
48      bool createDpSegmentInfoMap(std::unordered_map<std::string,
        SegmentInfo> &dpeSegmentInfoMap, const std::unordered_map<std::
        string, std::string> &dpeNameMap);
49      //creates empty MDSplus Array matching dp value type for
        beginning node segment
50      MDSplus::Array *getSegmentEmptyMDSArray(const DpeList_Value &
        dpeListVal, const int nDims, const int segmentSize);
51      //creates data MDSplus Array matching dp value type for putting
        data to node segment
52      MDSplus::Array *getSegmentDataMDSArray(const DpeList_Value &
```

```
    dpeListVal, const int nDims, const int timesSize);
53  //creates new segment for dp data in values node of shot Tree
54  bool createNewDPValuesSegment(MDSplus::TreeNode *&
    nodeDPTimesShotTree, MDSplus::TreeNode *&nodeDPValuesShotTree,
    SegmentInfo &dpSegInfo, const DpeList_Value &dpeListVal, int nDims
    , int segmentSize);
55
56  //std::string shotTreePath = "";
     //path where pulse files of shotTree are to be created
57  std::string trendTreeDir = "";
    //directory of related trendTree
58  std::string trendTreeName = "";
    //name of releated trendTree
59  bool writeToShotEnabled = false;
    //flag to report if writing to shot pulse file is enabled
60  std::unordered_map<std::string, SegmentInfo> dpeSegmentInfoMap;
    //map for ids of value nodes. It contains pairs (dpeName, id of
    values node) which map node names to their value node IDs
61 };
```

List of Listings B.4: The ShotTree class

REFERENCES

[1] T. Fredian *et al.* MDSplus Extensions for Long Pulse Experiments. *Fusion Eng. Des.*, 83(1-3):317–320, April 2008. https://doi.org/10.1016/j.fusengdes.2007.08.022.

[2] *V. Authors.* World Energy Outlook 2023. Technical report, International Energy Agency, 2023.

[3] *V. Authors.* Energy Roadmap 2050. Technical report, European Commission, 2012.

[4] C. Bustreo *et al.* How Fusion Power Can Contribute to a Fully Decarbonized European Power Mix after 2050. *Fusion Eng. Des.*, 146:2189–2193, September 2019. https://doi.org/10.1016/j.fusengdes.2019.03.150.

[5] J. Ongena. Fusion: a True Challenge for an Enormous Reward. *EPJ Web Conf.*, 189:00015, 2018. https://doi.org/10.1051/epjconf/201818900015.

[6] Jeffrey P. Freidberg. *Plasma Physics and Fusion Energy.* Cambridge University Press, 2007.

[7] O. A. Hurricane *et al.* Physics Principles of Inertial Confinement Fusion and U.S. Program Overview. *Rev. Mod. Phys.*, 95:025005, June 2023. https://doi.org/10.1103/RevModPhys.95.025005.

[8] H. Zohm *et al.* Studies of Edge Localized Modes on ASDEX. *Nuclear Fusion*, 32(3):489, March 1992. https://doi.org/10.1088/0029-5515/32/3/I11.

[9] K. H. Burrell *et al.* Physics of the L-mode to H-mode Transition in Tokamaks. *Plasma Physics and Controlled Fusion*, 34(13):1859, December 1992. https://doi.org/10.1088/0741-3335/34/13/014.

[10] ITER official website, 2024. URL http://www.iter.org/. Accessed on 06 January 2024.

[11] A.C. Riviere. Penetration of Fast Hydrogen Atoms into a Fusion Reactor Plasma. *Nuclear Fusion*, 11(4):363, August 1971. https://doi.org/10.1088/0029-5515/11/4/006.

[12] *V. Authors. Fusion Physics.* Non-serial Publications. INTERNATIONAL ATOMIC ENERGY AGENCY, Vienna, 2012. ISBN 978-92-0-130410-0. URL https://www.iaea.org/publications/8879/fusion-physics.

[13] J. Maglica. Plasma Heating with Neutral Beam Injection. University of Ljubljana, 2005. Seminar.

[14] V. Toigo *et al.* A Substantial Step Forward in the Realization of the ITER HNB System: The ITER NBI Test Facility. *Fusion Eng. Des.*, 123:32–39, 2017. ISSN 0920-3796. https://doi.org/10.1016/j.fusengdes.2016.11.007. Proceedings of the 29th Symposium on Fusion Technology (SOFT-29) Prague, Czech Republic, September 5-9, 2016.

[15] V. Toigo *et al.* The ITER Neutral Beam Test Facility towards SPIDER Operation. *Nuclear Fusion*, 57(8):086027, July 2017. https://doi.org/10.1088/1741-4326/aa7490.

[16] V. Toigo *et al.* The PRIMA Test Facility: SPIDER and MITICA Test-beds for ITER Neutral Beam Injectors. *New Journal of Physics*, 19(8):085004, August 2017. https://doi.org/10.1088/1367-2630/aa78e8.

[17] M. J. Singh *et al.* Heating Neutral Beams for ITER: Negative Ion Sources to Tune Fusion Plasmas. *New Journal of Physics*, 19(5):055004, May 2017. https://doi.org/10.1088/1367-2630/aa639d.

[18] S. Ortolani and D. D. Schnack. *Magnetohydrodynamics of Plasma Relaxation.* World Scientific, 1993.

[19] Taylor and J. Brian. Relaxation of Toroidal Plasma and Generation of Reverse Magnetic Fields. *Physical Review Letters*, 33(19):1139, 1974. https://doi.org/10.1103/PHYSREVLETT.33.1139.

[20] V. Antoni *et al.* MHD Stability Analysis of Force-free Reversed Field Pinch Configurations. *Nuclear Fusion*, 26(12):1711, December 1986. https://doi.org/10.1088/0029-5515/26/12/012.

[21] S. Cappello and D. Escande. Bifurcation in Viscoresistive MHD: the Hartmann Number and the Reversed Field Pinch. *Physical Review Letters*, 85 18:3838–41, 2000. https://doi.org/10.1103/PHYSREVLETT.85.3838.

[22] P. R. Brunsell *et al.* Coherent Magnetic Field Fluctuations and Locked Modes in a Reversed-field Pinch. *Physics of fluids. B, Plasma physics*, 5:885–895, 1993. https://doi.org/10.1063/1.860939.

[23] Susanna Cappello *et al.* Stationary Quasi Single Helicity States in RFX. In *26th EPS Conf. on Controlled Fusion and Plasma Physics*, volume 23J, pages 1149–1152, 1999. URL http://ocs.ciemat.es/EPS1999/web/index.htm.

[24] P. Martin. Magnetic and Thermal Relaxation in the Reversed Field Pinch. *Plasma Physics and Controlled Fusion*, 41(3A):A247, March 1999. https://doi.org/10.1088/0741-3335/41/3A/018.

[25] D. Escande *et al.* Quasi-single-helicity Reversed-field-pinch Plasmas. *Physical Review Letters*, 85 8:1662–5, 2000. https://doi.org/10.1103/PHYSREVLETT.85.1662.

[26] R. Lorenzini *et al.* Self-organized Helical Equilibria as a New Paradigm for Ohmically Heated Fusion Plasmas. *Nature Physics*, 5(8):570–574, August 2009. https://doi.org/10.1038/nphys1308.

[27] P. Piovesan *et al.* RFX-mod: A Multi-configuration Fusion Facility for Three-dimensional Physics Studies. *Physics of Plasmas*, 20:056112, 2013. https://doi.org/10.1063/1.4806765.

[28] P. Zanca *et al.* Beyond the Intelligent Shell Concept: the Clean-mode-control. *Nuclear Fusion*, 47(11):1425, October 2007. https://doi.org/10.1088/0029-5515/47/11/004.

[29] L. Marrelli *et al.* Magnetic Self Organization, MHD Active Control and Confinement in RFX-mod. *Plasma Physics and Controlled Fusion*, 49(12B):B359, November 2007. https://doi.org/10.1088/0741-3335/49/12B/S33.

[30] G. Bateman and Ray C. Grimm. MHD Instabilities. *Physics Today*, 32(10):61–62, October 1979. ISSN 0031-9228. https://doi.org/10.1063/1.2995243.

[31] J. P. Freidberg. *Ideal MHD*. Cambridge University Press, 2014. https://doi.org/10.1017/CBO9780511795046.

[32] B. B. Kadomtsev. *Tokamak Plasma: a Complex Physical System*. Plasma physics series. Taylor & Francis, 1992. ISBN 9780750302340.

[33] H. K. Moffatt. Nonlinear Magnetohydrodynamics. By D. Biskamp. Cambridge University Press, 1993, 378 pp. *Journal of Fluid Mechanics*, 263:375–376, March 1994. https://doi.org/10.1017/S0022112094214167.

[34] S. Ortolani. Active MHD Control Experiments in RFX-mod. *Plasma Physics and Controlled Fusion*, 48, December 2006. https://doi.org/10.1088/0741-3335/48/12B/S34.

[35] P. Sonato *et al.* Machine Modification for Active MHD Control in RFX. *Fusion Eng. Des.*, 66-68:161–168, 2003. ISSN 0920-3796. https://doi.org/10.1016/S0920-3796(03)00177-7. 22nd Symposium on Fusion Technology.

[36] M. Zuin *et al.* Overview of the RFX-mod Fusion Science Activity. *Nuclear Fusion*, 57(10):102012, June 2017. https://doi.org/10.1088/1741-4326/aa61cc.

[37] M. Veranda *et al.* Magnetohydrodynamics Modelling Successfully Predicts New Helical States in Reversed-field Pinch Fusion Plasmas. *Nuclear Fusion*, 57(11):116029, August 2017. https://doi.org/10.1088/1741-4326/aa7f46.

[38] P. Zanca *et al.* Feedback Control Model of the m = 2, n = 1 Resistive Wall Mode in a Circular Plasma. *Plasma Physics and Controlled Fusion*, 54(9): 094004, August 2012. https://doi.org/10.1088/0741-3335/54/9/094004.

[39] M. Spolaore *et al.* H-mode Achievement and Edge Features in RFX-mod Tokamak Operation. *Nuclear Fusion*, 57(11):116039, August 2017. https://doi.org/10.1088/1741-4326/aa7f1e.

[40] P. Zanca *et al.* Advanced Feedback Control of Magnetohydrodynamic Instabilities: Comparison of Compensation Techniques for Radial Sensors. *Plasma Physics and Controlled Fusion*, 54(12):124018, November 2012. https://doi.org/10.1088/0741-3335/54/12/124018.

[41] L. Marrelli *et al.* Upgrades of the RFX-mod Reversed Field Pinch and Expected Scenario Improvements. *Nuclear Fusion*, 59(7):076027, June 2019. https://doi.org/10.1088/1741-4326/ab1c6a.

[42] Gianmaria de Tommasi *et al.* Plasma Magnetic Control in Tokamak Devices. *Journal of Fusion Energy*, 38:406–436, 2019. https://doi.org/10.1007/s10894-018-0162-5.

[43] A. Pironti M. Ariola. *Magnetic Control of Tokamak Plasmas*. Advances in Industrial Control. Springer London, 1992. ISBN 978-1-84996-783-9. https://doi.org/10.1007/978-1-84800-324-8.

[44] L. Pigatto *et al.* Control System Optimization Techniques for Real-Time Applications in Fusion Plasmas: The RFX-mod Experience. *IEEE Transactions on Nuclear Science*, 64(6):1420–1425, 2017. https://doi.org/10.1109/TNS.2017.2695372.

[45] F. Janky *et al.* Plasma Density Control in Real-time on the COMPASS Tokamak. *Fusion Eng. Des.*, 96-97:637–640, 2015. ISSN 0920-3796. https://doi.org/10.1016/j.fusengdes.2015.04.065. Proceedings of the 28th Symposium On Fusion Technology (SOFT-28).

[46] G. Manduchi *et al.* The New Feedback Control System of RFX-mod Based on the MARTe Real-Time Framework. *IEEE Trans. Nucl. Sci.*, 61(3):1219–1221, June 2014. https://doi.org/10.1109/TNS.2014.2321185.

[47] G. De Tommasi *et al.* Plasma Position and Current Control System Enhancements for the JET ITER-like Wall. *Fusion Eng. Des.*, 89(3):233–242, 2014. ISSN 0920-3796. https://doi.org/10.1016/j.fusengdes.2013.06.010. Design and implementation of real-time systems for magnetic confined fusion devices.

[48] MARTe2 Real-Time Framework. https://vcis.f4e.europa.eu/marte2-docs/master/html/overview.html, 2022. Accessed on 06 January 2024.

[49] MDSplus Data System. https://mdsplus.org/, 2022. Accessed on 06 January 2024.

[50] Experimental Physics and Industrial Control System. https://epics.anl.gov/, 2023. Accessed on: 06 January 2024.

[51] LabVIEW by National Instruments. https://www.ni.com/it-it/shop/product/labview.html, 2024. Accessed on 06 January 2024.

[52] SIMATIC WinCC Open Architecture. https://www.winccoa.com/, 2024. Accessed on 06 January 2024.

[53] Wonderware by AVEVA. https://www.aveva.com/en/solutions/operations/wonderware/, 2024. Accessed on 06 January 2024.

[54] TANGO Controls. https://www.tango-controls.org/, 2024. Accessed on 06 January 2024.

[55] VxWorks. https://www.windriver.com/products/vxworks, 2024. Accessed on 06 January 2024.

[56] RTEMS. https://www.rtems.org/, 2024. Accessed on 06 January 2024.

[57] EPICS Channel Access. https://epics.anl.gov/docs/ca.php, 2018. Accessed 06 January 2024.

[58] EPICS pvAccess. https://epics-controls.org/resources-and-support/documents/pvaccess/, 2024. Accessed on 06 January 2024.

[59] OPC Unified Architecture. https://opcfoundation.org/about/opc-technologies/opc-ua/, 2024. Accessed on 06 January 2024.

[60] R. Lange. CODAC Core System Overview. https://www.iter.org/mach/codac/coresystem, 2022. Accessed 06 January 2024.

[61] V. Toigo *et al.* Progress in the ITER Neutral Beam Test Facility. *Nuclear Fusion*, 59:086058, 2019. https://doi.org/10.1088/1741-4326/ab2271.

[62] G. Serianni *et al.* First Operation in SPIDER and the Path to Complete MITICA. *Review of Scientific Instruments*, 91:023510, 2020. https://doi.org/10.1063/1.5133076.

[63] V. Toigo *et al.* On the Road to ITER NBIs: SPIDER Improvement After First Operation and MITICA Construction Progress. *Fusion Eng. Des.*, 168: 112622, 2021. https://doi.org/10.1016/j.fusengdes.2021.112622.

[64] A. Luchetta *et al.* Control and Data Acquisition of the ITER Full-scale Ion Source for the Neutral Beam Test Facility. *Fusion Eng. Des.*, 96-97:5, 2015. https://doi.org/10.1016/j.fusengdes.2015.06.001.

[65] A. Luchetta *et al.* Integrating Supervision, Control and Data Acquisition-The ITER Neutral Beam Test Facility Experience. *Fusion Eng. Des.*, 112:928–931, 2016. https://doi.org/10.1016/j.fusengdes.2016.05.012.

[66] A. Luchetta *et al.* Implementation of the SPIDER Central Interlock. *Fusion Eng. Des.*, 123:990–994, 2017. https://doi.org/10.1016/j.fusengdes.2017.02.075.

[67] S. Dal Bello *et al.* Safety Systems in the ITER Neutral Beam Test Facility. *Fusion Eng. Des.*, 146:246–249, 2019. https://doi.org/10.1016/j.fusengdes.2018.12.037.

[68] A. Luchetta *et al.* As-built Design, Commissioning and Integration of the SPIDER and NBTF Central Safety Systems. *Fusion Eng. Des.*, 190:113536, 2023. https://doi.org/10.1016/j.fusengdes.2023.113536.

[69] R. S. Hemsworth and A. Tanga. Status of the ITER Neutral Beam Injection System. *Review of Scientific Instruments*, 79:02C109, 2008. https://doi.org/10.1063/1.2814248.

[70] B. Heinemann *et al.* Upgrade of the BATMAN Test Facility for H- Source Development. In *AIP Conference Proceedings*, volume 1655, page 060003, 2015. https://doi.org/10.1063/1.4916472.

[71] M. Hanada *et al.* Progress in Development and Design of the Neutral Beam Injector for JT-60SA. *Fusion Eng. Des.*, 86:835–838, 2011. https://doi.org/10.1016/j.fusengdes.2011.04.068.

[72] Y. Takeiri *et al.* High Performance of Neutral Beam Injectors for Extension of LHD Operational Regime. *Fusion Science and Technology*, 58, 2010. https://doi.org/10.13182/FST10-A10834.

[73] J.-Y. Journeaux. *Plant Control Design Handbook*, April 2013. URL https://www.iter.org/doc/www/content/com/Lists/ITER%20Technical%20Reports/Attachments/15/ITR_20_009_Plant_Control_Design_Handbook_v1.pdf.

[74] HPE Aruba. 8200, 5400, and 2530 Switch Families. https://www.hpe.com/, 2023. Accessed on 06 January 2024.

[75] Mainberg. Grand Master Clock LANTIME M600/GPS/PTPv2. https://www.meinberg-usa.com/products/ptp-ieee-1588/ptp-v2-network-time-server-m600-gps-ptpv2.htm, 2023. Accessed on: 06 January 2024.

[76] INCAA Computers. CP-DIO4-5024 Timing Generator Data Sheet. https://www.incaacomputers.com/, 2022. Accessed on 06 January 2024.

[77] National Instruments. PXI 8866 Synchronization Module. https://www.ni.com/docs/en-US/bundle/pxi-6683-specs/page/specs.html, 2023. Accessed on 06 January 2024.

[78] IEEE. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pages 1–269, 2008. https://doi.org/10.1109/IEEESTD.2008.4579760.

[79] IEEE. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)*, pages 1–499, 2020. https://doi.org/10.1109/IEEESTD.2020.9120376.

[80] Red Hat Enterprise Linux. https://www.redhat.com/, 2022. Accessed 06 January 2024.

[81] G. Manduchi *et al.* A Portable Control and Data Acquisition Solution Using EPICS, MARTe and MDSplus. *Fusion Eng. Des.*, 127:50–53, 2018. https://doi.org/10.1016/j.fusengdes.2017.12.012.

[82] G. Manduchi *et al.* MARTe2 and MDSplus Integration for a Comprehensive Fast Control and Data Acquisition System. *Fusion Eng. Des.*, 191:111982, 2020. https://doi.org/10.1016/j.fusengdes.2020.111892.

[83] G. Manduchi *et al.* New EPICS Channel Archiver Based on MDSplus Data System. *IEEE Transactions on Nuclear Science*, 58(6):3158–3161, 2011. https://doi.org/10.1109/TNS.2011.2167349.

[84] PXIe 6368. https://www.ni.com/docs/en-US/bundle/pxie-6368-specs/page/specs.html, 2023. Accessed on 06 January 2024.

[85] PXIe 6259. https://www.ni.com/docs/en-US/bundle/pci-pcie-pxi-pxie-usb-6259-specs/page/specs.html, 2023. Accessed on 06 January 2024.

[86] Generic Interface for Cameras Standard. https://www.emva.org/standards-technology/genicam, 2023. Accessed on 06 January 2024.

[87] GigE Vision Standard. https://www.automate.org/a3-content/vision-standards-gige-vision, 2023. Accessed on 06 January 2024.

[88] G. Manduchi *et al.* CODAS for Long Lasting Experiments: The SPIDER Experience. In *32nd SOFT*, Dubrovnik, Croatia, 2022.

[89] Grafana. https://grafana.com/, 2023. Accessed on 06 January 2024.

[90] C. Taliercio *et al.* Distributed Continuous Event - Based Data Acquisition Using IEEE 1588 Synchronization and FlexRIO FPGA. *IEEE Transactions on Nuclear Science*, 64:1970–1974, 2017. https://doi.org/10.1109/TNS.2017.2713101.

[91] G. Liu, P. Makijarvi, and N. Pons. The ITER CODAC Network Design. *Fusion Eng. Des.*, 130:6–10, 2018. https://doi.org/10.1016/j.fusengdes.2018.02.072.

[92] G. Manduchi *et al.* The Timing System of the Iter Full Size Neutral Beam Injector Prototype. This conference, 2019.

[93] Open Source BSD License. http://www.linfo.org/bsdlicense.html, 2019. Accessed on 06 January 2024.

[94] European Union Public License - EUPL v.1.1. http://ec.europa.eu/idabc/eupl.html, 2019. Accessed on 06 January 2024.

[95] O. Barana *et al.* Data Acquisition Upgrade in the RFX Experiment. *Fusion Eng. Des.*, 74(1-4):841–845, November 2009. https://doi.org/10.1016/j.fusengdes.2005.06.250.

[96] O. Barana *et al.* Real-time Control Environment for the RFX Experiment. *Fusion Eng. Des.*, 74(1-4):617–621, November 2005. https://doi.org/10.1016/j.fusengdes.2005.06.146.

[97] A. C. Neto *et al.* MARTe: A Multiplatform Real-Time Framework. *IEEE Trans. Nucl. Sci.*, 57(2):479–486, April 2010. https://doi.org/10.1109/TNS.2009.2037815.

[98] N. Marconato *et al.* Accurate Magnetic Sensor System Integrated Design. *Sensors*, 20(10):2929, May 2020. https://doi.org/10.3390/s20102929.

[99] A. J. N. Batista *et al.* ATCA Control System Hardware for the Plasma Vertical Stabilization in the JET Tokamak. In *16th IEEE-NPSS Real Time Conference*, pages 308–313, 2009. https://doi.org/10.1109/RTC.2009.5321967.

[100] *PicoZed Board*, 2023. URL https://www.avnet.com/wps/portal/us/products/avnet-boards/avnet-board-families/picozed/.

[101] G. Manduchi *et al.* MARTe2 and MDSplus Integration for a Comprehensive Fast Control and Data Acquisition System. *Fusion Eng. Des.*, 161:111892, December 2020. https://doi.org/10.1016/j.fusengdes.2020.111892.

[102] O. Barana *et al.* A General-purpose Java tool for Action Dispatching and Supervision in Nuclear Fusion Experiments. *IEEE Trans. Nucl. Sci.*, 49(2): 469–473, April 2002. https://doi.org/10.1109/TNS.2002.1003777.

[103] A. Luchetta *et al.* Real-time Communication for Distributed Plasma Control Systems. *Fusion Eng. Des.*, 83(2):520–524, 2008. ISSN 0920-3796. https://doi.org/10.1016/j.fusengdes.2007.12.026. Proceedings of the 6th IAEA Technical Meeting on Control, Data Acquisition, and Remote Participation for Fusion Research.

[104] S. Vitturi, O. N. Hemming. *CT/131-B Sistema di Gestione dei Messaggi per Controllori Programmabili: Manuale Utente - Versione 2.2.* Istituto Gas Ionizzati - Progetto RFX - Consiglio Nazionale delle Ricerche - Associazione Euratom-ENEA-CNR, Corso Stati Uniti, 4 - 35020, Padova (ITALY), February 1990.

[105] A. C. Neto *et al.* From Use Cases of the Joint European Torus Towards Integrated Commissioning Requirements of the ITER Tokamak. *Fusion Eng. Des.*, 96-97:672–675, October 2015. https://doi.org/10.1016/j.fusengdes.2015.02.024.

[106] A. Wallander and B. Bauvir. ITER Controls Approaching One Million Integrated EPICS Process Variables. In *Proc. 19th Int. Conf. Accel. Large Exp. Phys. Control Syst. (ICALEPCS'23)*, number 19 in International Conference on Accelerator and Large Experimental Physics Control Systems, pages 6–11. JACoW Publishing, Geneva, Switzerland, 02 2024. ISBN 978-3-95450-238-7. https://doi.org/10.18429/JACoW-ICALEPCS2023-MO1BCO02.

[107] R. Castro *et al.* Soft Real-time EPICS Extensions for Fast Control: A Case Study Applied to a TCV Equilibrium Algorithm. *Fusion Eng. Des.*, 89(5):638–643, 2014. ISSN 0920-3796. https://doi.org/https://doi.org/10.1016/j.fusengdes.2014.03.044. Proceedings of the 9th IAEA Technical Meeting on Control, Data Acquisition, and Remote Participation for Fusion Research.

[108] A. Barbalace *et al.* Performance Comparison of EPICS IOC and MARTe in a Hard Real-Time Control Application. In *2010 17th IEEE-NPSS Real Time Conference*, pages 1–5, 2010. https://doi.org/10.1109/RTC.2010.5750376.

[109] O. Barana *et al.* Comparison Between Commercial and Open-source SCADA Packages - A Case Study. *Fusion Eng. Des.*, 85(3-4):491–495, July 2010. https://doi.org/10.1016/j.fusengdes.2010.02.004.