



MURDOCH
UNIVERSITY

PERTH, WESTERN AUSTRALIA

ENG460: Engineering Thesis

Honeywell Experion System for Teaching Purposes

A report submitted to the School of Engineering and Energy, Murdoch University in partial fulfilment of the requirements for the degree of Bachelor of Engineering

Edwin Hug Hou Lum

30637469

2011

Supervisor:
Associate Professor Graeme R. Cole

Abstract

The Experion Process Knowledge System developed by Honeywell is an engineering industrial tool that Murdoch University has acquired so that enrolled undergraduate students studying electrical engineering are given the opportunity to understand its operation.

This document will provide a summary of the works that were accomplished as part of the goal to develop and implement a fully operational Experion system into the Industrial Computer Systems Engineering Facility at Murdoch University.

Due to the complex nature of the Experion software platform, only the most important aspects that the project explored will be covered in this thesis report. Much of the background which is required to be able to understand the Experion package is explained in addition to covering the tasks achieved. Detailed technical user guides are also included to assist future students who intend to continue the development and expansion of an Experion system.

As this thesis topic has no prior involvement in any other subjects, it will be the first to explore and understand the Experion platform purely for simulation purposes. The Experion Process Knowledge System platform developed by Honeywell is considered to be an industry leading control tool. Therefore, by having Experion system operational in the Industrial Computer Systems Engineering Facility so that it may be utilised as a teaching and learning tool can only be a benefit to the many future undergraduate electrical engineering students.

Acknowledgements

This thesis project could not have reached its current stage without the continued assistance, commitment and support of the following people:

Associate Professor Graeme R. Cole

Mr William Stirling

This page is intentionally left blank.

Table of Contents

| | |
|---|------|
| Abstract..... | i |
| Acknowledgements..... | ii |
| List of Figures | vi |
| List of Tables | vii |
| Acronym Definitions | viii |
| Chapter One: Introduction..... | 1 |
| 1.1 Background..... | 1 |
| 1.2 Statement of Problem | 1 |
| 1.3 Objectives | 2 |
| Chapter Two: Technical Reviews | 4 |
| 2.1 Experion PKS & Network Architecture | 4 |
| 2.2 Honeywell C300 Controller & Control Execution Environment | 7 |
| 2.3 Configuration Studio..... | 9 |
| 2.4 Control Builder | 12 |
| 2.5 Enterprise Model Builder | 14 |
| 2.6 HMIWeb Display Builder | 15 |
| 2.7 Quick Builder | 15 |
| 2.8 Station | 15 |
| 2.9 System Displays | 17 |
| Chapter Three: Accomplishments..... | 18 |
| 3.1 Experion PKS Installation & Setup | 18 |
| 3.2 SIM-C300 Configuration | 20 |
| 3.3 Control Execution Environment | 23 |
| 3.4 PID Function Block..... | 24 |
| 3.4.1 Equation A..... | 28 |
| 3.4.2 Equation B..... | 29 |
| 3.4.3 Equation C..... | 29 |
| 3.4.4 Equation D..... | 29 |
| 3.5 Microsoft Excel Data Exchange | 29 |
| 3.6 A Simulated System | 30 |
| Chapter Four: Future Projects & Tasks | 33 |
| 4.1 SIM-C300 | 33 |
| 4.2 Implementation into ISCE Facility..... | 33 |
| Chapter Five: Conclusion | 34 |

| | |
|--------------------|----|
| Bibliography | 35 |
| Appendix A | 37 |
| Appendix B | 38 |
| Appendix C | 67 |
| Appendix D | 82 |
| Appendix E | 89 |

List of Figures

| | |
|--|----|
| Figure 1: Basic Architecture of an Experion System | 6 |
| Figure 2: Configuration Studio | 10 |
| Figure 3: Connect Dialog Window | 11 |
| Figure 4: Control Builder | 13 |
| Figure 5: experion1 System Server | 19 |
| Figure 6: Pilot Plant System Server | 20 |
| Figure 7: Errors Detected | 21 |
| Figure 8: PID Function Block | 24 |
| Figure 9: Server Display Showing Manual & Automatic Modes | 26 |
| Figure 10: Server Display Showing Operator & Program Mode Attributes | 27 |
| Figure 11: Set Periodic Recalculation Interval Window | 30 |
| Figure 12: Simulated System Using MEDE | 31 |
| Figure 13: Server Display Showing a Simulated System | 32 |

List of Tables

| | |
|---|------|
| Table 1: Table of Acronym Definitions..... | viii |
| Table 2: Acronyms used in the PID Equations | 28 |
| Table 3: Equation Notations | 31 |

Acronym Definitions

| Table of Acronym Definitions | |
|------------------------------|--|
| ASM | Abnormal Situation Management |
| CEE | Control Execution Environment |
| CM | Control Module |
| CPM | Central Processing Module |
| DCS | Distributed Control System |
| FB | Function Block |
| HMI | Human Machine Interface |
| ICSE | Industrial Computer Systems Engineering |
| IO | Input & Output |
| IOM | Input & Output Module |
| MEDE | Microsoft Excel Data Exchange |
| MV | Manipulated Variable |
| OP | Control Output |
| PID | Proportional Integral Derivative |
| PLC | Programmable Logic Controller |
| PKS | Process Knowledge System |
| PV | Process Variable |
| REGCTL | Regulatory Control |
| RTU | Remote Terminal Unit |
| SCADA | Supervisory Control And Data Acquisition |
| SCM | Sequential Control Modules |
| SIM-C300 | Simulated Honeywell C300 Controller |
| SP | Set Point |

Table 1: Table of Acronym Definitions

Chapter One: Introduction

This first chapter of document will introduce the topic at hand along with providing an outline of what is hoped to be achieved. Details on what challenges lie in wake for the Honeywell Experion System for Teaching Purposes thesis project will also be provided.

1.1 Background

In its effort to continually provide a high quality environment for learning, Murdoch University's School of Engineering and Energy made a decision to develop, integrate and implement a new control system in the form of Honeywell's Experion Process Knowledge System (PKS) into the Industrial Computer Systems Engineering (ICSE) Facility.

Honeywell's Experion PKS is an industry leading control infrastructure platform system and one which is at the forefront of its type of technology. It is therefore natural for Murdoch University to want a learning engine that will help teach undergraduate electrical engineering students how to use industry standard tools, which in this case is Honeywell's Experion PKS.

1.2 Statement of Problem

The Experion PKS developed by Honeywell is a high performance software platform that is used extensively as an engineering control tool. This breakthrough software package allows the configuration as well as integration of both Distributed Control System (DCS) and Supervisory Control and Data Acquisition (SCADA) topologies. The Experion platform was successfully launched in the year 2003 and is known to be an effective tool in the engineering industry. As such, it is important for students that are pursuing a career in electrical engineering to be somewhat familiar with this software.

Seeing that there has been no prior development of the Experion platform for the ICSE Facility, this project will be the first to do so for simulation purposes. Honeywell's Experion PKS must be fully compatible and operational with the facility's current instruments by the completion of the project. This is another step that Murdoch University has taken to show its continued investment into the engineering program and its students as well as the importance that is placed on industry knowledge.

Since Honeywell's Experion PKS is still a fairly new software package at Murdoch University, a large degree of research and autonomous learning must be accomplished to ensure its proper deployment in the ICSE Facility. The only location at Murdoch University where Honeywell's

Experion PKS has been successfully implemented is at the Pilot Plant Facility which was completed as part of previous thesis topics. One Key difference between this implementation and the previous implementation at the Pilot Plant Facility besides the location is the fact that the controller needed for the Experion software to operate will be simulated.

1.3 Objectives

Detailed in this section of the report are the many primary objectives that is hoped to be achieved before this thesis project concludes. It has been stated earlier that this project revolves heavily around Honeywell's Experion PKS platform. There is the intention of implementing this control software in the ICSE Facility at Murdoch University once it is deemed operational so it may serve as a teaching tool for enrolled electrical engineering students. Therefore in these circumstances, the majority of the tasks aim to firstly establish an operational Experion system for experimental and testing purposes.

Honeywell's Experion PKS is an exceptionally complex and extensive piece of software to configure owing to the fact that it was developed with the ability to be implemented into almost any process situation requiring control in mind. Along with the software being a fairly recent acquisition for Murdoch University, there are a host of challenges to overcome. Experience in the operation of the software is limited and thus, most of the knowledge required will have to be sourced from Honeywell documentation. Therefore, an objective will be to review said documents from Honeywell.

There are many legitimate reasons for wanting to experiment with the software before an actual implementation occurs. The key motive for doing so however is to ensure that the commissioning and testing process will not cause any downtime to the facility itself due to the multiple uses that the ICSE Facility functions as. Therefore a separate server and a single client machine will first be used to set up Honeywell's Experion PKS software platform. If successful, it would just be a matter of relocating the server to the facility and loading the appropriate software onto the future workstations.

For this to occur, the required software, Honeywell's Experion PKS with server applications must be installed onto an actual server computer. A copy of the Experion software suite with client applications will have to be installed on any associated workstation machines. In this case, there will be only one such workstation.

Once an empty shell of Honeywell's Experion PKS has been successfully installed onto the relevant machines, the next task to follow on from this point will be the deployment, testing and integration of a simulated Honeywell C300 controller, otherwise known as a SIM-C300, into the control system along with any of its extra associated components required.

The development of a program that a SIM-C300 that will use the Control Execution Environment (CEE) software provided will then be considered as it is through the CEE where Honeywell's Experion PKS is configured for a system and thus, employed as a teaching tool. This will also serve as an experiment to confirm its basic functionalities and thus allow for further development down the track.

Due to the time constraints that are placed upon this thesis project in conjunction with the topic's vastness, these objectives are just the tip of the iceberg to put it in layman's terms. There are a number of future project prospects that this thesis project leads into and provides a basis for, all of which will be discussed further down the track.

Chapter Two: Technical Reviews

Detailed in this chapter of the report is much of the background knowledge required to understand what is trying to be accomplished in this thesis project. Explored are the many different and unique tools that a thesis of this scale uses along with detailed specific definitions.

2.1 Experion PKS & Network Architecture

Successfully launched in the year 2003, Honeywell's Experion PKS has proved to be an effective tool in the engineering industry as high performance software platform that allows for the configuration as well as integration of both DCS and SCADA topologies (Honeywell International Inc., No Date Given).

Honeywell's Experion PKS mainly consists of multiple development tools and end-user environments. As it is one large integrated software suite, the Experion PKS platform therefore allows for the construction and configuration of unique control systems. Experion is equipped to be able to handle systems based on DCS, SCADA or an integration of both these topologies in every aspect possible (Honeywell International Inc., No Date Given).

DCS and SCADA are control system topologies each with its own advantages and disadvantages. Since the introduction of Programmable Logic Controllers (PLCs) in addition to computer based control systems across a wide range of industries, DCS and SCADA have emerged to be the two most popular of such technology. The advent of these technologies have seen an exponential increase in the complexity and versatility of such control systems where in today's world, just one control system is required as it has the capability to operate an entire system (Bailey & Wright, 2003). Honeywell's Experion PKS is such a system where all the advantages of DCS and SCADA are merged into one.

A SCADA system consists of a number of Remote Terminal Units (RTUs) collecting field data which is then connected back to a master station via a communications link. The master station is therefore able to display the acquired data which is usually in real-time and as accurate as possible (Boyer, 1999). As such, the SCADA topology operates in a master slave hierarchy and does not take the advantages of peer to peer communications (Clarke, Reynders, & Wright, 2004). In addition to collecting vital information in real-time, a master station also allows for an operator to perform remote control tasks which assists in optimising the operations of the system (Boyer, 1999). By operating a successful automated computer-

based control system consequently results in many benefits including a more efficient, reliable and safer operation as well as lower operational costs compared to a non-automated system. A well implemented and successful SCADA system depends on multiple factors. Utilising proven and reliable technology is one. With the advances in technology today, hardware reliability is no longer such a problem. Software is where new challenges are produced and is usually where problems present themselves. Issues usually arise from the increasing complexity of the software aspect where all personnel involved in the operation of the system have not received adequate and comprehensive training in its use (Bailey & Wright, 2003).

In a DCS, the data acquisition and control functionalities are performed by multiple distributed microprocessor-based units that are situated near to the devices being controlled or the instrument from which data is being gathered. Unlike a SCADA system, DCSs do employ the use of peer to peer communication. This is due to the fact that the autonomous control strategies used by a DCS is based on Input and Output (IO) signals and information acquired by other process controllers on the peer to peer network (Bailey & Wright, 2003). The control actions are based on control schemes that have been earlier configured on a supervisory level and downloaded to the controller.

A control system based on DCS architecture is able to transmit diagnostic information and data. Unless additional programming and configuration is performed, such features commonly go unsupported on a system which is based on SCADA (Bailey & Wright, 2003).

The software application used across an entire system which has been implemented with a computer-based control of DCS design is the same. Whether it is the integrated set of interfaces to allow for an easy system configuration to the operator control screens or developing control strategies, the software remains unchanged because in most cases the physical components are provided by only one manufacturer due to compatibility issues (Bailey & Wright, 2003).

Conversely, SCADA systems allows for a host of multiple different software environments to be employed since its control scheme architecture often utilises physical components and instruments from several manufacturers. This however, limits configuration to local parameters only (Clarke, Reynders, & Wright, 2004).

SCADA systems are more suited and ideal for systems whose operations are over vast remote areas. This is because it is able to utilise wireless communications between devices and instruments. A DCS however, is vastly more suited for a detailed real-time control in processing applications involving highly interactive systems (Bailey & Wright, 2003).

Experion PKS is very much a complex software platform and to understand its every function available would be immensely difficult. The main foundation of Experion PKS is built around seven core software environment components; Configuration Studio, Control Builder, Enterprise Model Builder, HMIWeb Display Builder, Quick Builder, Station and the System Displays.

An Experion system's basic architecture can be separated into three components; the physical system itself (such as Murdoch University's Pilot Plant Facility), an Experion ready server machine, and workstations loaded with Experion client applications. In a very general sense, the Experion system operates in the following manner. User commands are sent through client workstation machines which are linked through an Experion server to the associated instrument(s). The purpose of the configured Experion server is to basically act as an intermediary between the physical system and the Experion client machine(s). This is illustrated in Figure 1 below. The system must consist of a controller or controllers depending on the size of the system, capable of understanding the Experion environment. In most, if not all locations that operates Honeywell's Experion PKS, follows this model of operation.

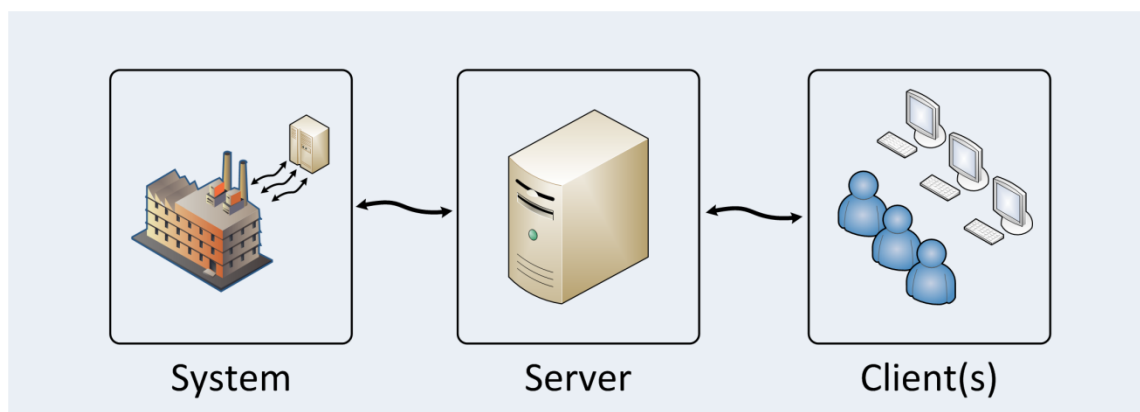


Figure 1: Basic Architecture of an Experion System

Through the development and implementation of a SIM-C300, a number of the software environments were used from the collection integrated into Experion PKS. Configuration Studio, Control Builder, and Station were the key programs utilised. The other significant

environments provided with the Experion PKS package are an important part of the platform but were not used as extensively in this thesis project. In truth, the development of a complete Experion system would have to incorporate the extensive use of all the software environments provided in the Experion PKS software suite.

As all of the seven main core software environments plays its own individual role in Experion PKS that collectively adds up to allow for a fully operational Experion system, each will be given a concise overview explaining briefly its purpose and outlining its importance later in this document.

2.2 Honeywell C300 Controller & Control Execution Environment

The main concept that this thesis project examines besides exploring Honeywell's Experion PKS, is the implementation and operation of a SIM-C300. Due to this fact, an understanding of what an actual Honeywell C300 controller is and how it operates must first be considered.

Taking advantages from both DCS and SCADA topologies, the Honeywell C300 Controller is considered a hybrid controller that offers the benefits of real time control processing between peer to peer devices as well as a potential master slave hierarchy between components from multiple manufacturers (Honeywell International Inc., No Date Given).

Ideal for implementation across all industries, the C300 controller was designed and built to be the best in its class for process control. Along with the fact that it provides support for a wide variety of process control situations such as continuous and batch, Honeywell's C300 controller also allows for the integration with smart field devices across multiple manufacturers (Honeywell International Inc., No Date Given).

Continuous process control is exercised where a system is required to be operational in a smooth and uninterrupted manner throughout time (Honeywell International Inc., No Date Given). An example of this could be keeping a temperature setting. This is achieved through utilising an array of standard functions that are built into control strategies which the C300 controller then executes.

Batch controlled processes on the other hand are used where applications that require strategies to be sequenced in a specific method such as combining material quantities or time duration settings. The C300 controller has been designed to be able to support an

internationally recognised batch control standard. This control method is executed in an almost near perfect manner through the integration of the sequenced instructions with field instruments. These field instruments are able to track the state of the sequences and thus able to perform pre-configured actions. This tight integration leads to quicker transitions and a reduced lost time margin between sequences resulting in increased throughput (Honeywell International Inc., No Date Given).

Honeywell, with all its creativity, has also designed the controller to support its own patented advanced process control algorithm along with custom algorithm blocks which allows users create programs which operate in the C300 controller.

Just like in previous Honeywell controller models, the C300 follows on from past traditions by operating Honeywell's deterministic CEE software which executes control strategies on a constant and predictable schedule that developers have implemented. The CEE that is loaded into the C300 controller's memory provides for an execution platform which may include a comprehensive set of automatic control, logic, data acquisition and calculation function blocks. Each function block (FB) contains a set of pre-defined features such as alarm settings and maintenance statistics that have all been set by the developer. This embedded functionality guarantees a process control strategy execution that is consistently reliable. In addition to all these features, the controller also supports many IO families, including the Series C IO, Process Manager IO, plus other protocols such as FOUNDATION Fieldbus, Profibus, DeviceNet, Modbus, and HART (Honeywell International Inc., No Date Given).

By implementing a Honeywell C300 controller, engineers and operators are able to address the most demanding process control requirements, from integration with complicated batch systems, to controlling devices on a variety of networks such as FOUNDATION Fieldbus, Profibus, or Modbus. As the controller also supports Honeywell's patented advance control which puts model-based predictive control directly in the controller, it may also minimise instrumentation wear and maintenance (Honeywell International Inc., No Date Given).

Part of the Experion PKS package is a core piece of software known as Control Builder which will be later discussed. Control Builder relates in the fact that it is the environment where all of the control strategies are developed for the CEE.

The CEE has the capability of operating control strategies that consists of components from both SCADA and DCS topologies which is heavily emphasised as well as many other functionalities in the system. The code employed by the CEE to write programs and implement control strategies is unique in the aspect that it differs from those currently seen in industry. Honeywell has developed the CEE to use a coding standard based loosely on both Ladder Logic and Function Block Diagrams.

This approach provides a more visual method to writing program codes which dictate control strategies. Each FB has a number of predefined features which, depending on the block used, may include alarm settings, control algorithm selection and maintenance statistics. FBs are combined and interconnected via soft wiring in either Sequential Control Modules (SCMs) or Control Modules (CMs) to create a program that executes in the CEE.

CMs provide the platform that, as stated before, a program, whether it be a complex control strategy, or just a simple two way toggle switch function, is built from. A complex control strategy usually consists of multiple CMs each with unique FB code inside. When CMs are assigned to a CEE, they automatically become a point that is created in the system.

Each FB within all the CMs then becomes a parameter of the CM point. Through this parameter identification scheme that is used throughout the Experion system, every piece of information within a CM may be accessed via its associated point.

The Honeywell C300 controller's CEE supports an execution period that ranges from 50 milliseconds to 2000 milliseconds set by the developer and does not depend on the number of CMs loaded. One of the benefits that the CM architecture offers is that operators can deactivate selected CMs while the remainder in the CEE continues to execute. This then logically, allows for new CMs to be developed and loaded to the C300's CEE without the need of deactivating existing CMs. Another allowance is that full control over FB execution orders within each CM and CM execution orders within the CEE is granted for the user. This is of great importance when coding a sequential execution of FBs is required.

2.3 Configuration Studio

With Configuration Studio, most, if not all, of the basic development and configuration for an Experion system can be accomplished from one central location. This is because all of the major different development tools provided with the Experion platform are launched from

Configuration Studio. As such, this means that Configuration Studio is the central software environment that developers of an Experion system will utilise to initialise the other software configuration tools provided with Honeywell's Experion PKS platform. Illustrated in Figure 2 is Configuration Studio's window and as it can be seen, the number of operations and tasks is quite extensive.

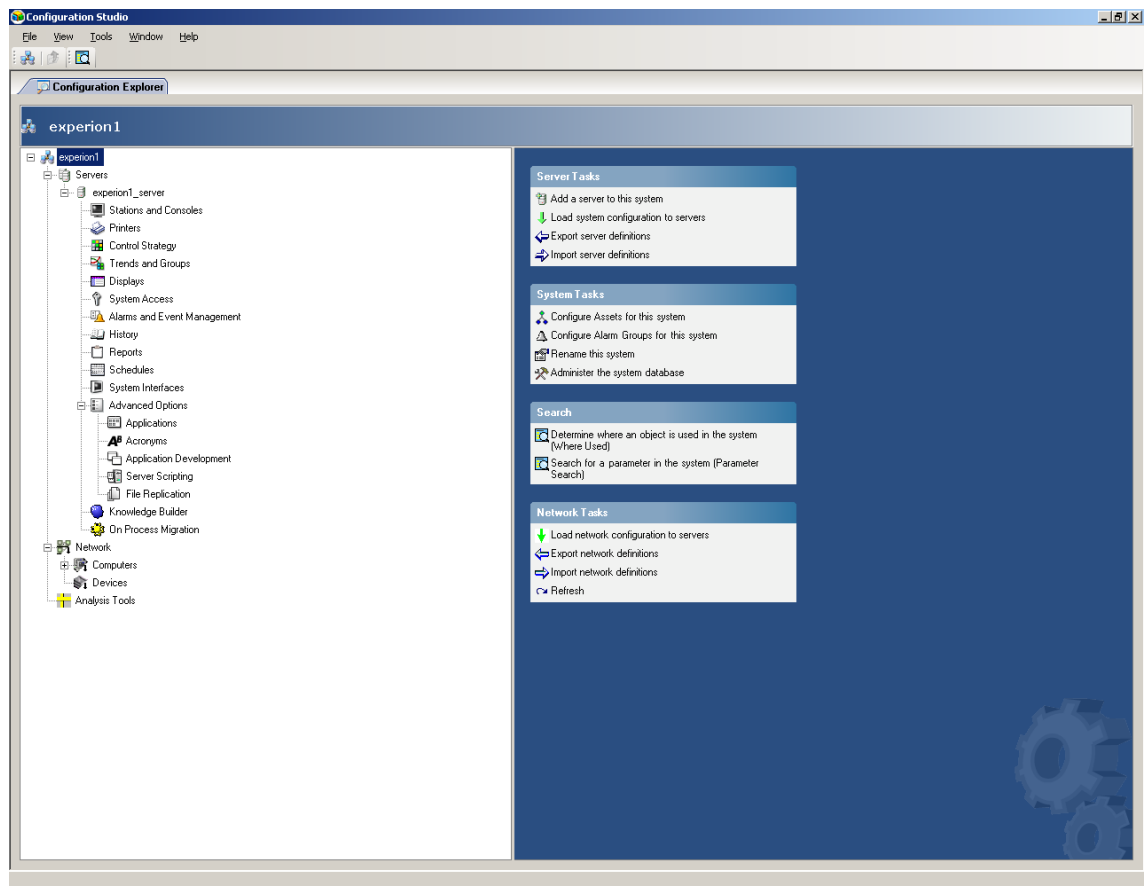


Figure 2: Configuration Studio

The Configuration Studio program also allows users to view and access the many different Experion systems which may be attached to the network when initially launched through a Connect dialog box in the Local Targets tab. An example of this is seen in Figure 3. From here, the desired system that is to be configured is chosen. Access to a system is restricted and the rights to do so has to be firstly, granted by the network administrator.

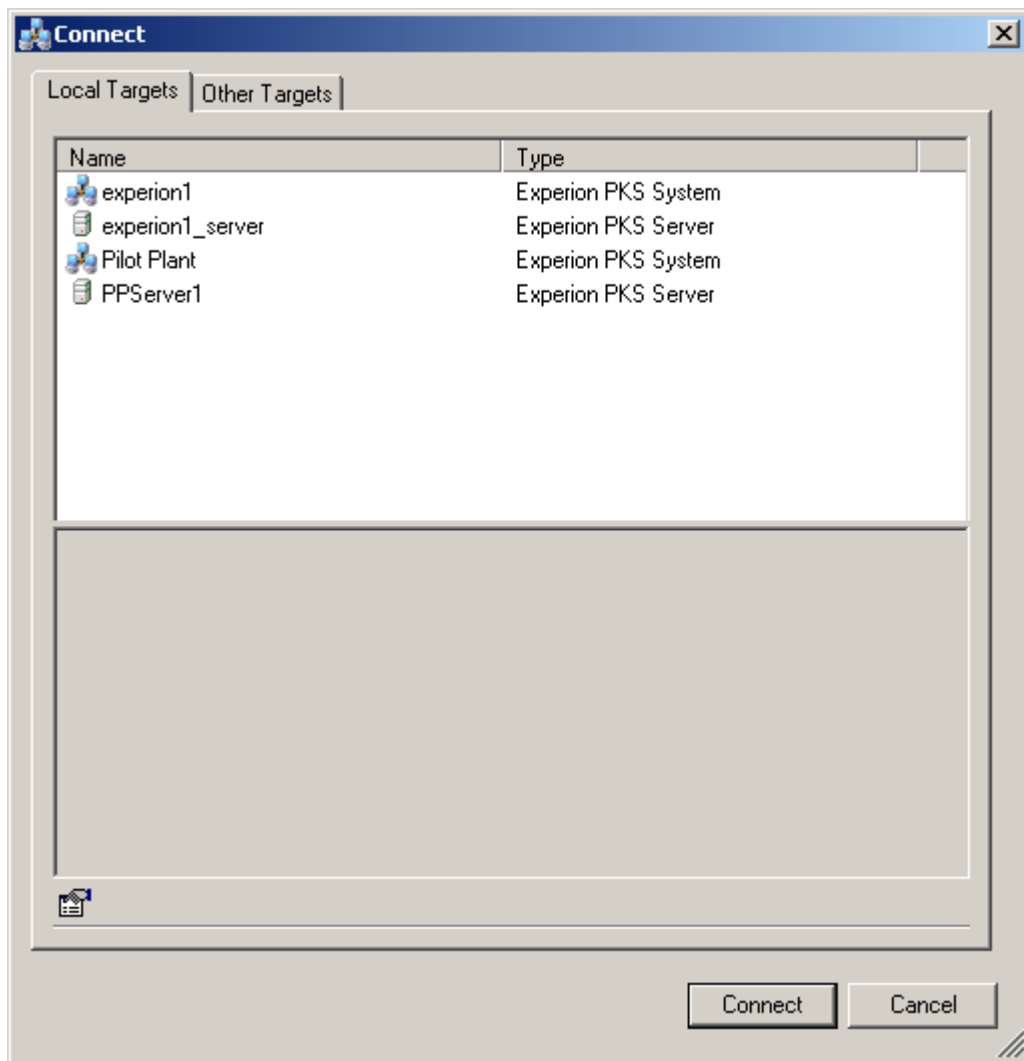


Figure 3: Connect Dialog Window

Worthy of mention is a fact that launching Configuration Studio from experion1's server allows for all other Experion systems to be located on the network which can be seen in Figure 3 just as described earlier. However, this is not the case when Configuration Studio is accessed through the Pilot Plant's server where it is only able to see its own system. The cause of this is believed to be that Honeywell's Experion PKS loaded onto experion1 has a feature which is directly linked with its license. This feature, a distributed server component, is not available with the license provided with Pilot Plant's Experion system.

Another explanation which is provided by Honeywell's Knowledge Builder, states that when Configuration Studio is used to connect to a system or server through a router that has multicast traffic blocked, the Local Targets tab of the Connect dialog box will not list the system or server names that belongs to a remote subnet. This is, as of yet, to be explored fully in detail.

To be able to access or view other Experion systems when operating Configuration Studio through the Pilot Plant's server, the user must locate the particular system in question through the Other Targets feature.

2.4 Control Builder

Control Builder is the main development tool which is used for the implementation of controllers along with the development and loading of control strategies. It possesses an intuitive graphical user interface for building or programming control strategies into the CEE of a controller, which in Murdoch University's case, is either a real or simulated Honeywell C300 controller.

For purposes of this thesis project, Control Builder was used to implement and program a simulated Honeywell C300 controller which will be discussed in this paper at a later stage.

In Control Builder, Honeywell has developed a coding language that is somewhat rather unique but at the same time similar, in comparison to those that are currently available such as Ladder Logic.

Ladder Logic is programming language that is primarily used in the development of software for Programmable Logic Controllers (PLCs). It is a language where the program is represented through using graphical diagrams that are based on circuit diagrams of relay logic hardware. Due to this intuitive nature for people with the appropriate background and knowledge, it has become an immensely popular language used to program PLCs that are employed in industrial control applications.

With regards to Control Builder, the programming strategy or ideology as mentioned before is loosely based on Ladder Logic in combination alongside the idea of Function Block Diagrams which is also popular among PLCs and other control systems such as DCS. Honeywell, through this object-orientated and graphical code language, has developed an innovative approach to programming. The philosophy behind developing an alternative coding standard is based on the idea in the hopes that it will reduce the amount of effort that is required by developers to design, implement and document control applications for an Experion system.

This methodology has both its advantages and disadvantages. Ideally, the approach could work as intended and allows developers to have a simplistic way of coding. However, looking on the other side of the fence, this initiative that Honeywell has taken could backfire as the language is unique and thus, it may be troublesome for new users. Since it is a new and proprietary style of coding, experienced users, or in other words Honeywell, would have to be contracted in to assist which could be part of their intended marketing strategy. However, one could argue that Honeywell has designed Experion PKS in mind that the intended customer base would be people with the background and knowledge in such systems already. But as always, in reality, aid would still be required for such a complex software suite.

Looking at the specifics of Control Builder, there are multiple operations and options that are available to the user. This can be examined in Figure 4. From Figure 4, it can be seen that Control Builder has three main sections, a Project window which contains two tabs (Section 1); Project and Monitoring, a Library Window (Section 2) and an area for where control strategies are edited (Section 3).

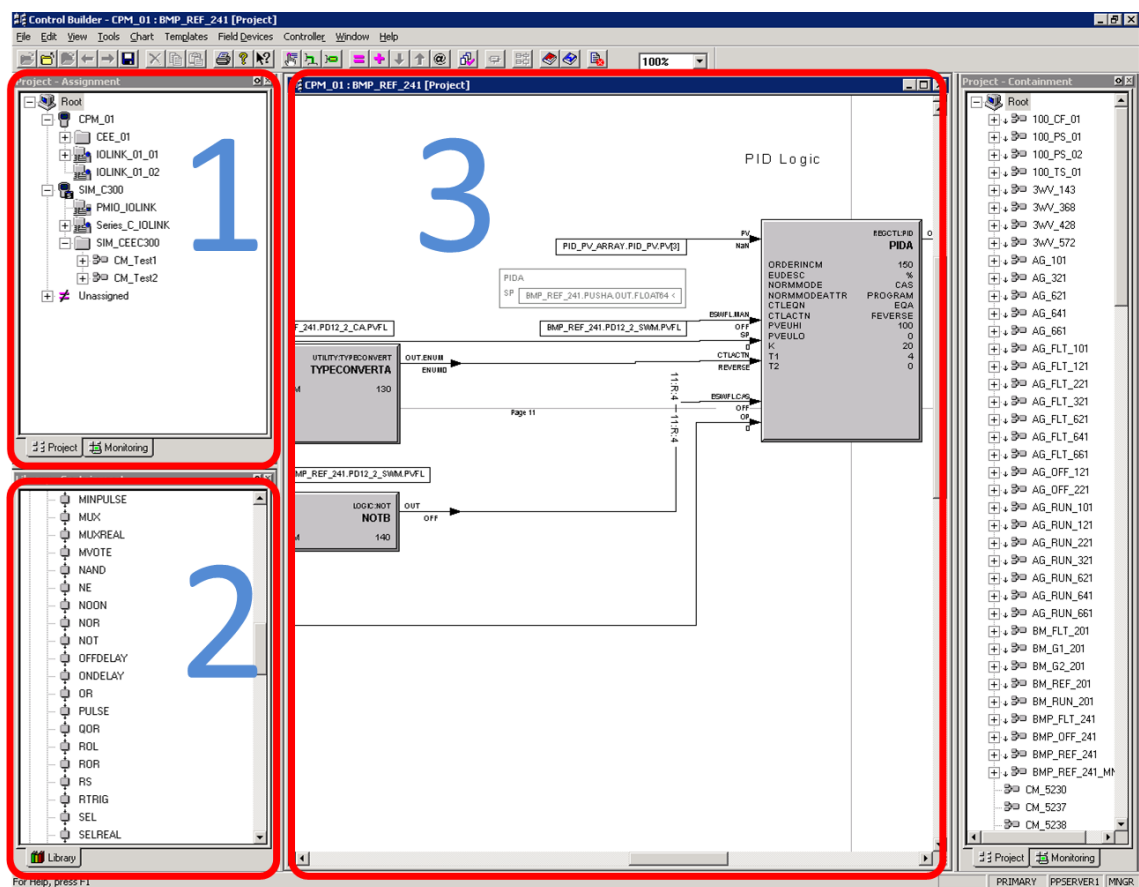


Figure 4: Control Builder

The Project tab part of the Project window consists of a tree list for each and every Central Processing Module (CPM) available. Within each CPM, a CEE is housed. To current knowledge, each CPM may only house one CEE. Found inside the CEE are all the CMs which it supports. Contained inside all of these CMs is the code that is behind the operation of an Experion system. This code, as outlined earlier, is loosely based on Ladder Logic in combination alongside the idea of Function Block Diagrams.

As seen in the Project tab's tree list, located on the same level as the CEE, is the IOLINK which consists of all the Input and Output Modules (IOMs). Part of the Project window, is also the Monitoring tab. This tab is mainly just a reflection of the Project tab with the difference being that it allows for online monitoring of all the FBs. As seen in Section 2 of Figure 4, is the Library window and tab. Within the library, is an extensive collection of FBs, where during the development of control strategies, are obtained to be placed into the CM dragging the required block into it.

In Figure 4, Section 3 is an area where all the control strategies are edited or monitored depending on the type of CM that is open. If the CM is opened from the Project tab, it will thus allow the user to be able to edit the code that is contained within with the appropriate privileges. If opened from the Monitoring tab, the CM provides for the online monitoring of all FBs, or the program that it contains.

It should be mentioned that the final developed block diagram or control strategy allows for its own Human Machine Interface (HMI) that may be developed in Experion's HMIWeb Display Builder.

2.5 Enterprise Model Builder

Although not used or explored in this thesis project, a short description will be provided on Enterprise Model Builder due to the fact that it may or may not have important implications when constructing a full Experion system.

Enterprise Model Builder is an application provided with Honeywell's Experion PKS software suite that is used to build, edit and download an enterprise model into an Experion system. The enterprise model provides a means of organising the Experion PKS system around key entities such as assets, material, activities and people (Honeywell International Inc., 2007). There need to explore Enterprise Model Builder to fully understand its inner workings and how

it functions in relation to an Experion system. From current understanding, Enterprise Model Builder provides many of the functions that Configuration Studio is able to execute and thus may not be needed immensely.

2.6 HMIWeb Display Builder

HMIWeb Display Builder is a specialised drawing application used by developers to create custom displays for Station. These custom displays are usually operator screens that displays information linked to a fully operational Experion system. The screens are almost always presented in a sophisticated and user-friendly manner so as to enable the operators of Station to interpret what is being seen at ease. The HMIWeb Display Builder software also allows for the Abnormal Situation Management (ASM) standards to be taken into account during the construction and implementation of displays if desired (Honeywell International Inc., 2007). HMIWeb Display Builder was not fully utilised throughout this thesis project.

2.7 Quick Builder

Although Quick Builder was not employed in this thesis project, it has one of the most vital roles in constructing an Experion system. In general, Quick Builder has the tools that allows for the higher level configuration of Experion system servers and Station client machines.

Through the development of Experion systems, Quick Builder is deployed for linking collected data from SCADA points to servers. It is also used because it allows for the construction of controllers and points that are directly associated with SCADA channels. In essence, Quick Builder allows the developer to create and modify configuration databases. Databases play an important role in an Experion setup because of the fact that they define how system items, such as controllers, points and Flex Stations are structured (Honeywell International Inc., 2007).

2.8 Station

Station is the client software that provides the main user operating interface to most, if not all developed Experion systems. Station works in conjunction with an Experion server to monitor and control a system. In effect, Station is primarily a set of control panels presented through a series of displays. There are two fundamental types of displays offered in Station; supplied displays which are pre-built, and custom displays which are specifically created. Station also has the ability to display Web pages and files, such as Microsoft Word documents, which typically contain operating procedures (Honeywell International Inc., 2007).

In addition, there are several types of Station and several different environments in which it operates. Only the two main types of Station, Flex Station and Console Station will be discussed. The term Station is generically used throughout this document. Flex Station or Console Station will be directly termed if there are applications and functionalities particular to the type of Station (Honeywell International Inc., 2007).

Flex Station is an interface that is used to interact with the system, as described above. Generally, a Flex Station is installed on a computer other than the Experion server and can use either the rotary or static connection type. The differences of these connection types will be explored after an explanation is provided on Console Stations (Honeywell International Inc., 2007).

Console Stations provides all of the functionalities that are available on Flex Stations, but in addition Console Stations have direct access to the process controllers. This thus enables an operator to monitor and control a system regardless of the state that the Experion server is in. There are key differences between a Console Station and a Flex Station. Due to Console Stations having direct access to process controllers and thus bypassing an Experion server, it has its own connection type. Console Stations are also not built using Configuration Studio, and thus is installed in a different manner to Flex Stations (Honeywell International Inc., 2007).

In an effort to allow a fuller understanding, rotary and static connections will now be explored in greater detail. A static connection provides a permanent, dedicated link to a specific Station interface and as such, should only be utilised when Station will be in operation around the clock. In direct opposition is the rotary connection where a link to Station is provided on an as required basis and is the recommended connection type where Station is not used full-time (Honeywell International Inc., 2007).

Regardless of whether a Station is defined as static or rotary, the maximum number of Stations that can be connected to the server (and running the Station software) at any one time is determined by the license purchased from Honeywell. By defining most, if not all rotary when a static connection is unneeded, a larger number of Stations can be installed and configured in the system. A limitation of this is that all the Stations defined cannot all be connected to the server simultaneously. As a result, rotary connections are advantageous from a licensing

perspective because Experion PKS only specifies the total number of simultaneously connected Stations (Honeywell International Inc., 2007).

Station itself is very much configurable in the fact that it allows for web access, changes to its menus, toolbars, keyboard shortcuts, connections, and appearance. The Station application allows operators of an Experion system a means of interacting with the FBs and the associated code written in Control Builder through the supplied displays or custom displays designed through HMIWeb Display Builder. In Station, the user may operate the Experion system through the developed displays and also execute many other functions such as tune preconfigured control loops, view trends and view historical data. With the appropriate level of access, Station also allows capability for system configuration such as building new control and SCADA points and defining operator logon accounts through the System Displays whose functionalities is explained in the next paragraph (Honeywell International Inc., 2007).

2.9 System Displays

The system displays that can be called up in Configuration Studio are used to configure a number of items such as reports, group display, trends, Station settings, Console Stations, and so on. Like Configuration Studio, it is also capable of launching the other software environments in the Experion PKS package. These system displays may also be launched through the client machine application, Station. Other functionalities that the system displays offer include the ability to execute the following listed tasks:

- Check the status of components to verify that they have been configured correctly.
- Check that all components are at an operational status.
- Respond to system alarms relating to errors such as communications failures, Station failures, operator logon failures etc.
- Monitor the status of the entire Experion system to prevent problems and errors from occurring.
- Diagnose problems in the system when they do arise.
- Review the firmware versions loaded onto C300 controllers.

Chapter Three: Accomplishments

This chapter of the report examines the tasks that were completed as part of the thesis project undertaken. Discussed are the challenges encountered and how they were overcome and the many goals that were accomplished.

3.1 Experion PKS Installation & Setup

As described earlier, the architecture of an Experion system uses a server to act as the interface between the physical system setup and the client workstation computers. These workstation computers are installed with an Experion PKS software version that supports client applications as opposed to server applications. The main environment provided with the client version of Experion PKS is Station, which is used to operate a system, and with the right level of access, a portal into the server to allow for the configuration of the entire system through its ability to launch many of the software tools required for setup.

Before any development or exploration of Honeywell's Experion PKS could take place, the server package had to be installed onto a dedicated machine which will operate, as the name suggests, a server. The client package of Experion PKS had to be also installed on machines, or in the case of this thesis project, just one machine that will be dedicated to becoming a workstation for the new Experion system.

William (Will) Stirling, a technical staff member of Murdoch University's School of Engineering and Energy took charge of the installation process.

By establishing a new Experion server, a new system was automatically created for it. This system would be then independent of the server. Basically, the new Experion server and newly created system are of separate entities and are not initially linked and had to be manually set up so that they are associated with one another. The server machine houses the new configured system represented in a software sense, which houses the server itself. In the most basic of terms, the server machine is just a piece of hardware. By examining the setup of the Experion software, the physical system is represented in a software sense with the server being part of the system. Figure 5 shown below outlines this explanation and should assist with the understanding.

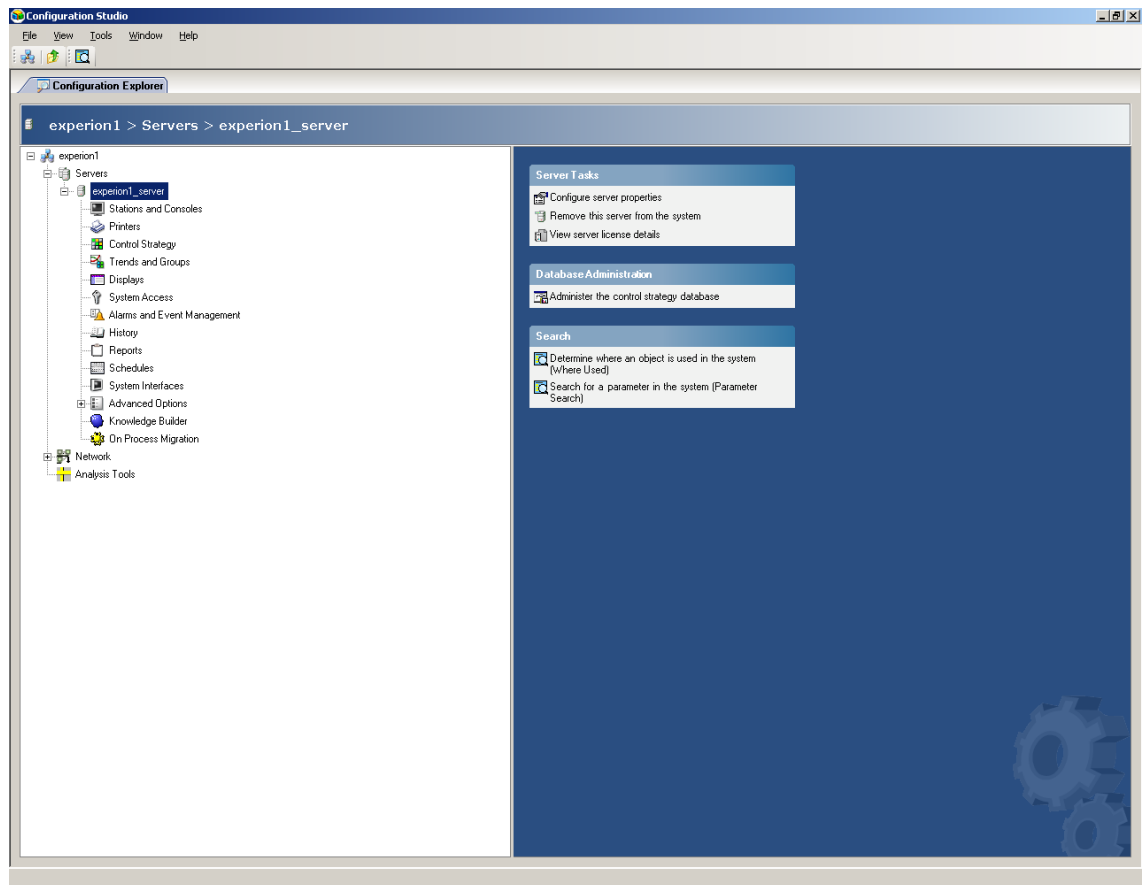


Figure 5: experion1 System Server

Murdoch University's Pilot Plant Experion setup is a prime example to showcase this explanation. The Pilot Plant is a physical system (named Pilot Plant) and is represented in Configuration Studio as so (Figure 6). Its server is named ppserver1 and is part of the system. The server sits in the system and the system is accessed through the server machine.

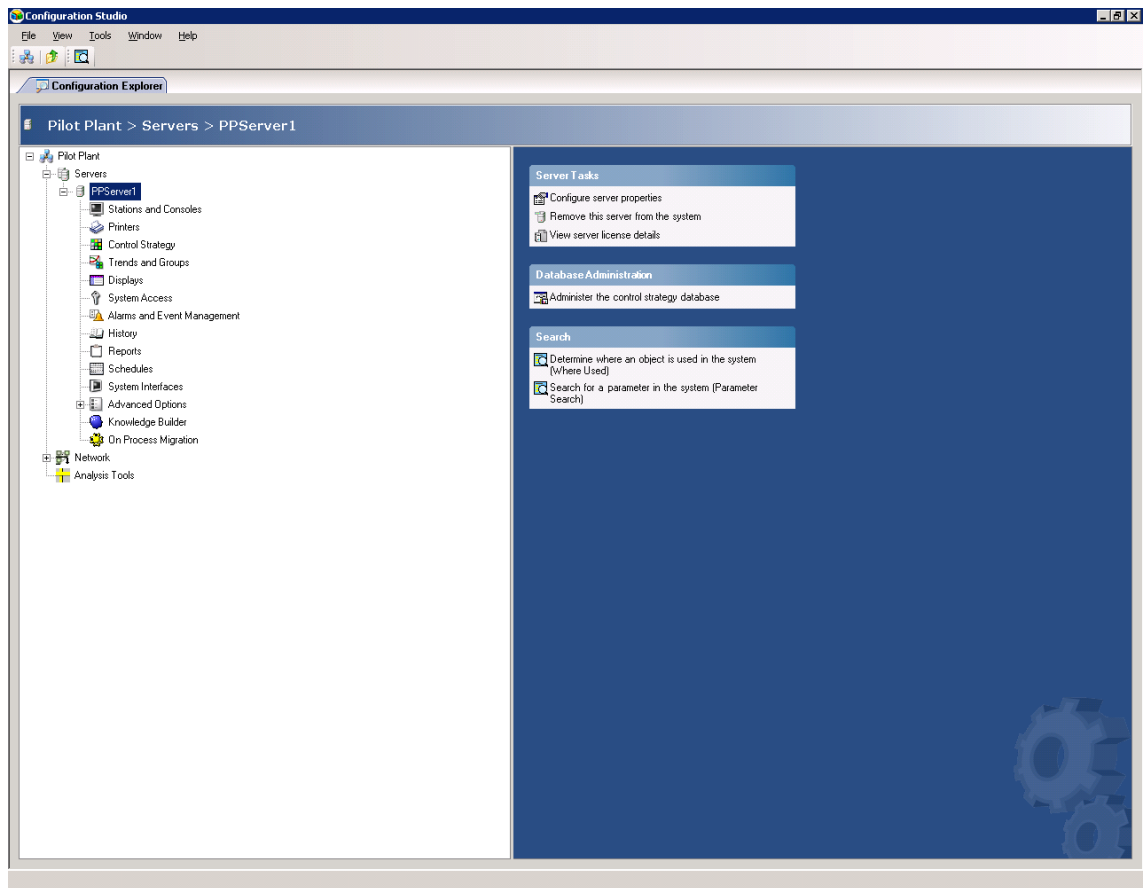


Figure 6: Pilot Plant System Server

For the new Experion system to be operational, it first was linked to its dedicated server. This task was the first step that was executed before any other activity related to a SIM-C300 controller occurred, which in this case, was completed without any incidents. The system was named to be experion1, and its associated server, experion1_server as seen in Figure 5.

3.2 SIM-C300 Configuration

The SIM-C300, as stated many times earlier, supports the full simulation of an actual Honeywell C300 controller. Honeywell's Experion software suite does not differentiate between an actual or simulated C300 controller and will operate according to how it has been set up.

Honeywell's Experion PKS software package has many different tools each specific to its tasks. At this stage, to implement the SIM-300, it is only crucial to understand how to operate two of the many software tools available; Configuration Studio and Control Builder.

As explained in detail earlier in this document, Configuration Studio is the main management program that holds all the software development tools to configure an Experion system and thus, is the first of the two software environments needed. In being the main management program inside Experion, Configuration Studio is the central access point, or operation environment to launch all the Experion system development tools as well as access to other Experion systems on the network. It thus, provides access to the second software environment needed to implement a SIM-C300, Control Builder.

Control Builder is the main development tool that handles the implementation of controllers, and as such, the SIM-C300 is implemented from this environment as stated earlier. The Control Builder environment is also where control strategies are developed and loaded from, into the CEE of a C300 controller or SIM-C300. The two main steps involved in implementing a controller are to firstly create it, and secondly, load the created controller onto the system. Other details which are minor in comparison at this stage are creating IOMs as well as loading a CEE. This is because without the parent SIM-C300, none of these would matter, and thus, would be categorised at this stage of the project as minor.

Creating the SIM-C300, IOMs and a blank CEE via Control Builder occurred without any issues. Upon attempting to load the created simulated controller into the new Experion system, experion1, however, did not.

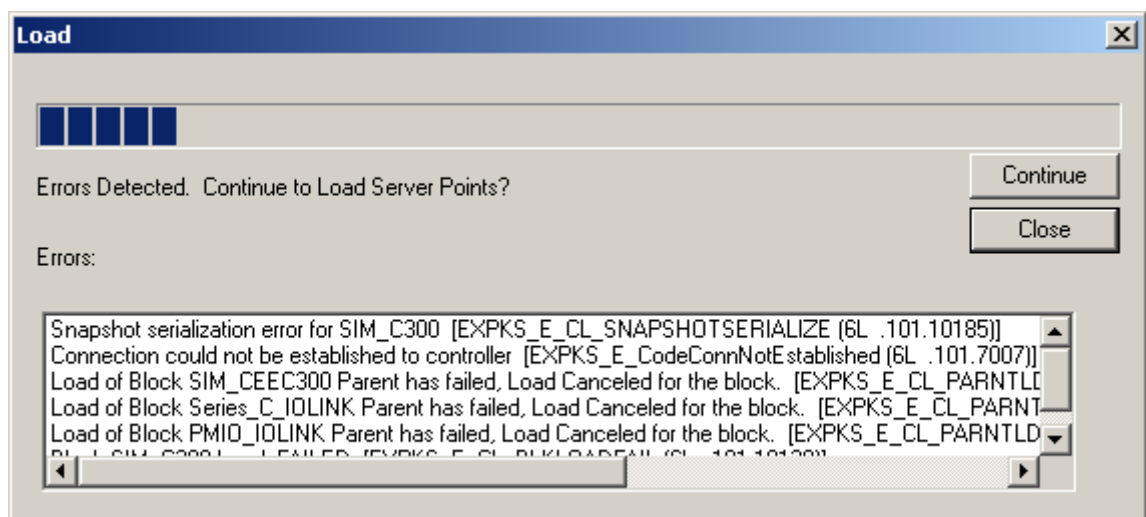


Figure 7: Errors Detected

It was quickly realised that the loading process for the SIM-C300 and all other associated parts (IOMs and CEE) was failing on each attempt as seen in Figure 7 above. Error messages indicating a load failure had occurred appeared during the load phase. These messages are seen below.

- Snapshot serialization error for SIM_C300 [EXPKS_E_CL_SNAPSHOTSERIALIZE (6L .101.10185)]
- Connection could not be established to controller [EXPKS_E_CodeConnNotEstablished (6L .101.7007)]
- Load of Block SIM_CEEC_300 Parent has failed, Load Canceled for the block. [EXPKS_E_CL_PARNTLDFAIL (6L .101.10342)]
- Load of Block Series_C_IOLINK Parent has failed, Load Canceled for the block. [EXPKS_E_CL_PARNTLDFAIL (6L .101.10342)]
- Load of Block PMIO_IOLINK Parent has failed, Load Canceled for the block. [EXPKS_E_CL_PARNTLDFAIL (6L .101.10342)]
- Block SIM_C300 Load FAILED [EXPKS_E_CL_BLKLOADFAIL (6L .101.10139)]

By analysing the error messages provided by the failed loading of the SIM-C300, it was determined that the initial SIM-C300 block did not appear to load due to a connection error and thus, caused a chain reaction of other load failures as all other blocks rely on the SIM-C300 to be successfully loaded. At this stage, it is still unclear why this connection error occurs.

A test was then executed to see if the same errors would arise on a different Experion system, and in this case, the Experion system that is operational at the Murdoch University's Pilot Plant Facility.

Following the exact same procedure that was used to implement a SIM-C300 on experion1, a SIM-C300 was able to be loaded successfully without any issues at the Pilot Plant Facility which gives an indication that the problem lies within experion1. An attempt to rectify this issue was set into motion. This involved clearing the, at the time, current setup of Honeywell's Experion PKS on experion1 establishing a new install. The difference between the two is that certain features that were not part of the original installation process were added.

Due to the fact that the Experion platform installation requires approximately a period of two to four working days to complete, a decision was made to continue working during this time just in case the attempt to resolve the problem fails. Hence, the Pilot Plant Facility was utilised

once again as it was already proven that a SIM-C300 could operate at the location. There are however, certain aspects of the Experion PKS platform at the Pilot Plant Facility which are not to be altered because it serves as an important learning tool and thus, must be able to operate at any time.

Upon the news that the new experion1 system was again ready for configuration, a SIM-C300 was immediately implemented and assessed for operation. The attempt to rectify the issue however, did not result in the connection error to the SIM-C300 being resolved. As of present, the dilemma is in spite of everything, still being investigated to try and provide a solution along with an explanation as to why this is happening.

Consequently, the only viable option available for this thesis project was to continue using the Experion system employed at the Pilot Plant Facility. In doing so, certain aspects were not to be modified or altered in any manner with the two most important being the actual Honeywell C300 controller operating the facility, and the CEE loaded on it.

3.3 Control Execution Environment

For a C300 controller to operate, whether it is an actual or simulated controller, it must first be programmed with instructions to do so. The CEE is where this takes place.

Besides from a C300 controller, an Experion system requires many other instruments and devices. It is through IOMs that are associated with the controller in question, that form the communications link between the hardware to the software aspect. Two IO Links are automatically assigned to a controller when it is first created and can be configured by the developer. IOMs which are created are assigned to the respective IO Link. Channels of communication that are house inside each IOM can then be configured for operation.

In this case, one of the IO Links was configured to support an older communication link used with the dated Honeywell C200 controller and thus holds all the channels which are available for it. This IO Link was given the name PMIO_LINK. The other module was of course, set up so that it consists of the more recent Series C communication channels which are mainly used with current controller technology from Honeywell and was named Series_C_IOLINK to suit. The inclusion of the older IO Link shows the versatility of Experion PKS and how it is backwards compatible with more dated technology.

All IOMs associated with its controller are located under the IO Link which it has been assigned to in the Project tab tree. Consequently, all channels housed inside each IOM are located underneath it and thus are available for the SIM-C300's CEE.

Each channel within an IOM is automatically assigned a pre-configured FB with all its associations such as which IO Link it is from. With this, a channel can then be used in the development of a CEE. Channel FBs are also available in the library tab's collection however all of its pre-configuration settings linking it to a related IO Link are not set up. Therefore, it is highly recommended to use the channels available in the IOM when programming a CM for the CEE. The CEE of a controller allows for as many CMs as required and is where the backbone of where an Experion system is programmed. Each channel FB is restricted to be used for the CEE of the controller that it has been assigned to and no other.

3.4 PID Function Block

The Experion platform was designed to be a computer based control system and therefore its general operation in all cases would employ the use of a Proportional Integral Derivative (PID) block. For this reason, the CEE that is to be developed must incorporate such a block but before doing so, the block's operation must be understood. Figure 8 depicts the graphical representation of the PID block in Control Builder.

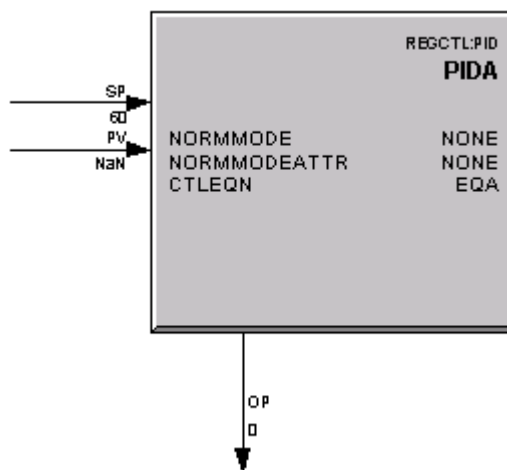


Figure 8: PID Function Block

An extensive collection of FBs lie within the library tab located inside the Control Builder environment. One such block is the PID block which is a regulatory control block that operates

as a PID controller which supports the Ideal form of calculating the PID terms. The Ideal form is often called and known as the digital computer version of the PID controller.

The PID block may be used in a single control loop or with multiple PIDs in a cascade strategy and has two analogue inputs; a Process Variable (PV) and a Set Point (SP) as would be expected. The Control Output (OP), which is generally taught at Murdoch University to be the Manipulated Variable (MV), is calculated to drive an error to zero. An error is the difference between PV and SP.

Located amongst the many other FBs found in the library's extensive collection, are advanced function blocks. These blocks differ from most in the fact that it allows for more than a singular function. The PID block is one such example of an advance function block which is also a Regulatory Control (REGCTL) block. A feature that separates the common basic and mathematical blocks to a PID block is the fact that its operation can be altered.

The means of changing its operation is accomplished through a server display which has been configured in the block itself. Server displays are a set of pre-configured HMIs for specific general purposes provided with Experion that can be utilised in Station. Manual and Automatic are the two most widely used options when selecting the mode of operation for the PID block. Figure 9 illustrates the server display which also highlights where the mode of operations are configured. The MD drop down menu is where the Manual and Automatic modes are configured from. Another option that is available when selecting the mode of operation but is more popularly seen and used in process industries is the Cascade mode.

Cascade mode is utilised when a PID block's SP is being set from a REGCTL block, which can be another PID block or even a switching block. This mode will then allow reading and writing between two REGCTL blocks and a host of other smart features which are beyond the scope of understanding and project at this point.

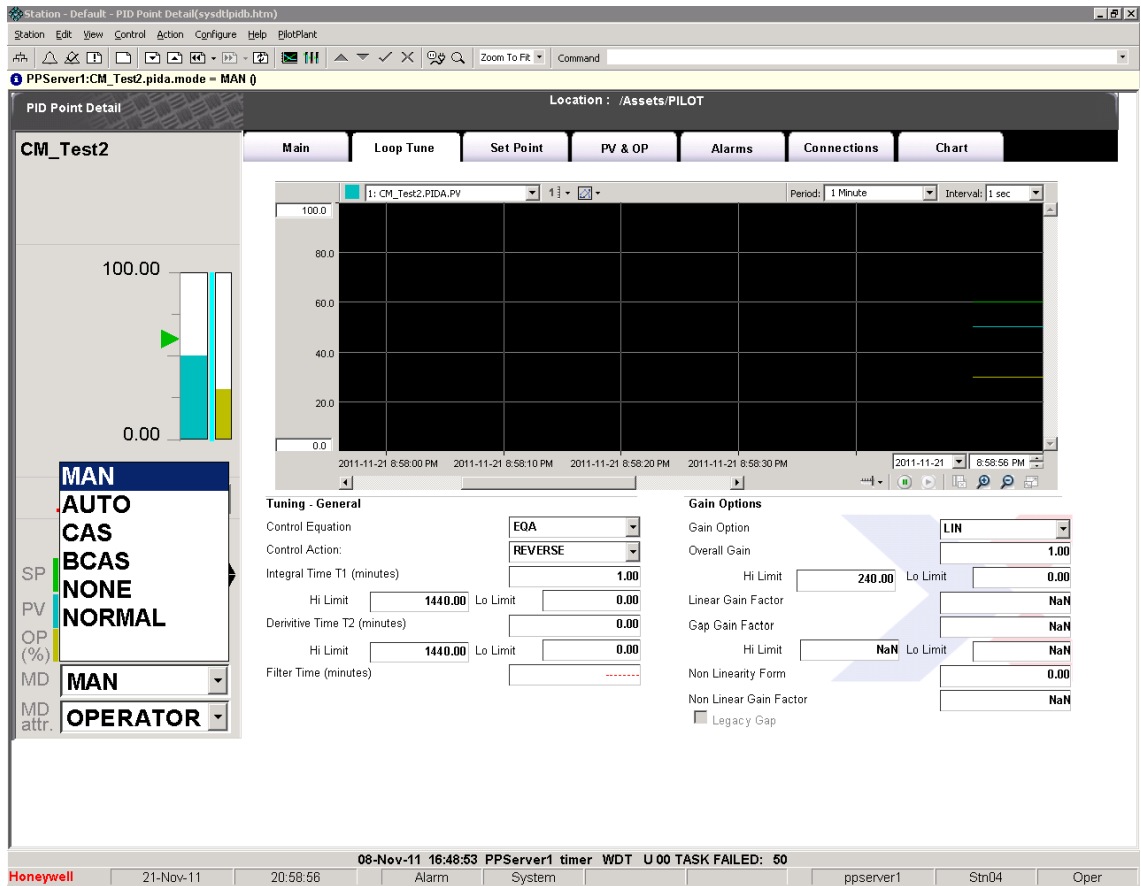


Figure 9: Server Display Showing Manual & Automatic Modes

Program and Operator are the two mode attributes available for selection in the PID block server display. Differences between these two mode attributes are directly related to how the block has been coded and programmed in the CM.

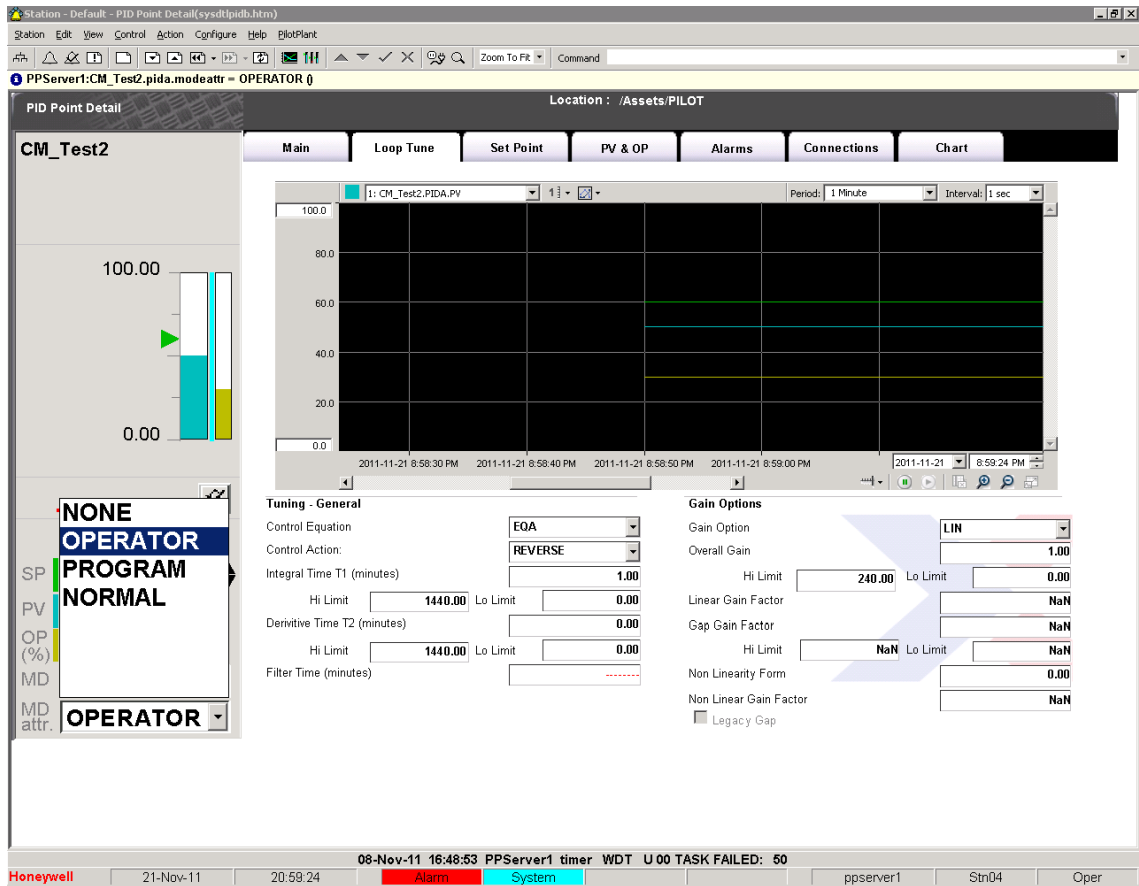


Figure 10: Server Display Showing Operator & Program Mode Attributes

When the PID block has been set to Program mode, data is received from other FBs. This means that all block functions are completely controlled by the individual blocks that are wired into inputs of the PID block. This leads to the PID block being pre-configured and thus, when placed in program mode it is able to apply its PID equation to the wired inputs and produce a single OP output. For example, if a numeric block has been wired to feed into the SP of a PID block which has been selected to operate in Program mode, then the SP of the block will be taken to be the value in the numeric block.

Operator mode allows for the user to write directly to the register of a PID block's point and thus does not require a FB wired into it. All FBs that are wired into the inputs of the block are therefore ignored. Consequently, a SP value can be changed directly from the server display.

Depending on what mode attribute has been selected, the mode of operation, Manual or Automatic, can determine what response is generated. Any selected mode of operation with a Program mode attribute will not allow for any changes via the server display which is to be expected given what Program mode does. With an Operator attribute however, Manual mode

allows for the user to control the entire response. In an Automatic mode with an Operator attribute, the PID takes over where only the SP can be changed by the user.

Many other aspects of the PID block can be altered and configured to suit development. One such aspect is direction of control action which can be altered between direct and reverse. By effectively changing the control action of the PID block changes the sign of the gain. With direct control, an increase in the error will see the output of the PID increase. A reverse control action will lead to a decrease in PID output when an increase in error is detected.

Another parameter that is configured by the developer, and perhaps the most important of all, is the equation that the PID block will take on when applying a control scheme. The PID block provides four different equations that can be applied to calculate its algorithm which can only be changed within the block's parameters and not through any other means such as station. Each equation that is available will now be explored.

For each equation, the following acronyms in the table below stand true.

| Acronyms used in the PID Equations | |
|------------------------------------|--|
| CVP | Output of PID in percentage |
| K | Gain (Proportional Term) |
| L^{-1} | Inverse of Laplace transform |
| PV | Process input value in engineering units |
| PVP | PV in percentage |
| α | 1/16 fixed rate amplitude |
| s | Laplace operator |
| SP | Set point value in engineering units |
| SPP | SP in percentage |
| T_1 | Integral time constant in minutes |
| T_2 | Derivative time constant in minutes |

Table 2: Acronyms used in the PID Equations

3.4.1 Equation A

Equation A is where all three terms of the PID block (Proportional, Integral and Derivative) act on the error. It is also the most commonly equation taught to undergraduate electrical engineering students at Murdoch University (Ogunnaike & Ray, 1994).

$$CV(t) = K \times L^{-1} \left[\left(1 + \frac{1}{T_1 s} + \frac{T_2 s}{1 + \alpha T_2 s} \right) \times (PVP(s) - SPP(s)) \right]$$

(Honeywell International Inc., 2007)

3.4.2 Equation B

This equation causes only the proportional and integral terms to act on the error whilst the derivative term acts on any changes to the PV. As a result, this leads to quick changes in the SP and therefore derivative spikes tend to be minimised or eliminated in the control action.

$$CV(t) = K \times L^{-1} \left[\left(1 + \frac{1}{T_1 s} + \frac{T_2 s}{1 + \alpha T_2 s} \right) \times PVP(s) - \left(1 + \frac{1}{T_1 s} \right) \times SPP(s) \right]$$

(Honeywell International Inc., 2007)

3.4.3 Equation C

If Equation C is utilised by the PID block, the integral term responds to the error and any changes to the PV are acted upon by both the proportional and derivative terms. Equation C provides the smoothest and steadiest response to changes in the SP compared to the three other equations although it is also the slowest.

$$CV(t) = K \times L^{-1} \left[\left(1 + \frac{1}{T_1 s} + \frac{T_2 s}{1 + \alpha T_2 s} \right) \times PVP(s) - \left(\frac{1}{T_1 s} \right) \times SPP(s) \right]$$

(Honeywell International Inc., 2007)

3.4.4 Equation D

This equation differs more from the previous three mentioned in the fact that it only employs the use of the integral term to produce a response.

$$CV(t) = L^{-1} \left[\frac{1}{T_1 s} \times (PVP(s) - SPP(s)) \right]$$

(Honeywell International Inc., 2007)

3.5 Microsoft Excel Data Exchange

Microsoft Excel Data Exchange (MEDE) is an Add-In option that allows for the capture of real-time point value and history information from an Experion system which can be then displayed in an Excel spread sheet.

To retrieve data, MEDE must first be configured using the MEDE Wizard or through cell formulas. Data retrieved may consist of current point parameter values or historical data, both of which are obtained from the server's database. Current captured data values may be refreshed at a maximum rate of one second set through the "Set Periodic Recalculation Interval" option.

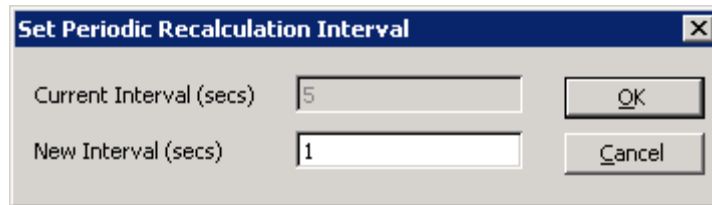


Figure 11: Set Periodic Recalculation Interval Window

3.6 A Simulated System

Upon the completion of establishing the core essentials that provide the operation of Experion, coding began into a CM belonging to the SIM-C300's CEE to test, experiment and demonstrate its functionality.

The program developed was based on a simple first order system which incorporated a number of numerical, data acquisition, and channel FBs along with one Proportional Integral Derivative (PID) block. By starting out with a simple and small scale system, the functionalities can be explored with ease. Any bugs in the code can be easily ironed out and fixed without causing a domino effect which may occur in a larger program. The equation that the system was based on is outlined below.

$$y_n = (\varphi \times y_{n-1}) + [k \times (1 - \varphi) \times u_n]$$

(Ogunnaike & Ray, 1994)

| Equation Notations | |
|--------------------|-----------------------------|
| y_n | Process variable |
| y_{n-1} | Previous process variable |
| k | Process gain |
| Δt | MEDE recalculation interval |

| | |
|-----------|---|
| u_n | Output or manipulated variable |
| τ | System's speed |
| φ | Constant: $e^{\left(\frac{-\Delta t}{\tau}\right)}$ |

Table 3: Equation Notations

By utilising Microsoft Excel Data Exchange, the system based on the above equation was able to be simulated as seen in Figure 10. Coded into a CM, the program allowed for read and write functionalities through the configured server display as well as with an Excel spread sheet.

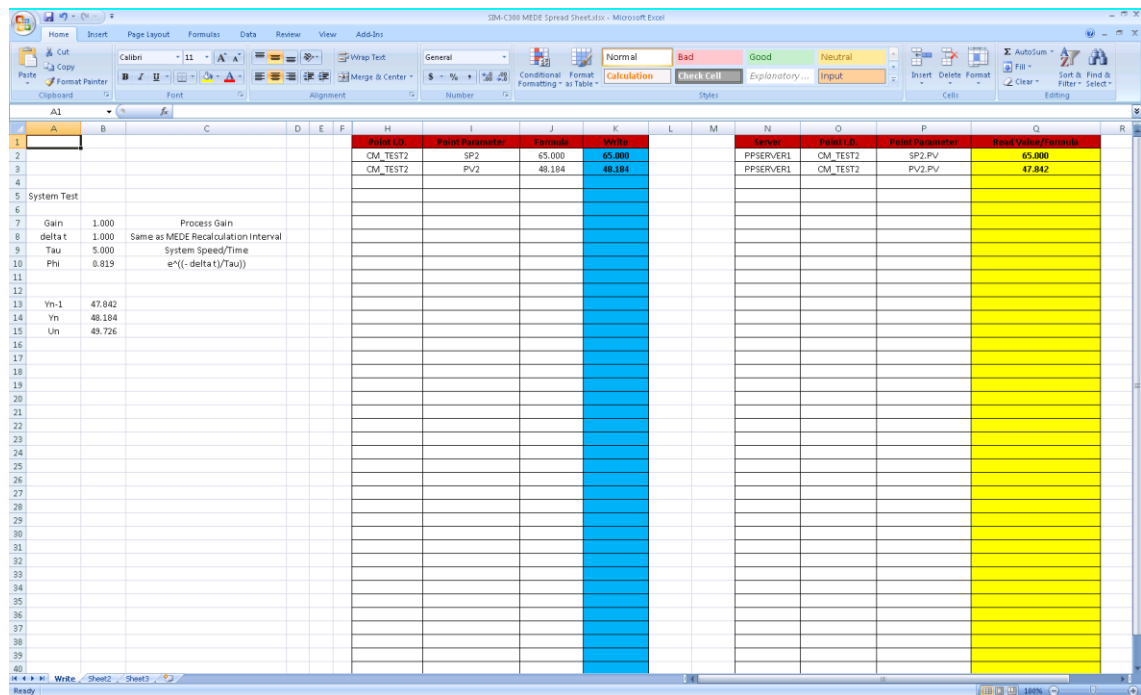


Figure 12: Simulated System Using MEDE

When the program for this simulated system was being developed, an issue encountered which as of present is yet to be resolved. The problem is concerned with wiring a numeric FB into a PID block's SP. Although the PID block's SP was programmed to receive the data from a numeric block, it does not perform as such during operation even when the mode attribute has been set to Program. Therefore, any SP value that is fed to the numeric FB via an Excel spread sheet will not be passed to the SP pin of the PID block. All SP changes must be manipulated through the server displays.

The server displays are configured through Control Builder when a CM has been created. Within the CM's Module Properties, there is an option to select the desired server display. For a CM that consists of a PID block, the server display "SysdtlpidA" should be used.

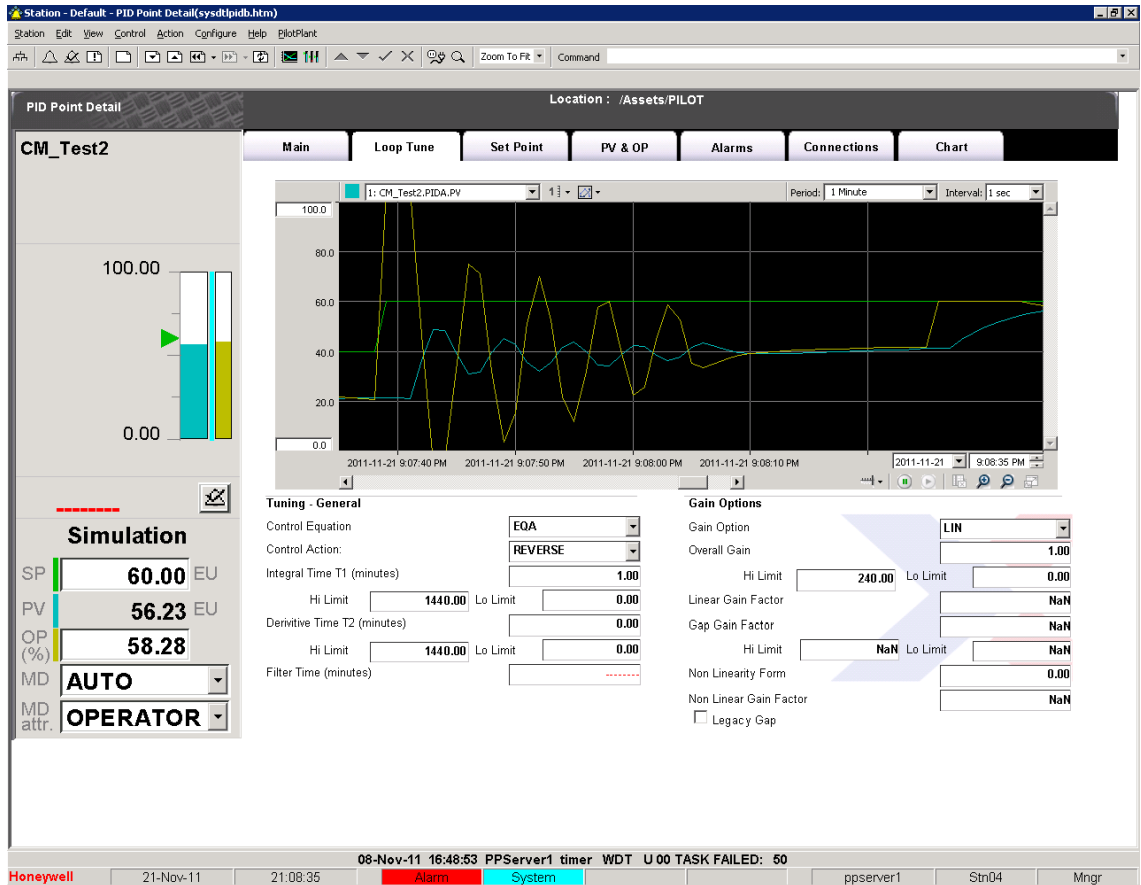


Figure 13: Server Display Showing a Simulated System

Chapter Four: Future Projects & Tasks

Discussed in this chapter are the projects and tasks that this thesis topic can lead into and provide a basis for.

4.1 SIM-C300

The main future task would be to explore the errors that were encountered when implementing a SIM-C300 into experion1. This however may require outside consultants who are already experienced in the use of Experion PKS. Following on from this, more than one SIM-C300 should be implemented because in a real industrial setting, there are usually a host of C300 controllers. This however may depend on licensing factors more than anything else. In the course of this thesis project, up to five SIM-C300s were able to be implemented at Murdoch University's Pilot Plant Facility.

4.2 Implementation into ISCE Facility

With further research and continuing development, an Experion PKS platform should be implemented into the ICSE facility. By doing so, the current setup of PLCs can be utilised to create a system. This would involve the PLCs communicating to a database of sort which will then be read by the SIM-C300 to perform actions based on the information received. In this aspect, part or all of real industrial process can be simulated.

Chapter Five: Conclusion

Over the course of the period when attempting to fulfil the objectives of thesis project, many challenges were encountered and overcome. High expectations were established before the project had commenced, however as development progressed, setbacks were encountered. This therefore meant that goals had to be lowered to a more realistic and achievable level given the restricted time available. Although it can be said that the route taken was not the path initially planned, objectives were met nonetheless.

With every challenge that presented itself, ideas and solutions were sought autonomously to overcome the issue all in the hopes of achieving the objectives set out. Much, if not all of Experion PKS was alien when the thesis project had commenced but with constant effort, it has become much less so. When each target objective was met, it was a defining moment for the project. This then meant further and continuing development could take place so much so that it has reached the stage that it is at now. On a personal level, time management was a personal skill that was developed as the project progressed along with problem solving skills.

At this current stage of the project, much of the background knowledge and initial development have been covered. It is therefore ideal to continue once this stage comes to a successful conclusion.

The Experion PKS software control platform is very much a complex and sophisticated engineering tool. Therefore, being given the opportunity to experiment with the software at the most basic and initial stages was very much a privilege. As a result, this thesis has laid the foundation and basis for its continued development along with other related future prospective projects that may arise.

Bibliography

- Bailey, D., & Wright, E. (2003). *Practical SCADA for Industry*. Burlington: Newnes.
- Boyer, S. A. (1999). *SCADA: Supervisory Control and Data Acquisition* (2nd ed.). Research Triangle Park: Instrument Society of America.
- Clarke, G., Reynders, D., & Wright, E. (2004). *Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems*. Burlington: Newnes.
- Honeywell International Inc. (2007). Honeywell Knowledge Builder (Version 4.5.0.18).
- Honeywell International Inc. (No Date Given). *About Experion*. Retrieved August 15, 2011, from Honeywell Process Solutions: <http://hpsweb.honeywell.com/Cultures/en-US/Products/Systems/ExperionPKS/default.htm>
- Honeywell International Inc. (No Date Given). *C300 Controller*. Retrieved October 29, 2011, from Honeywell: <https://www.honeywellprocess.com/en-US/explore/products/control-monitoring-and-safety-systems/integrated-control-and-safety-systems/experion-pks/controllers/Pages/C300.aspx>
- Honeywell International Inc. (No Date Given). *Experion Platform Infrastructure*. Retrieved August 15, 2011, from Honeywell Process Solutions: <http://hpsweb.honeywell.com/Cultures/en-US/Products/Systems/ExperionPKS/PlatformInfrastructure/default.htm>
- Honeywell International Inc. (No Date Given). *Experion Process Knowledge System (PKS)*. Retrieved August 15, 2011, from Honeywell Process Solutions: <http://hpsweb.honeywell.com/Cultures/en-US/Products/Systems/ExperionPKS/ExperionPlatformOverview/default.htm>
- Honeywell International Inc. (No Date Given). *Process Controllers*. Retrieved August 15, 2011, from Honeywell Process Solutions: <http://hpsweb.honeywell.com/Cultures/en-US/Products/Systems/ExperionPKS/ControlExecutionEnvironment/default.htm>
- Honeywell Pacific Automation College. (2009). *Experion PKS Server Configuration & Engineering R310 Rev 03.1 (Dec 09) Book 1 & 2*. North Ryde: Honeywell International Inc.
- Honeywell Pacific Automation College. (2009). *Experion PKS Server Configuration & Engineering R310 Rev 03.1 (Dec 09) Book 3 & 4*. North Ryde: Honeywell International Inc.
- Honeywell Pacific Automation College. (2009). *Experion PKS Server Configuration & Engineering R310 Rev 03.1 (Dec 09) Book 5 & 6*. North Ryde: Honeywell International Inc.

- Honeywell Pacific Automation College. (2010). *Experion PKS R310 Control Execution Environment C200/C300/ACE R310 Rev 04 (Apr 10) Book 1*. North Ryde: Honeywell International Inc.
- Honeywell Pacific Automation College. (2010). *Experion PKS R310 Control Execution Environment C200/C300/ACE R310 Rev 04 (Apr 10) Book 2*. North Ryde: Honeywell International Inc.
- Honeywell Pacific Technical Education Centre. (2007). *Experion PKS Graphics Design & Building Student Guide*. North Ryde: Honeywell International Inc.
- Hopkinson, E. (2010). *Murdoch University Pilot Plant Advanced Control Technology Upgrade*. Perth: Murdoch University.
- Ogunnaike, B. A., & Ray, W. H. (1994). *Process Dynamics, Modeling and Control*. New York: Oxford University Press.
- Punch, A. (2009). *Implementation of Advanced Control Technology in the Murdoch University Pilot Plant*. Perth: Murdoch University.

Appendix A

ENG460: Engineering Thesis
Academic Supervisor endorsement pro forma

I am satisfied with the progress of this thesis project and that the attached report is an accurate reflection of the work undertaken.

Signed:

Date:

Appendix B

A guide to Implementing a SIM-C300.



MURDOCH UNIVERSITY

PERTH, WESTERN AUSTRALIA

Appendix B: Guide to Implementing a SIM-C300

This document provides step-by-step instructions to implement simulated Honeywell C300 controllers.

Edwin Lum

Honeywell Experion for Teaching Purposes Thesis 2011

About This Guide

This document was written to assist in the process of implementing a simulated Honeywell C300 controller which is generally known as a SIM-C300. Broken down into three parts, the first two sections of this guide offers step-by-step instructions that will lead the user through creating a SIM-C300 along with an assortment of input and output modules. The last section looks at the final stages of actually implementing the controller for operation. Please read and understand the set of instructions before undertaking any work to avoid any confusion or mishaps.

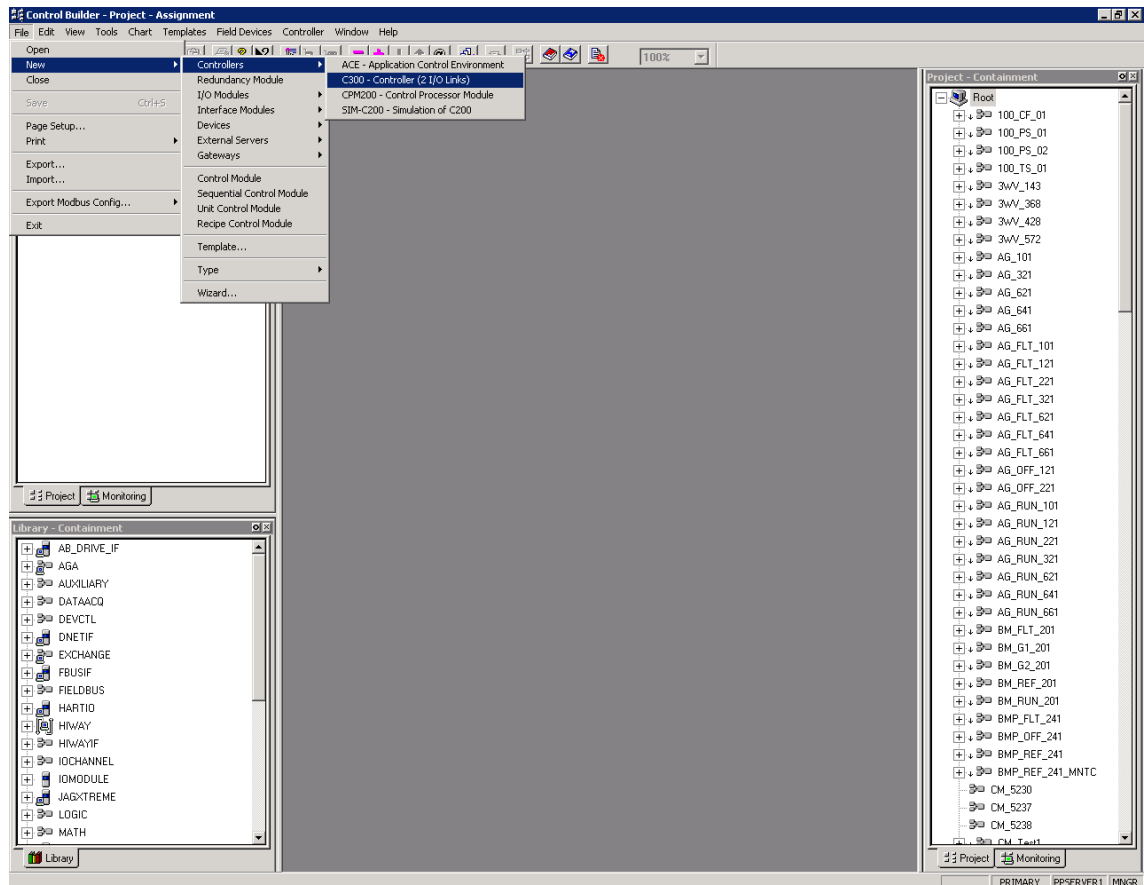
Section A: Creating a Simulated Honeywell C300 Controller

Section A – Step 1.

Enter Control Builder.

Section A – Step 2.

Go to File → New → Controllers → C300 – Controller (2 I/O Links).



Section A – Step 3.

Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- SIM_C300_X
- SIM_C300_Xitem

Check the Load to simulation Environment Box and fill in the Host IP Address field to be 127.0.0.1 or the localhost IP if known. The Host Name field should automatically be entered. If not, find out the localhost's name and enter it into the provided area. In this case, ppserver1.ad.murdoch.edu.au will be used. Now click OK.

SYSTEM:C300 Block, C300_5710 - Parameters [Project]

| Soft Failures | Server History | Server Displays | Control Confirmation | QVCS | Identification | | |
|---------------|----------------|-----------------|----------------------|----------------------|----------------|---------|---------|
| Main | System Time | Statistics | Peer Connections | Hardware Information | FTE | UDP/TCP | IP/ICMP |

Tag Name: SIM_C300_X

Item Name: SIM_C300_Xitem

Application Image Version: [Empty]

Controller Command: NONE

Network Address Configuration

Device Index: 0

Ethernet IP Address: 0.0.0.0

Redundancy Configuration

Module is redundant

Secondary Tag Name: [Empty]

State Information

Controller State: NOTLOADED

Redundancy Role: UNDEFINED

Synchronization State: ...

Battery State: OK

Soft Failures Present (See Soft Failures Tab for details)

Advanced Configuration

Alarming Enabled

GPS Time Source Enabled

Temperature High Alarm (degC): 80

CPU Free Low Alarm (%): 20

CPU Free Low Low Alarm (%): 10

Simulation Node Configuration

Load to simulation Environment

Host IP Address: 127 . 0 . 0 . 1

Host Name: ppserver1.ad.murdoch.edu.au

Simulation Node Operation

SIM Command: NONE

Simulation State: NONE

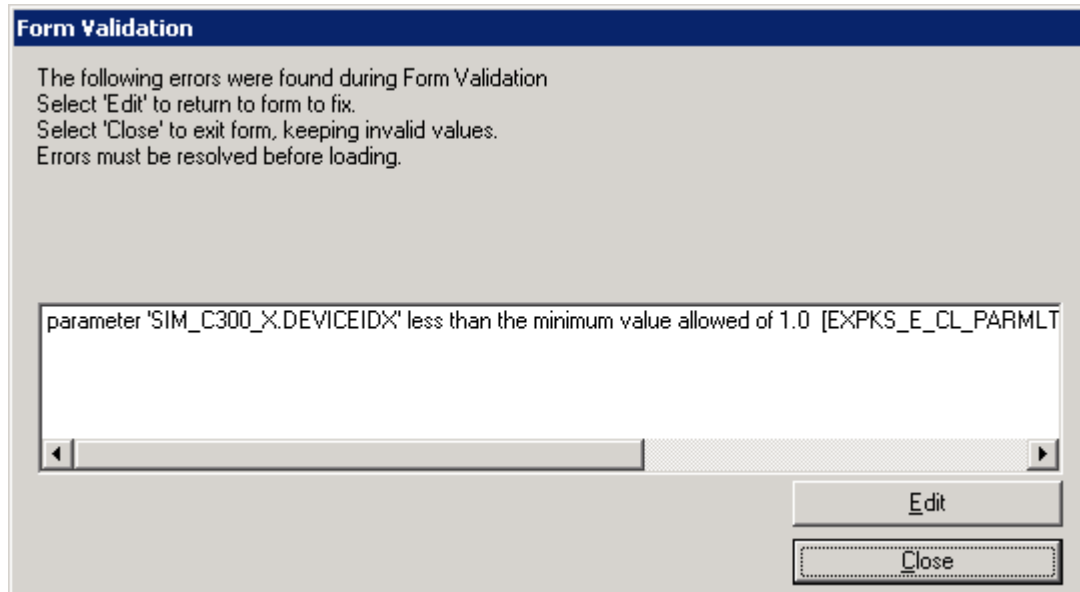
WIN32 Process Identifier: 0

Show Parameter Names

OK Cancel Help

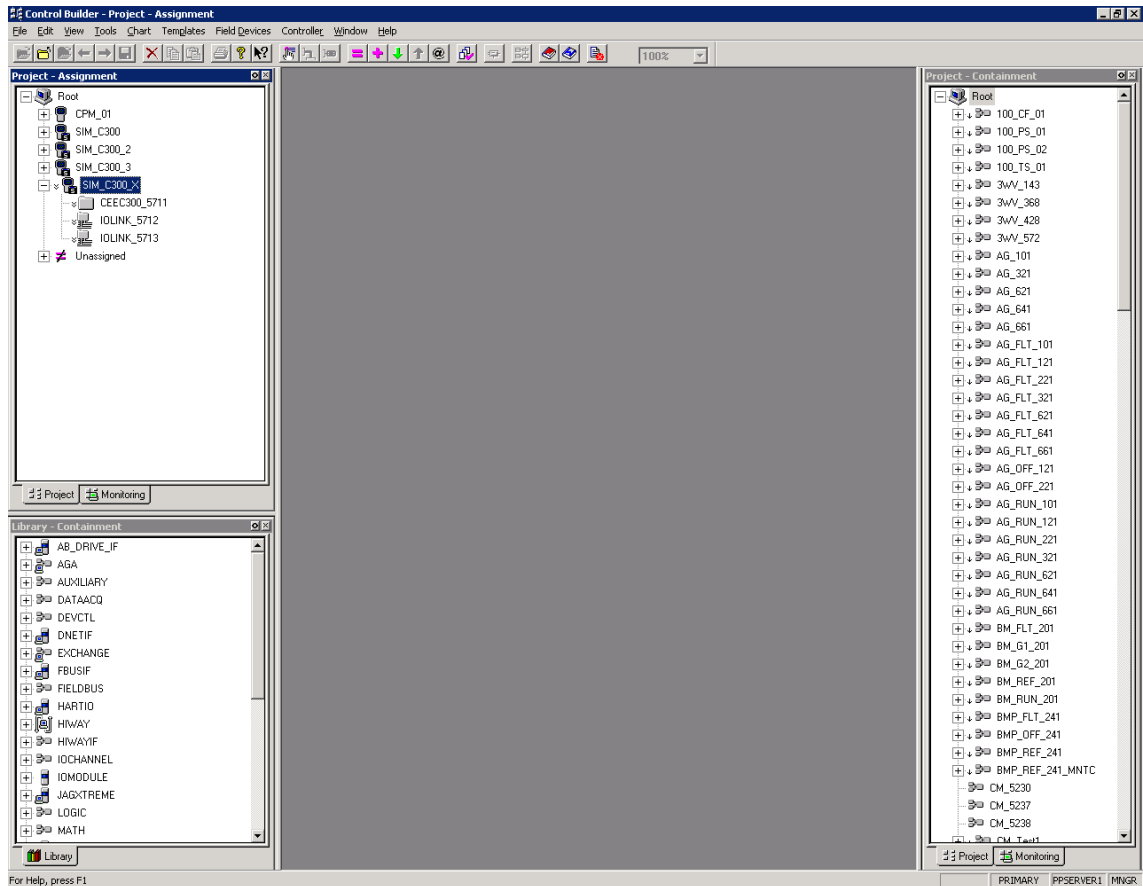
Section A – Step 4.

Upon seeing a Form Validation Window, click OK to continue.



Section A – Step 5.

Expand the newly created SIM-C300 in the Project tab by clicking on the + sign to the left of it.



Section A – Step 6.

Double click the Control Execution Environment (CEE) of the controller to open its Parameters window. Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- SIM_CEEC300_X
- SIM_CEEC300_Xitem

Keep all the other settings as default and click OK.

SYSTEM:CEEC300 Block, CEEC300_5711 - Parameters [Project]

Exchange Communications | Display Communications | Block Types Info | Server History | Server Displays | Control Confirmation | Identification

Main | Peer Configuration | Statistics | CPU Loading | CPU Overruns | Memory | Peer Communications

Tag Name: SIM_CEEC300_X

Item Name: SIM_CEEC300_Xitem

Base Execution Period: 50mS

Command/State

CEE Command: IDLE

CEE State: IDLE

User Lock for CEE Run: Operator

User Lock for CEE Idle: Supervisor

Program Access may command CEE from Idle to Run

Program Access may command CEE from Run to Idle

Alarm Info

In-Alarm Flag: OFF

Alarming Enabled

Enable Memory Limit Exceeded Alarm

Powerup Restart Settings

CEE State: IDLE

Warm Timeout:

Batch Events Settings

Batch Events Memory: Small

Time Info

Time Zone: 0

Daylight Savings Time

Year Format: YYYY

Weekday Format: 1Sunday

Simulation Info

Simulation State: NONE

Inhibit Notifications - CEE and Contents

Show Parameter Names

OK Cancel Help

Section A – Step 7.

Double click the first IOLINK to open its Parameters window. Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- Series_C_IOLINKX
- Series_C_IOLINKXitem

Now select the I/O Family type to be SERIES_C_IO_TYPE. Keep all the other settings as default and click OK.

SYSTEM:IOLINK Block, Series_C_IOLINKX - Parameters [Project]

Show Parameter Names

Section A – Step 8.

Exactly as before, but this time, double click on the second IOLINK to open its Parameters window. Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- PMIO_IOLINKX
- PMIO_IOLINKXitem

Now select the I/O Family type to be PM_IO_TYPE. Keep all the other settings as default and click OK.

SYSTEM:IOLINK Block, IOLINK_5713 - Parameters [Project]

Main | Memory Statistics | Statistics | I/O Link Status | I/O Status Summary | Server History | Server Displays | Identification

Tag Name: PMIO_IOLINKX

Item Name: PMIO_IOLINKXitem

Description:

I/O Family: PM_IO_TYPE

I/O Link Command: NONE

Priority IOMs: 0

I/O Link Cable Color: Violet

I/O Link Number: 2

I/O Link State: NOTLOADED

Simulation State: NONE

IOLINK Soft Fail Errors:

- Duplicate IOL Address
- IOL Channel A Failure
- IOL Channel B Failure
- IOL Maximum Errors Exceeded
- Not Active Supervisor
- IOLIM Daughter Card Soft Failure
- Partner I/F Not Visible On IOL
- Partner I/F Mismatch On IOL
- IOL Process Data Cycle Overruns
- UMS1 Diagnostic Exceeded Time Threshold
- UMS1 Diagnostic Overrun

Daughter Card Soft Failures:

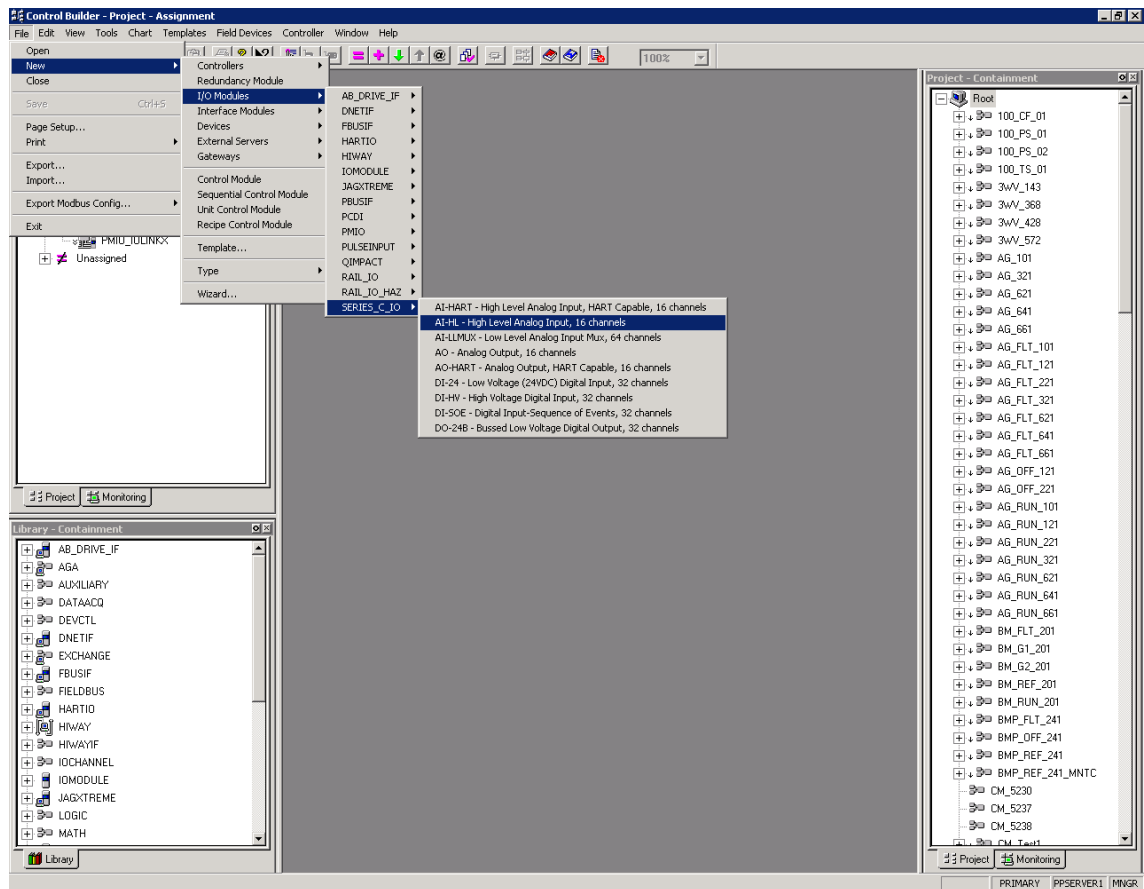
Show Parameter Names

OK Cancel Help

Section B: Creating Input & Output Modules

Section B – Step 1.

Go to File → New → I/O Modules → Series_C_IO → AI – HL High Level Analog Input, 16 channels.



Section B – Step 2.

Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

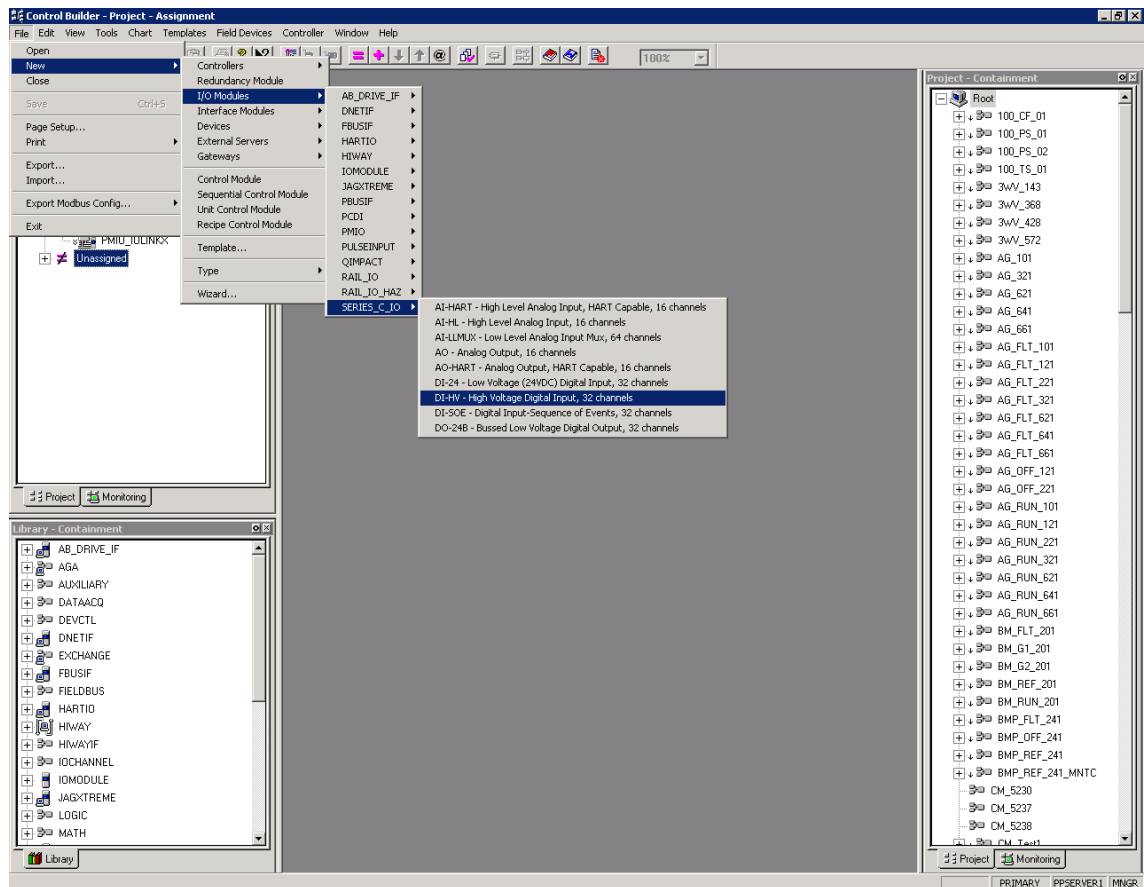
- C_AI_X
- C_AI_Xitem

In the IOM Number field, enter an unused number for an Input & Output Module (IOM). In this case, the digit 10 was used. Keep all the other settings as default and click OK.

The screenshot shows a software configuration window titled "SERIES_C_IO:AI-HL Block, AI_HL_5714 - Parameters [Project]". The window has a tabbed interface with four main tabs: "Server Displays", "Control Confirmation", "QVCS", and "Identification". The "Server Displays" tab is selected and contains several sub-tabs: "Main", "Status Data", "Maintenance", "Calibration", "Box Soft Failures", "Channel Soft Failures", and "Server History". The "Main" sub-tab is active, showing various configuration fields. The "Tag Name" field contains "C_AI_X", and the "Item Name" field contains "C_AI_Xitem". The "Module Type" is set to "High Level Analog Input, 16 chann". The "IOM Number" field contains "10". The "Execution State" is set to "Idle". The "Associated IOLINK" field is empty. There are checkboxes for "Database Valid" and "This IOM is redundant". The "IOM Partner A" and "IOM Partner B" sections show status, operation, and redundancy status fields. The "Command" field is set to "None". The "IOM Location" field is empty. The "Number of Channels" is set to "16". The "I/O Link Scan Rate" is set to "250_ms". The "I/O Link Cable Color" is set to "Gray". At the bottom of the window, there is a checkbox for "Show Parameter Names" and three buttons: "OK", "Cancel", and "Help".

Section B – Step 3.

Go to File → New → I/O Modules → Series_C_IO → DI – HV High Voltage Digital Input, 32 channels.



Section B – Step 4.

Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- C_DI_X
- C_DI_Xitem

In the IOM Number field, enter an unused number for an IOM. In this case, the digit 11 was used. Keep all the other settings as default and click OK.

SERIES_C_IO:DI-HV Block, DI_HV_5731 - Parameters [Project]

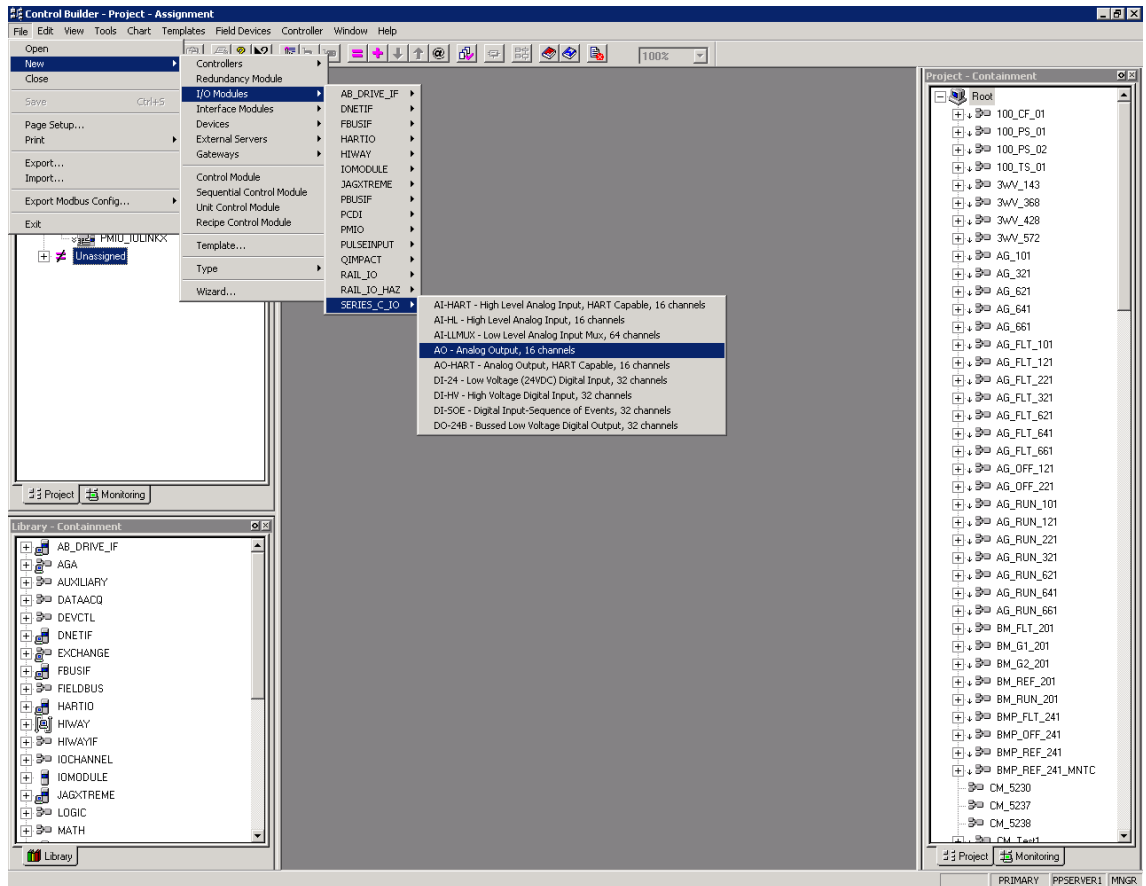
| Control Confirmation | | QVCS | | Identification | | |
|--------------------------------------|-------------------------------------|-------------|--|-----------------------|----------------|-----------------|
| Main | Status Data | Maintenance | Box Soft Failures | Channel Soft Failures | Server History | Server Displays |
| Tag Name | C_DI_X | | IOM Location | | | |
| Item Name | C_DI_Xitem | | Frequency 60/50Hz | 60Hz | | |
| Module Type | High Voltage Digital Input, 32 chan | | Number of Channels | 32 | | |
| Description | | | I/O Link Scan Rate | 250_ms | | |
| IOM Number | 11 | | I/O Link Cable Color | Gray | | |
| Execution State | Idle | | <input type="checkbox"/> This IOM is redundant | | | |
| Associated IOLINK | | | IOM Partner A | IOM Partner B | | |
| <input type="radio"/> Database Valid | | | Status | Status | | |
| Auto Synchronization State | ... | | Operation | Operation | | |
| | | | Redundancy Status | Redundancy Status | | |
| Command | None | | | | | |

Show Parameter Names

OK Cancel Help

Section B – Step 5.

Go to File → New → I/O Modules → Series_C_IO → AO – Analog Output, 16 channels.



Section B – Step 6.

Fill in the Tag Name and Item Name fields as desired.

Suggested names are provided below (Where X should be replaced by a number):

- C_AO_X
- C_AO_Xitem

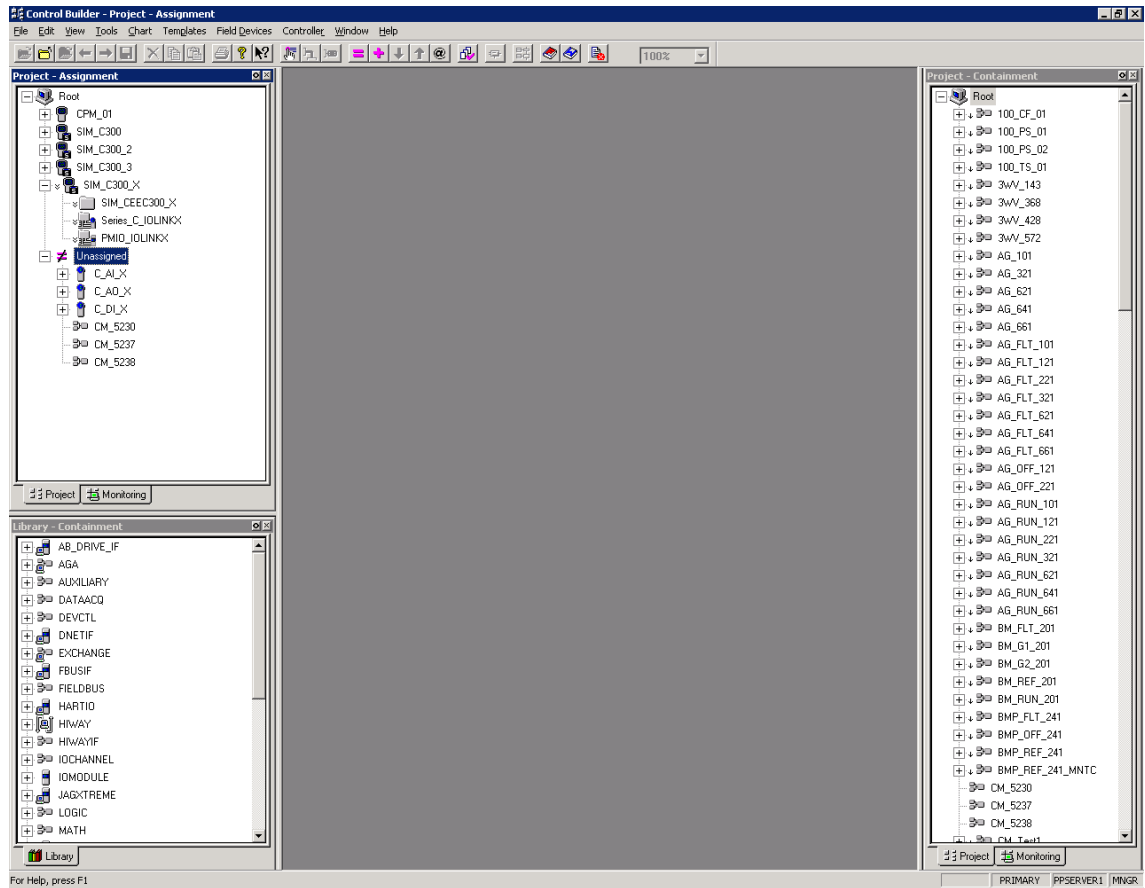
In the IOM Number field, enter an unused number for an IOM. In this case, the digit 12 was used. Keep all the other settings as default and click OK.

The screenshot shows a software configuration window titled "SERIES_C_IO:AO Block, AO_5764 - Parameters [Project]". The window is divided into several sections:

- Server Displays:** Includes sub-tabs for Main, Status Data, Maintenance, Calibration, Box Soft Failures, Channel Soft Failures, and Server History.
- Control Confirmation:** Contains fields for Tag Name (C_AO_X), Item Name (C_AO_Xitem), Module Type (Analog Output, 16 channels), Description, IOM Number (12), Execution State (Idle), Associated IOLINK, and a "Database Valid" radio button.
- QVCS:** Contains fields for IOM Location, Number of Channels (16), I/O Link Scan Rate (1S), and I/O Link Cable Color (Gray).
- Identification:** Contains a checkbox for "This IOM is redundant" and two sections for "IOM Partner A" and "IOM Partner B", each with Status, Operation, and Redundancy Status fields.
- Command:** A dropdown menu set to "None".
- Footer:** Includes a "Show Parameter Names" checkbox and "OK", "Cancel", and "Help" buttons.

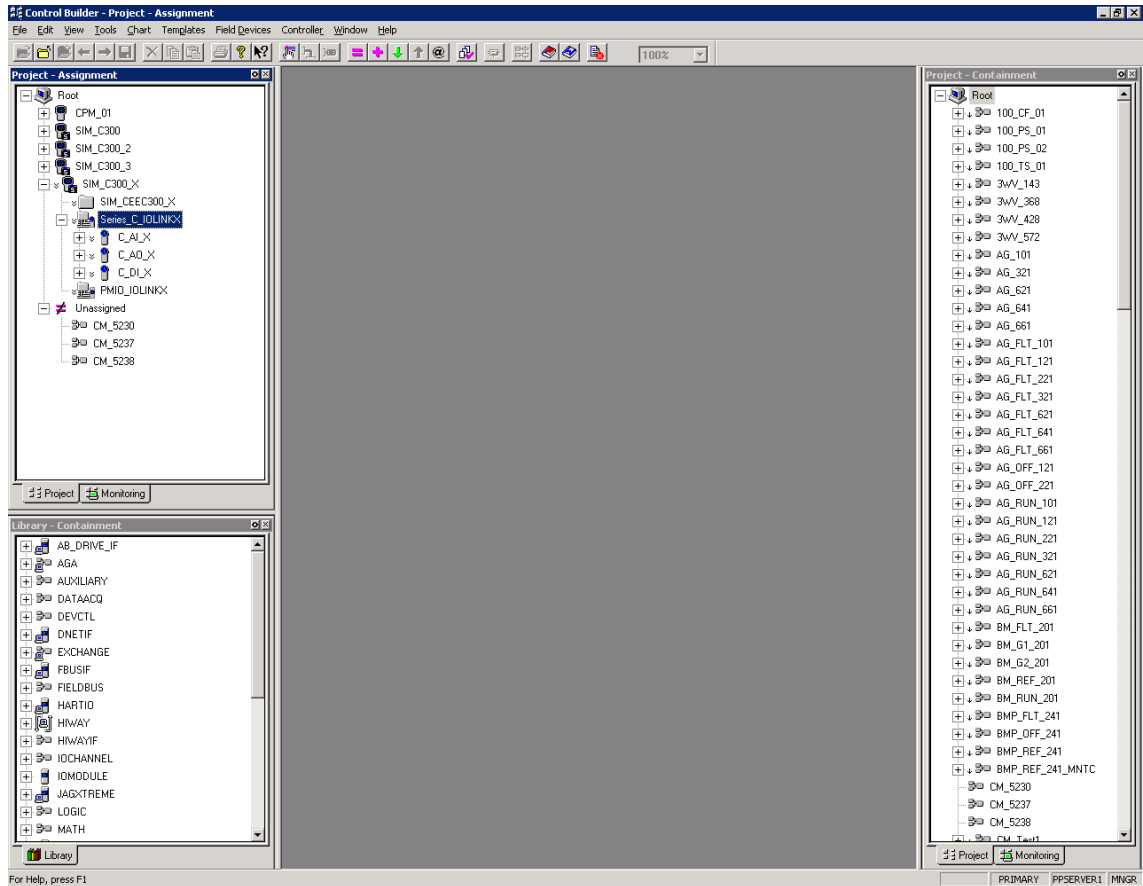
Section B – Step 7.

In the Project tab, select and expand the Unassigned tree list.



Section B – Step 8.

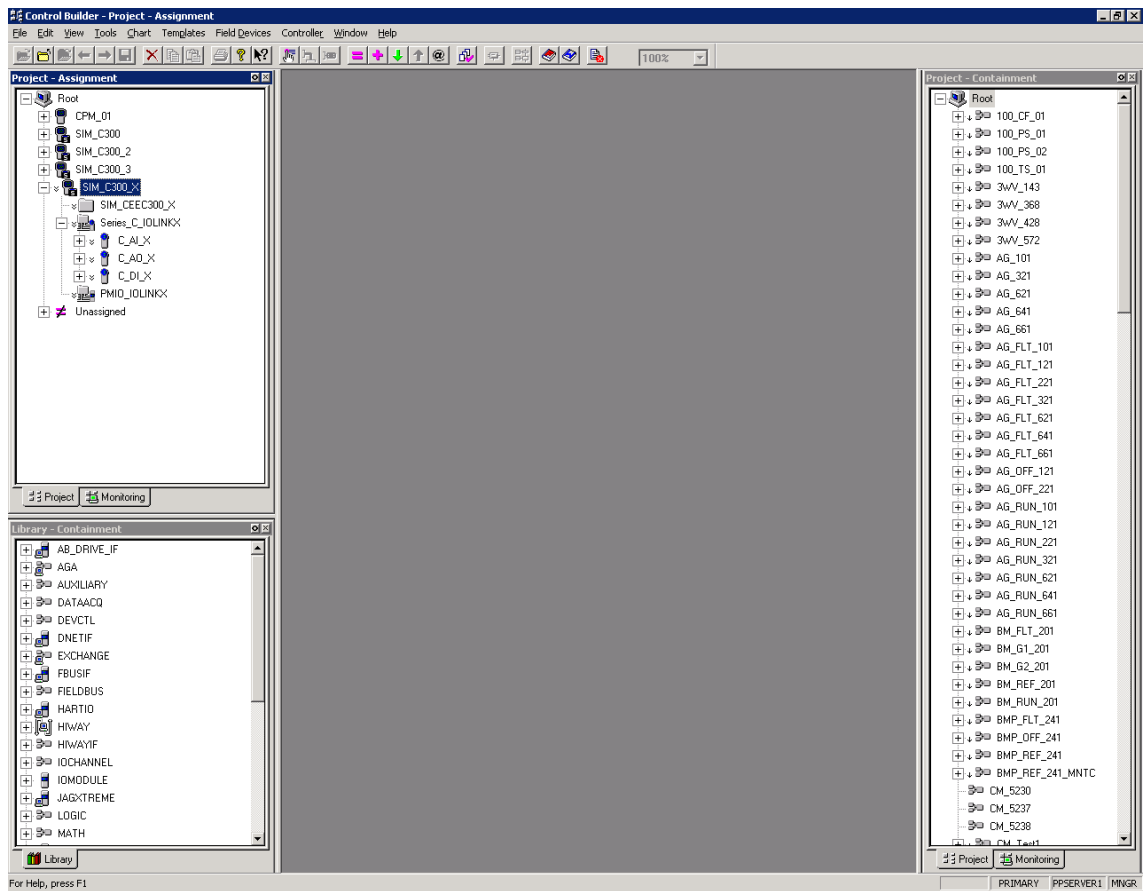
Drag and drop the three created IOMs to the Series_C_IOLINK to assign it. Note that each IOM has a number of channels which it holds inside. Each of these channels are preconfigured to belong to that specific IOM and can be used for programming at a later stage.



Section C: Implementing the Created SIM-C300 for Operation

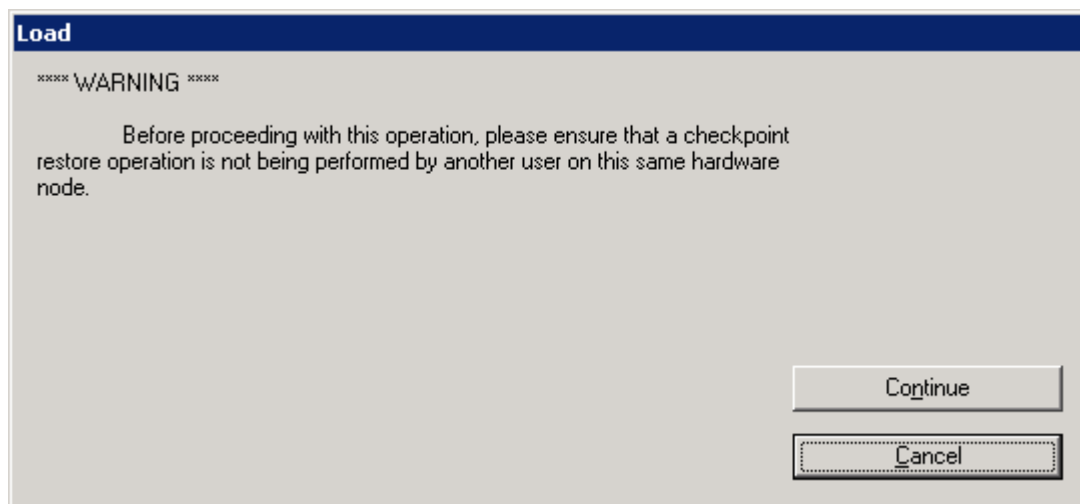
Section C – Step 1.

Select the SIM_C300_X in the Project tab and click the green downwards pointing arrow on the tool bar to download.



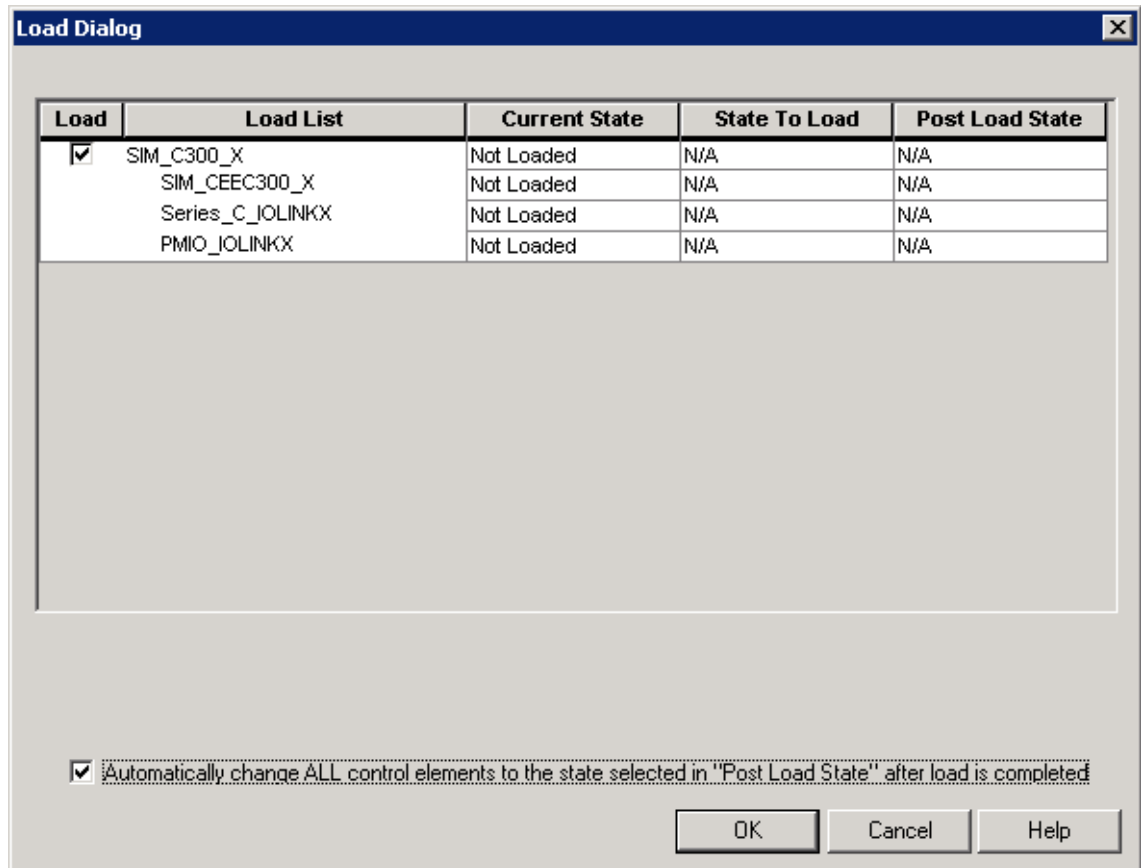
Section C – Step 2.

A Load window should appear. Click Continue.



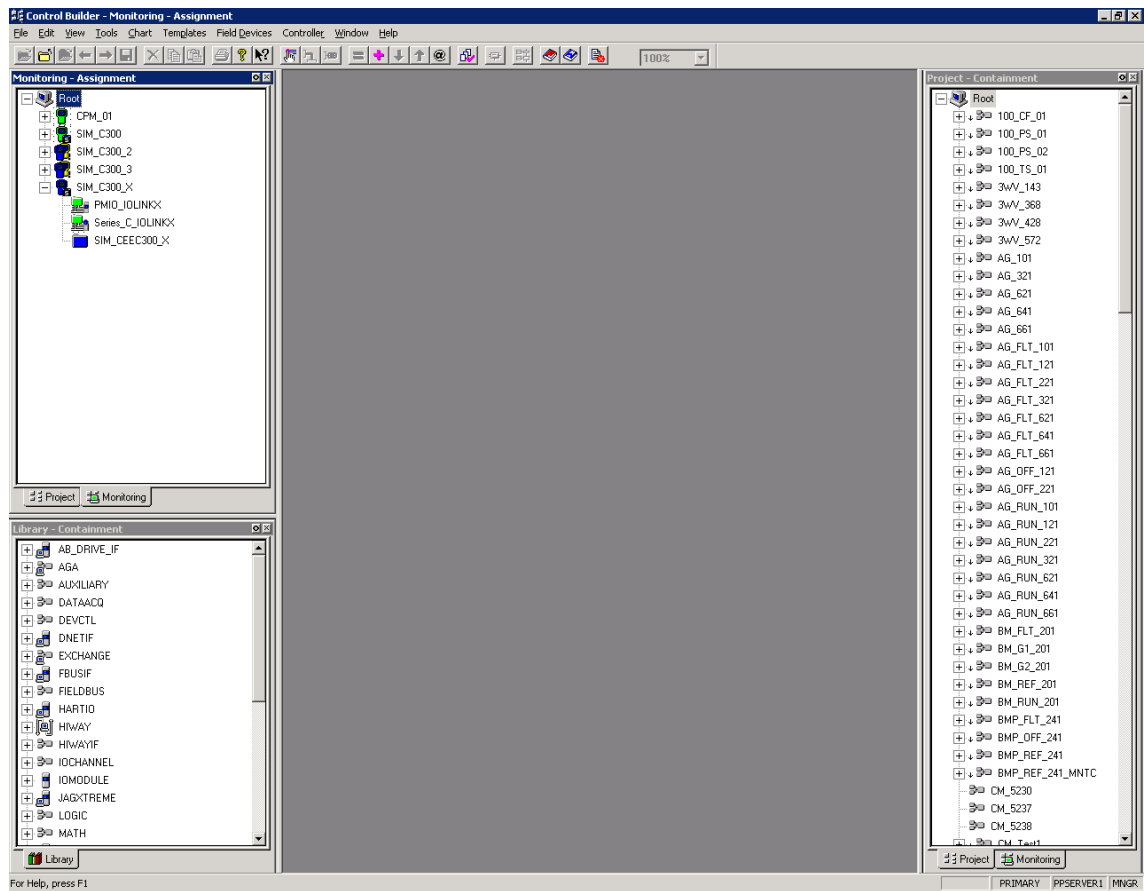
Section C – Step 3.

Upon seeing a Load Dialog window, check the box “Automatically change ALL control elements to the state selected in “Post Load State” after load is completed” and click OK.



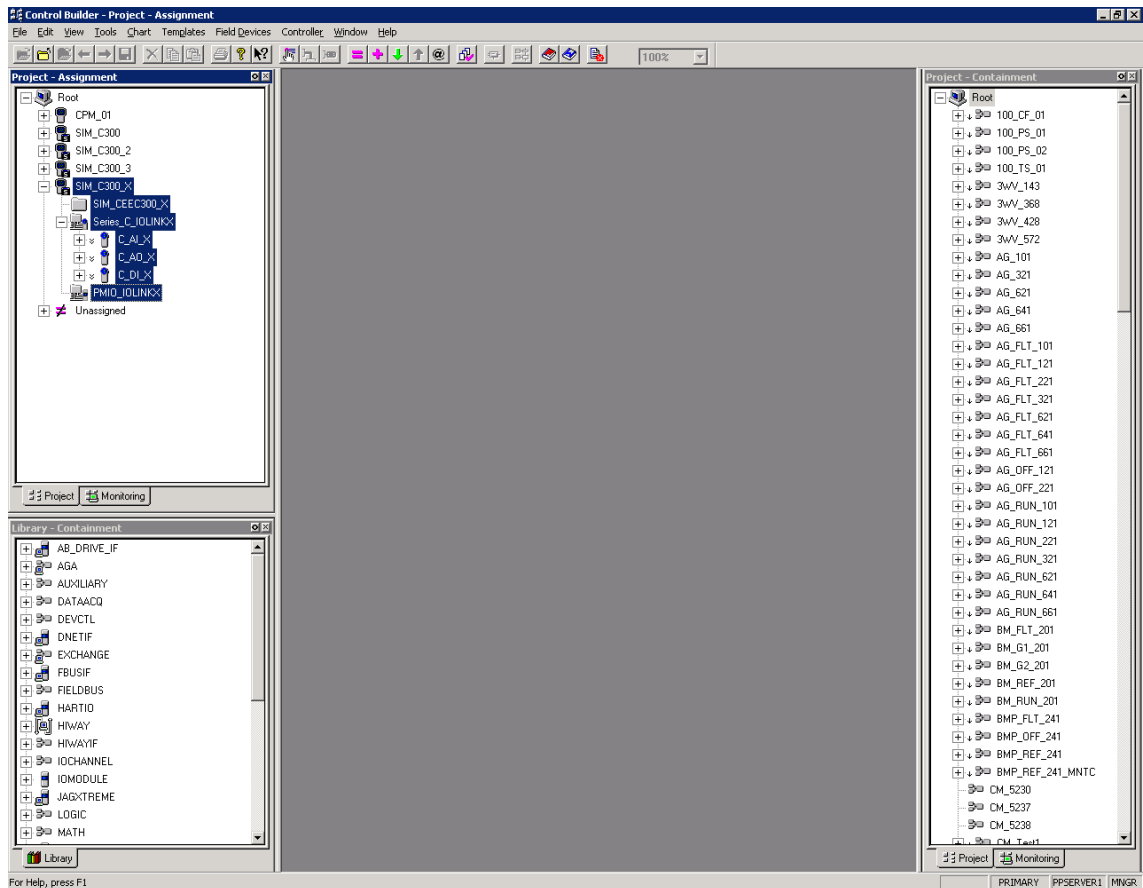
Section C – Step 4.

Now click the Monitoring tab and take note and check that the SIM_C300_X has been loaded but as of yet, is inactive as indicated by the blue icon.



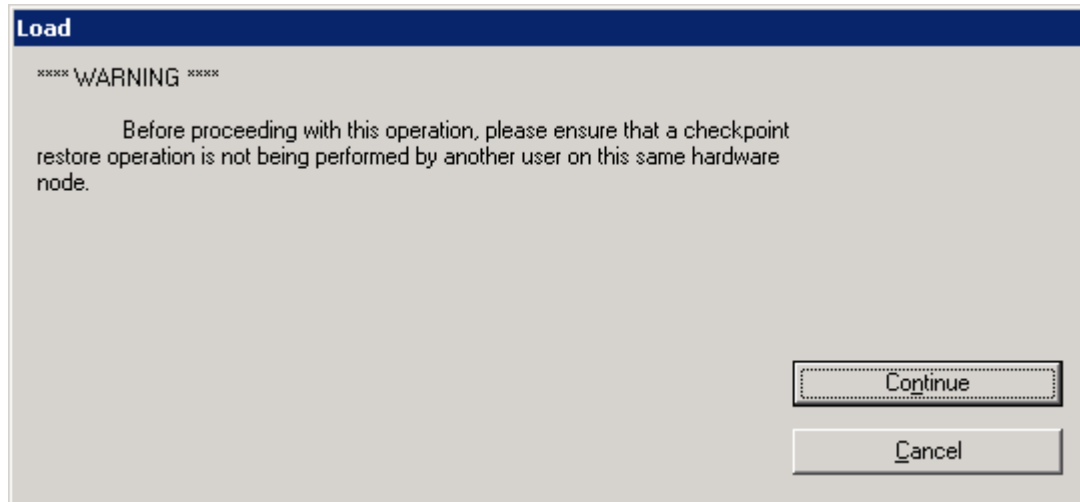
Section C – Step 5.

Click back into the Project tab to finish off the IOMs. IOLINKs are automatically loaded when a SIM-C300 is loaded but unfortunately, do not load the IOMs. Therefore, all IOMs must be loaded separately. In the Project tab, expand the SIM_C300_X and the Series_C_IOLINKX trees. Select all or just the IOMs and click on the green downwards pointing arrow on the tool bar to download.



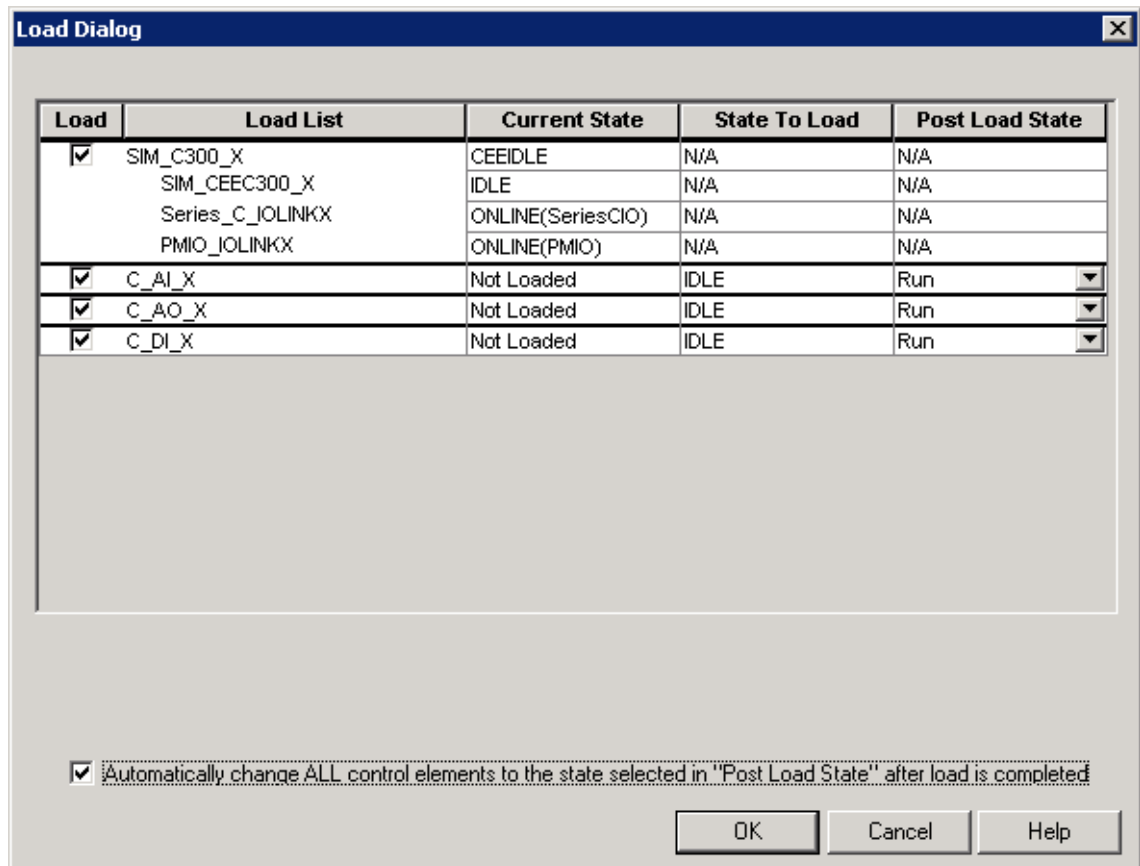
Section C – Step 6.

Click Continue when the Load window appears.



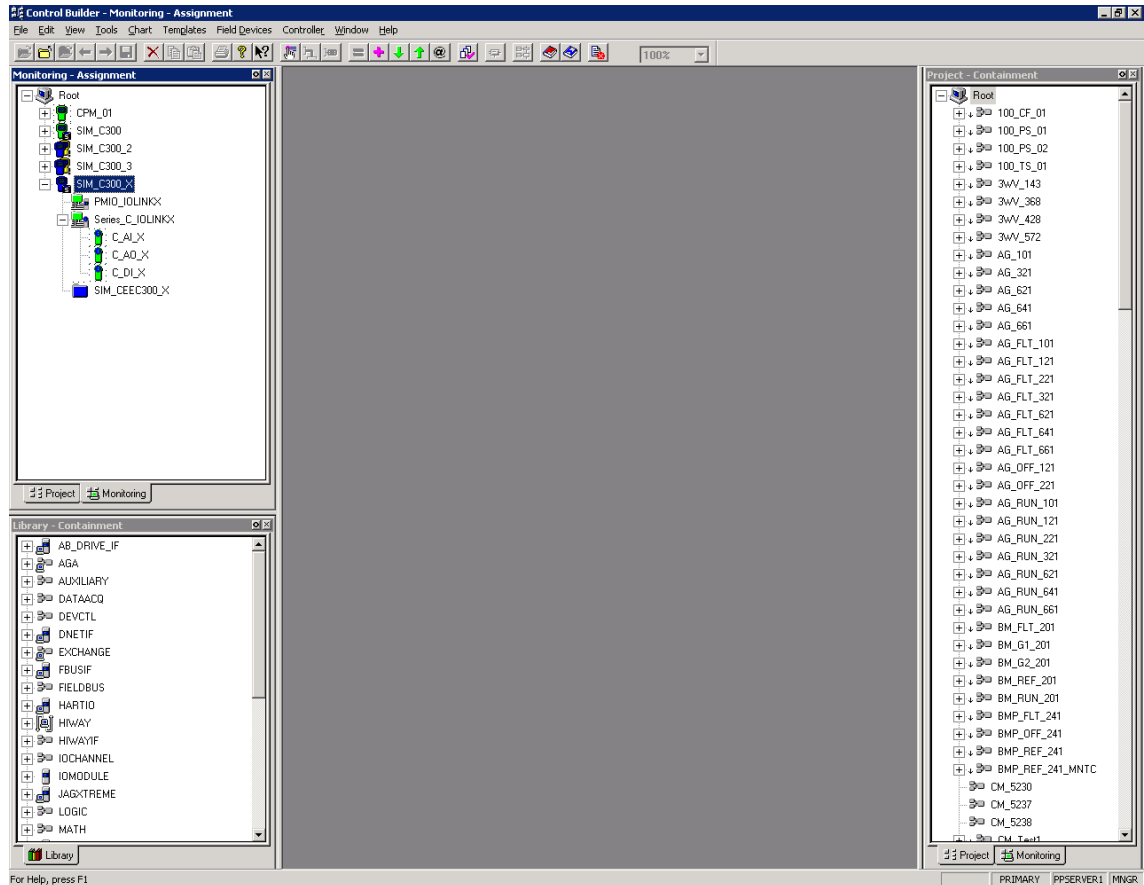
Section C – Step 7.

Upon seeing the Load Dialog window for the second time, check the box “Automatically change ALL control elements to the state selected in “Post Load State” after load is completed” and click OK.



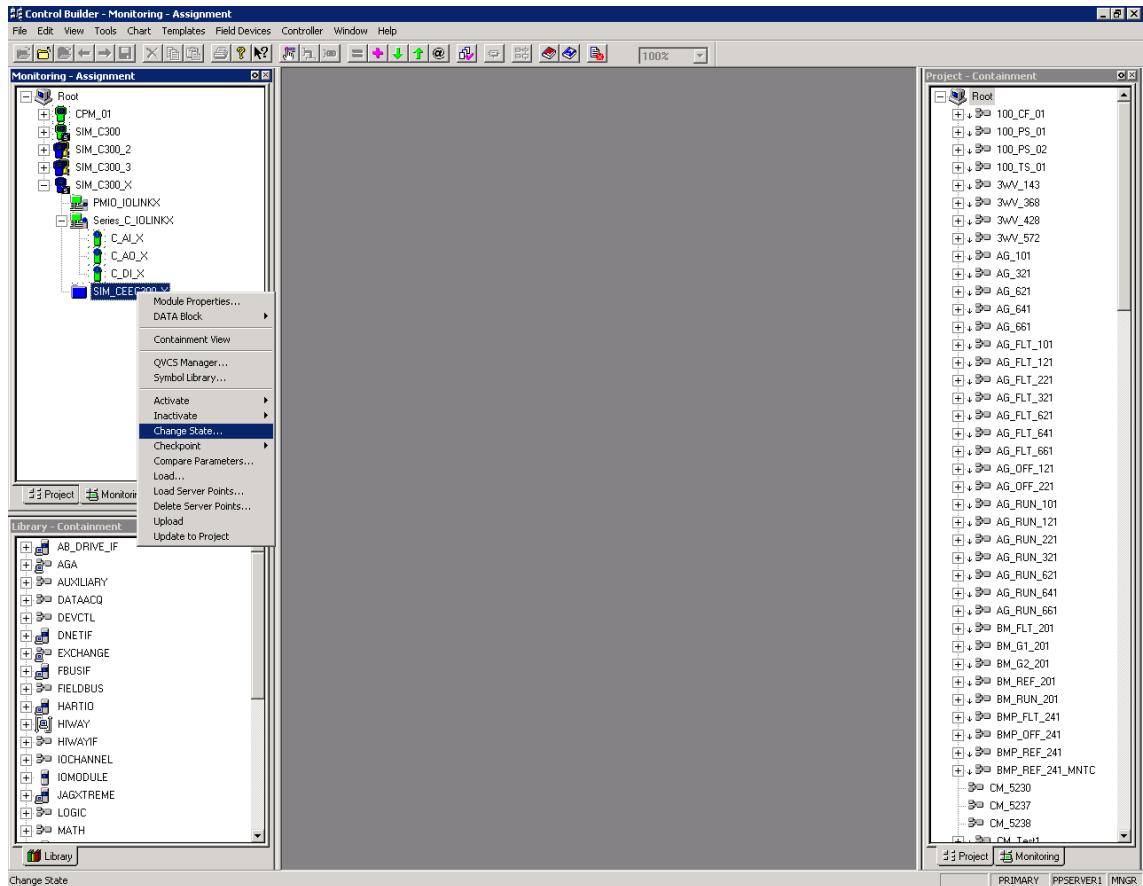
Section C – Step 8.

Switch over to the Monitoring tab and expand the SIM_C300_X and the Series_C_IOLINKX. It can be observed that the IOMs have now been loaded but the CEE at this point is still not active.



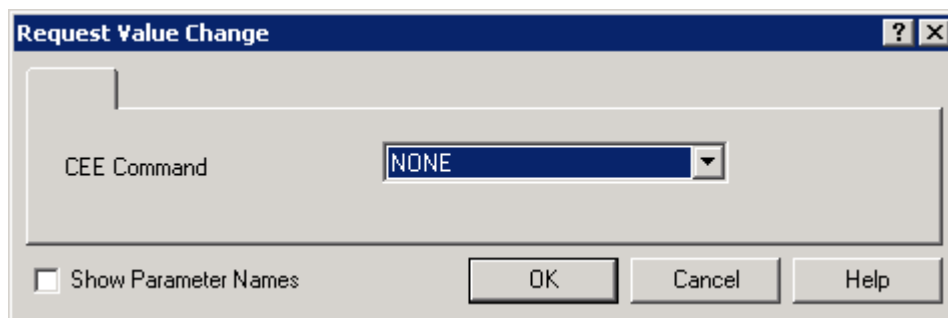
Section C – Step 9.

To activate the CEE, right click on it and select the option to Change State.



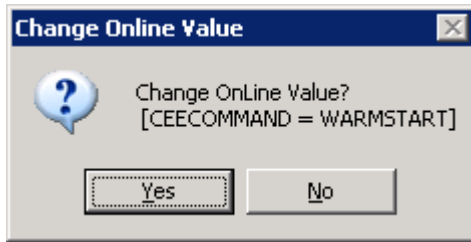
Section C – Step 10.

A Request Value Change window will then open. In the CEE Command of this window, change the value to WARMSTART.



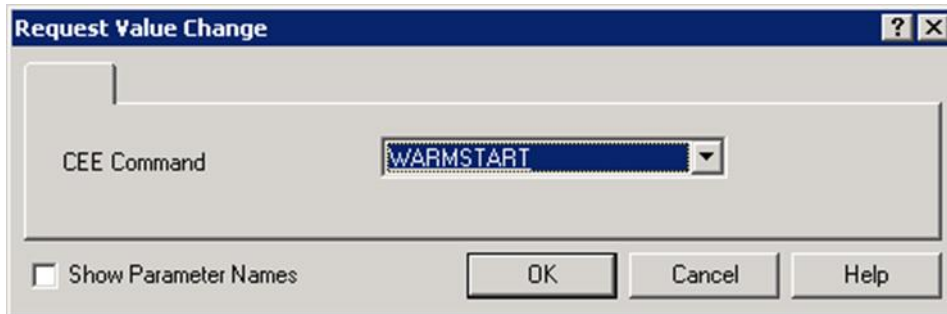
Section C – Step 11.

The Change Online Value window will then prompt for confirmation. Select Yes.



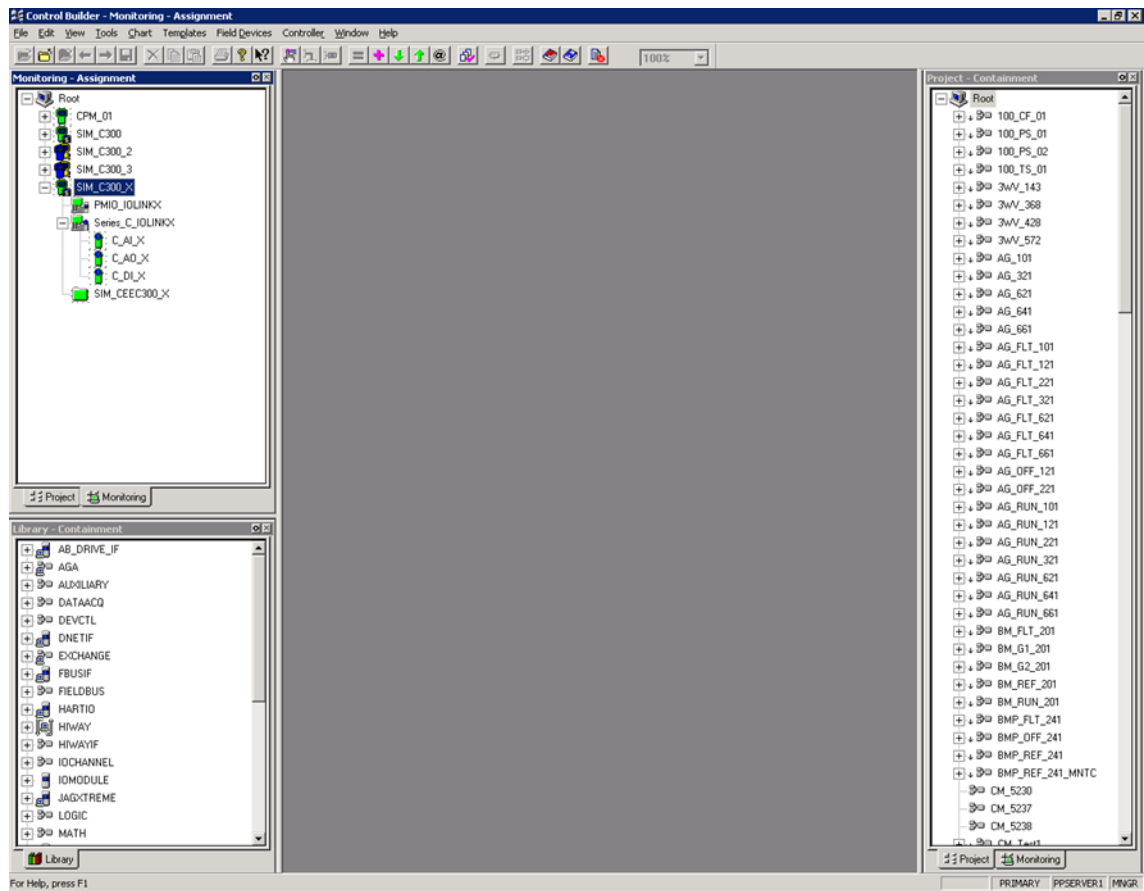
Section C – Step 12.

Back in the Request Value Change window, click OK.



Section C – Step 13.

Observe that the SIM-C300 in the monitoring tab is now active and in full operation as indicated by the green icons.



Appendix C

A Guide to programming a SIM-C300.



MURDOCH UNIVERSITY

PERTH, WESTERN AUSTRALIA

Appendix C: Guide Programming a SIM-C300

This document provides step-by-step instructions to program simulated Honeywell C300 controllers.

Edwin Lum

Honeywell Experion for Teaching Purposes Thesis 2011

About this Guide

This document was written to assist in the process of programming a simulated Honeywell C300 controller which is generally known as a SIM-C300. Offering step-by-step instructions, this guide will lead the user through creating Control Modules and provide insight into general programming techniques.

Instructions

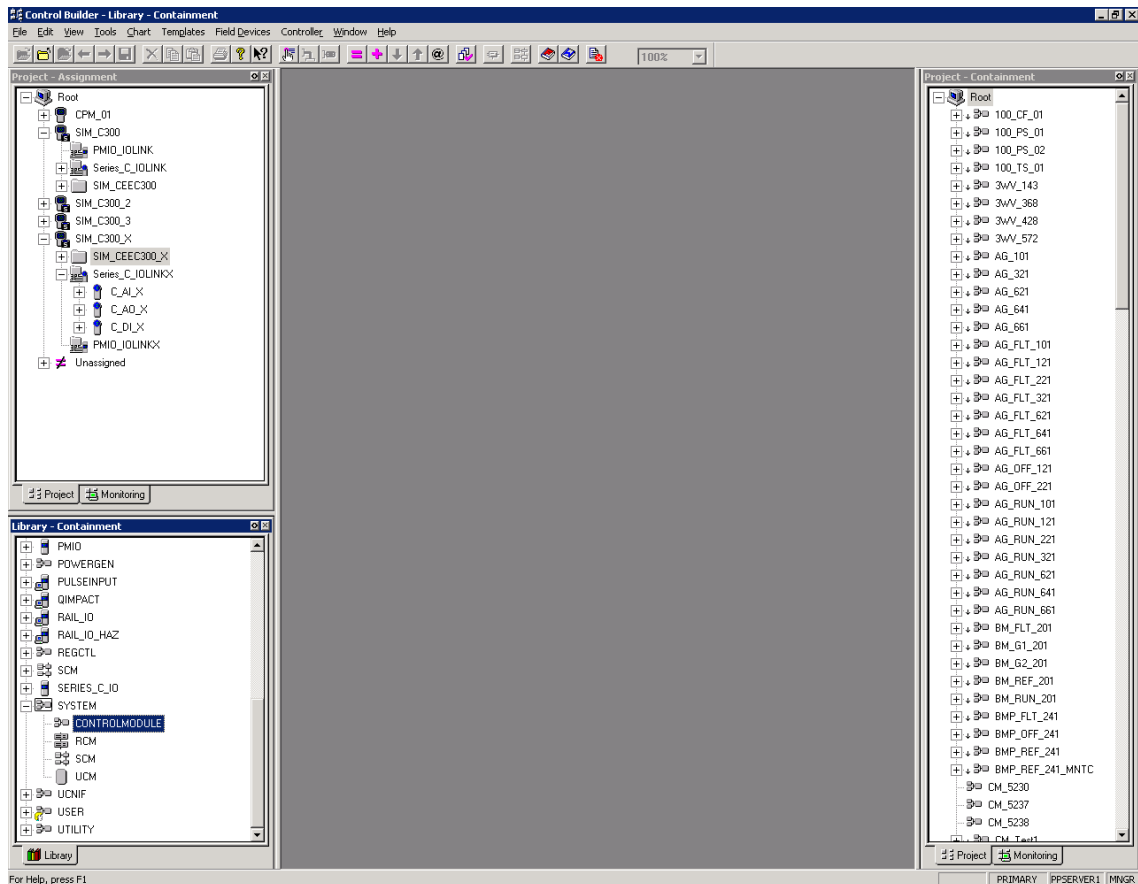
Please read and understand the set of instructions before undertaking any work to avoid any confusion or mishaps.

Step 1.

All programs are written inside of Control Modules (CMs) which are assigned to a SIM-C300's CEE. Therefore a CM must be firstly created and assigned to a CEE before coding. Open Control Builder and expand a SIM-C300 tree in the Project tab.

Step 2.

In the Library tab, scroll to find SYSTEM. Expand the SYSTEM tree and select CONTROL MODULE. Drag and drop the CONTROLMODULE into the CEE of the SIM-C300 to assign it.



Step 3.

Upon In the Name New Function Block(s)... dialog box that appears, change the Tag Names and Item Names in the Destination fields. Suggested names are provided below (Where X should be replaced by a number):

- CMX
- CMXitem

Click Finish to continue.

Name New Function Block(s)...

| Tag Names | | Item Names | | |
|-----------|---------|-------------|--------|-------------|
| | Source | Destination | Source | Destination |
| 1 | CM_5782 | CMX | | CMXitem |

Change the name in the destination column to the new desired name or accept the default.

Find/Replace...

< Back Finish Cancel Help

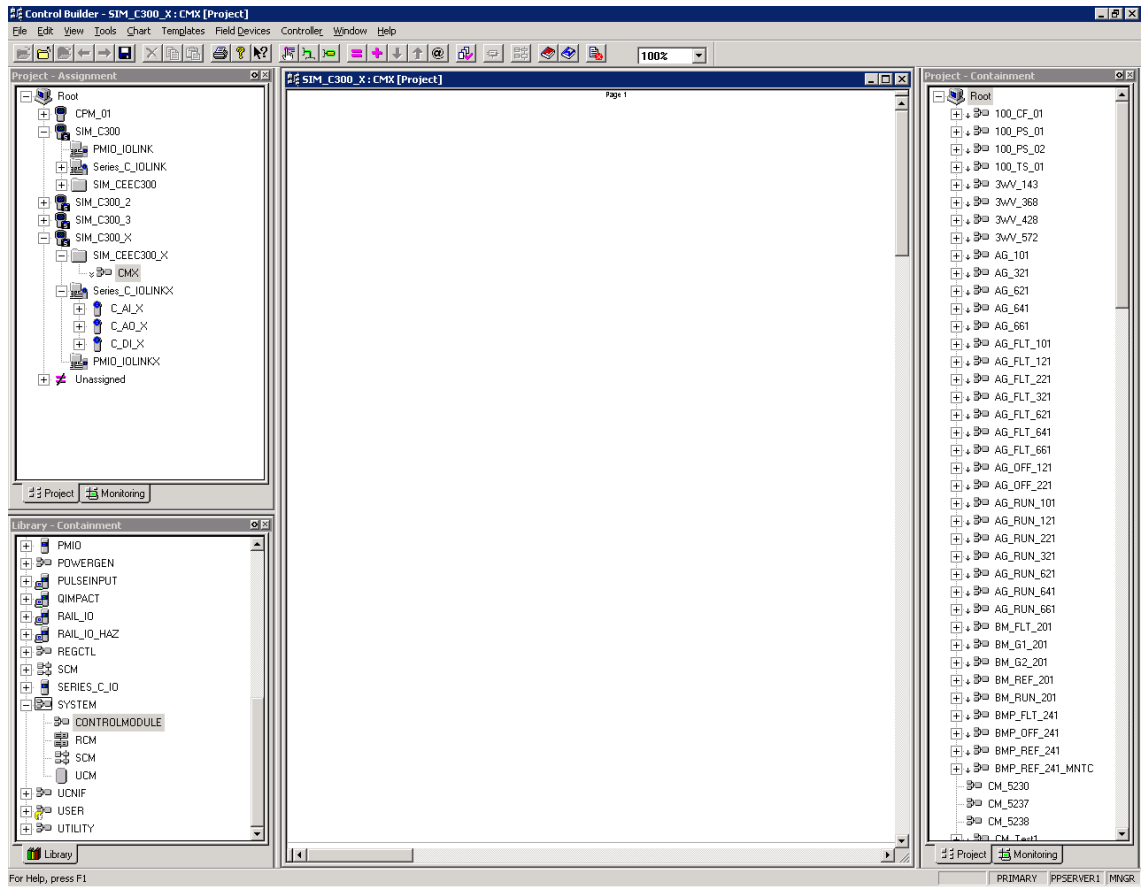
Step 4.

Right click on the newly created CM and select the Module Properties option. In the Parent Asset field, choose PILOT and then click OK.

The screenshot shows a software configuration window titled "SYSTEM:CONTROLMODULE Block, CMX - Parameters [Project]". The window has a tabbed interface with the following tabs: Projected Parameters, Block Pins, Configuration Parameters, Monitoring Parameters, Block Preferences, and Template Defining. The "Main" sub-tab is active, showing various configuration fields. The "Parent Asset" field is set to "PILOT". Other fields include Tag Name (CMX), Item Name (CMXitem), Description, Engr Units, Keyword, Execution Period (DEFAULT), Execution Order in CEE (10), Execution Phase (-1), CEE Restart Option (ALWAYS COLD), Enable Alarming Option (checked), Journal Only Option (unchecked), SCM Option (NONE), SCM Name, Mode Attribute Reference, Execution Order in LINK (10), Stale Count (3), and FF Execution Period (1s). At the bottom, there is a checkbox for "Show Parameter Names" and buttons for "OK", "Cancel", and "Help".

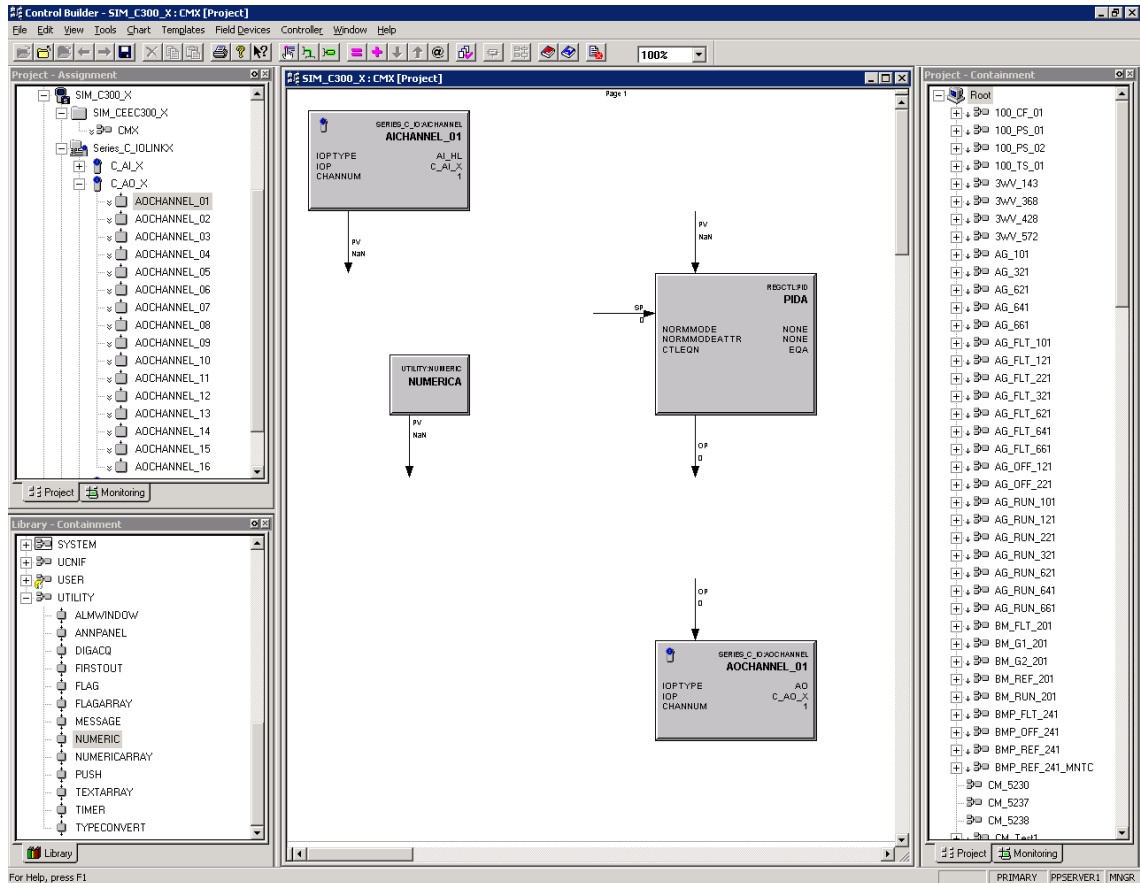
Step 5.

Expand the SIM-C300's CEE and double click the CM. It should open the programming environment as shown below.



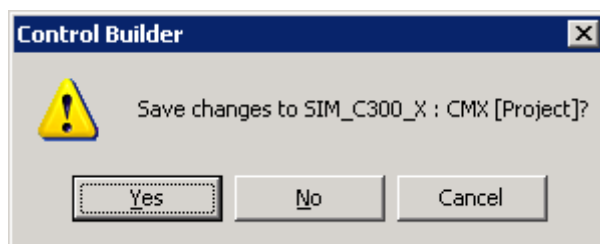
Step 6.

To start coding, drag and drop Function Blocks (FBs) from the Library tab into the CM programming environment. Preconfigured channels in the Input & Output Modules (IOMs) can be inserted into the environment using the same drag and drop method. The channels for C_AI_X can be seen in the illustration below along with FBs in the CM.



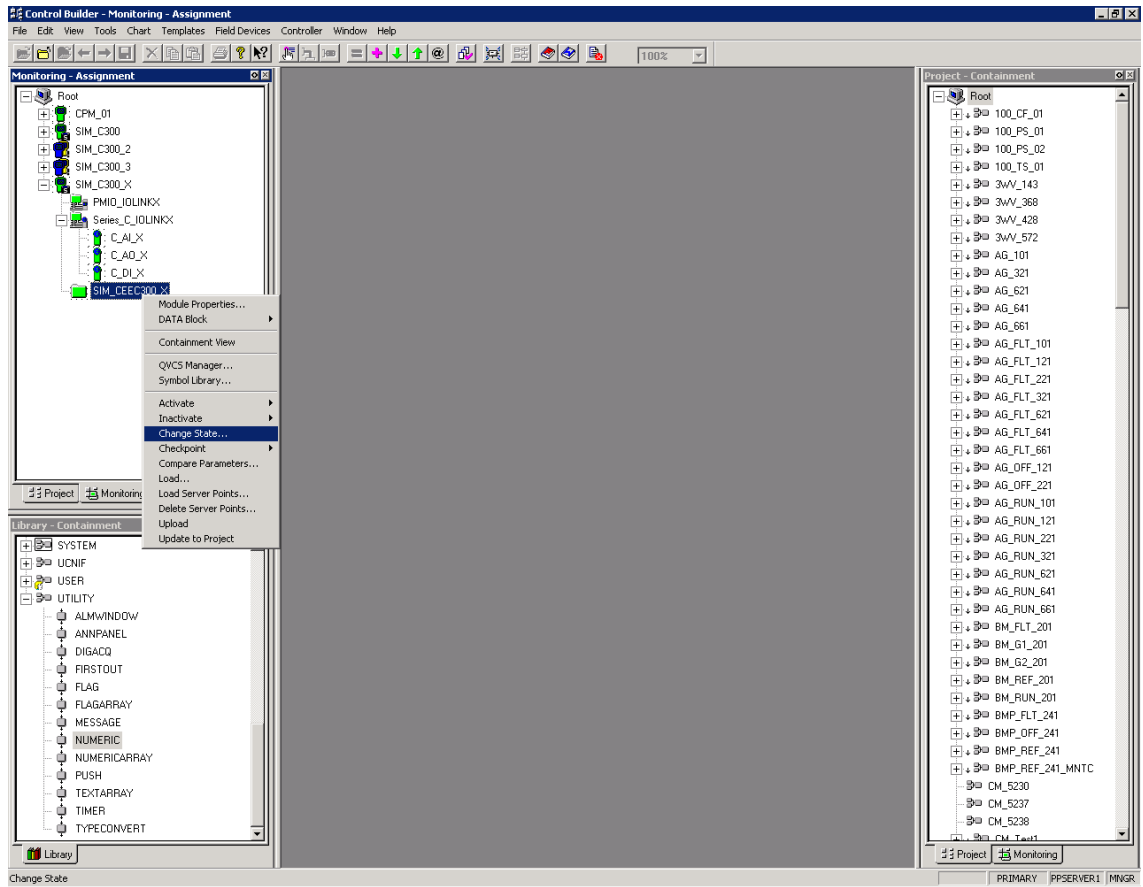
Step 7.

To download the written program, the CM environment must first be closed. When prompted to save, choose Yes.



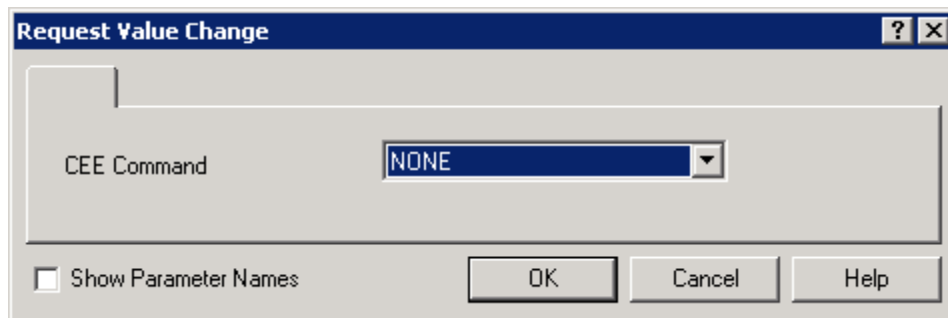
Step 8.

Switch over to the Monitoring tab and expand the SIM-C300. The current CEE of the SIM-C300 must be placed into a state of IDLE before downloading the new CEE.



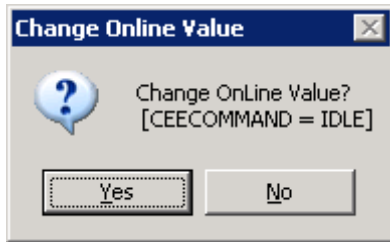
Step 9.

In the Request Value Change Window that appears, select the CEE Command to IDLE.



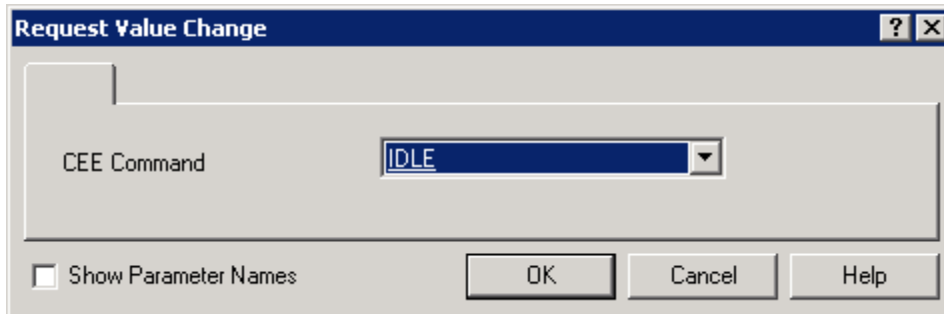
Step 10.

Choose Yes to confirm when prompted by the Change Online Value window.



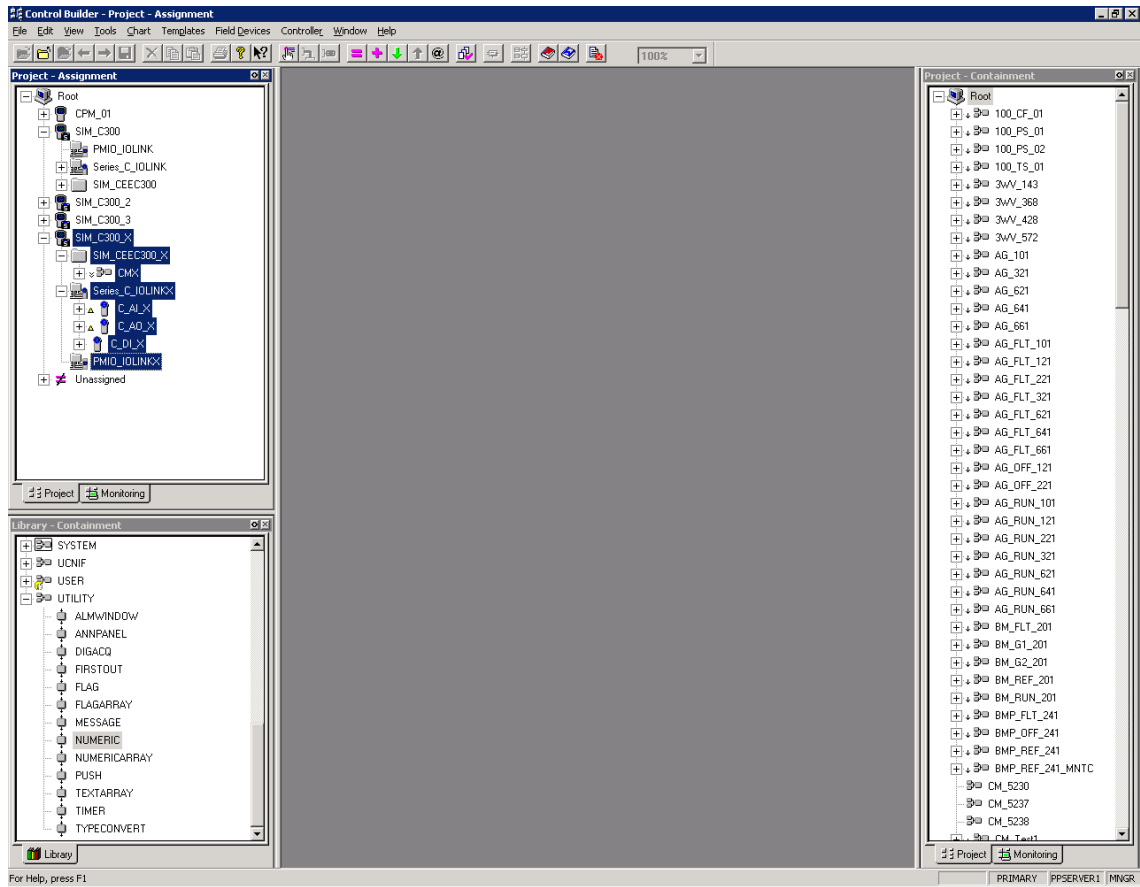
Step 11.

Back in the Request Value Change window, click OK.



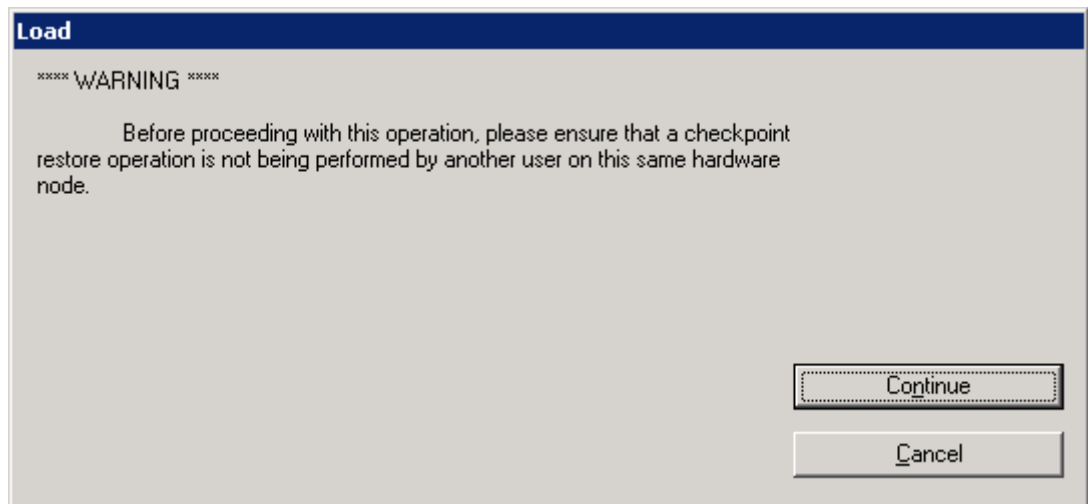
Step 12.

Switch back into the Project tab and expand the SIM-C300. Select all components in its tree list and click the green downwards pointing arrow on the tool bar to download.



Step 13.

Click Continue on the Load window when prompted.




Step 14.

In the Load Dialog box that follows, check both the boxes that state; “Automatically change ALL highlighted control elements to INACTIVE/OUT_OF_SERVICE before load” and “Automatically change ALL control elements to the state selected in “Post Load State” after the load is completed” then click OK.

The screenshot shows a 'Load Dialog' window with a table of control elements. The table has five columns: 'Load', 'Load List', 'Current State', 'State To Load', and 'Post Load State'. The 'State To Load' column for the last three rows is highlighted in yellow. Below the table, there is a warning icon and a message: 'Some control elements are currently in the ACTIVE/IN-SERVICE state!'. Two checkboxes are checked: 'Automatically change ALL highlighted control elements to INACTIVE/OUT_OF_SERVICE before load' and 'Automatically change ALL control elements to the state selected in "Post Load State" after load is completed'. At the bottom right, there are 'OK', 'Cancel', and 'Help' buttons.

| Load | Load List | Current State | State To Load | Post Load State |
|-------------------------------------|------------------|-------------------|---------------|-----------------|
| <input checked="" type="checkbox"/> | SIM_C300_X | CEERUN | N/A | N/A |
| | SIM_CEEC300_X | RUN | N/A | N/A |
| | Series_C_IOLINKX | ONLINE(SeriesCIO) | N/A | N/A |
| | PMIO_IOLINKX | ONLINE(PMIO) | N/A | N/A |
| <input checked="" type="checkbox"/> | CMX | Not Loaded | INACTIVE | ACTIVE |
| | CMX.AICHANNEL_01 | Not Loaded | Inactive | Active |
| | CMX.AOCHANNEL_01 | Not Loaded | Inactive | Active |
| <input checked="" type="checkbox"/> | C_AI_X | Run | IDLE | Run |
| <input checked="" type="checkbox"/> | C_AO_X | Run | IDLE | Run |
| <input checked="" type="checkbox"/> | C_DI_X | Run | IDLE | Run |

 Some control elements are currently in the ACTIVE/IN-SERVICE state !

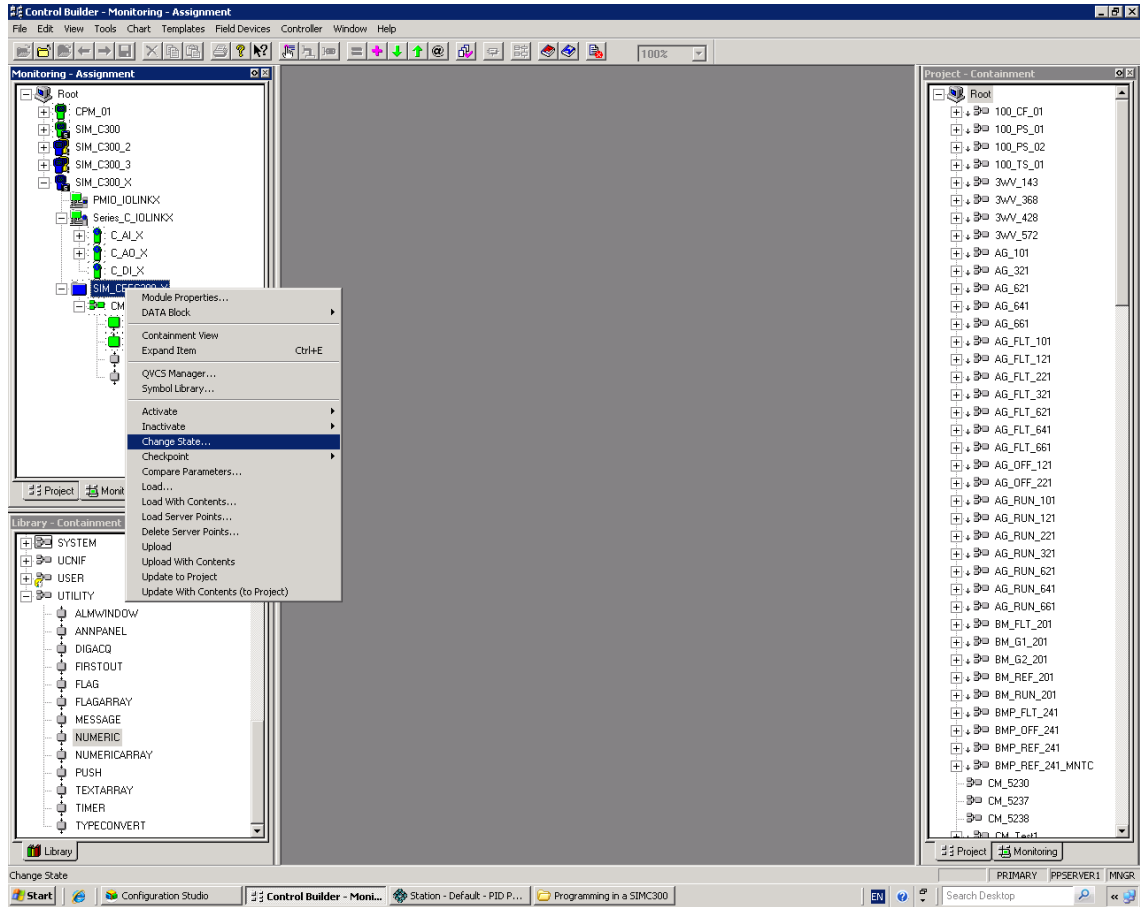
Automatically change ALL highlighted control elements to INACTIVE/OUT_OF_SERVICE before load

Automatically change ALL control elements to the state selected in "Post Load State" after load is completed

OK Cancel Help

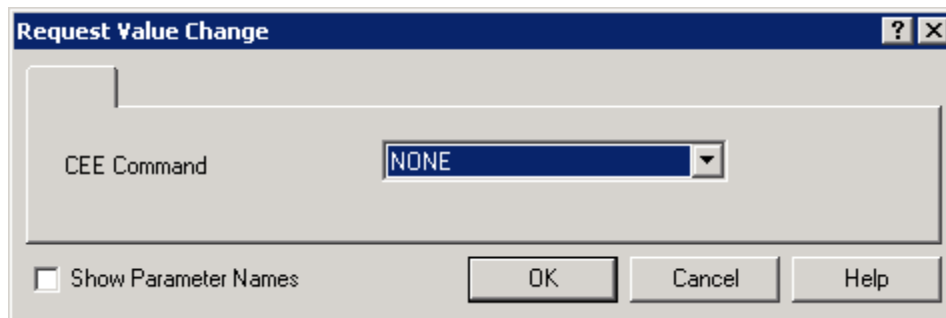
Step 15.

Proceed to switch back to the Monitoring tab and once again, the state of the CEE must be changed. Right click the CEE and select the option to Change State.



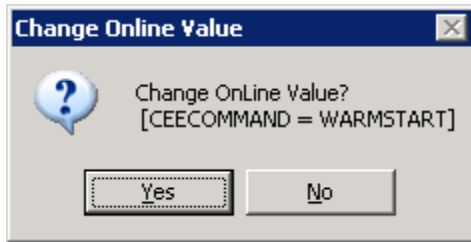
Step 16.

This time, the CEE Command must be changed to WARMSTART.



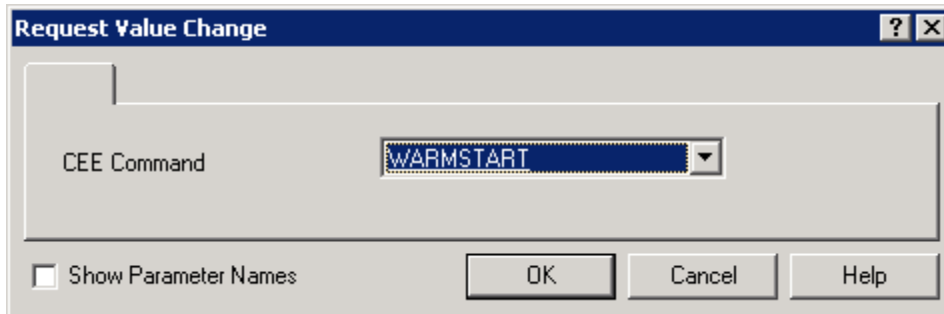
Step 17.

When prompted to confirm, click OK.



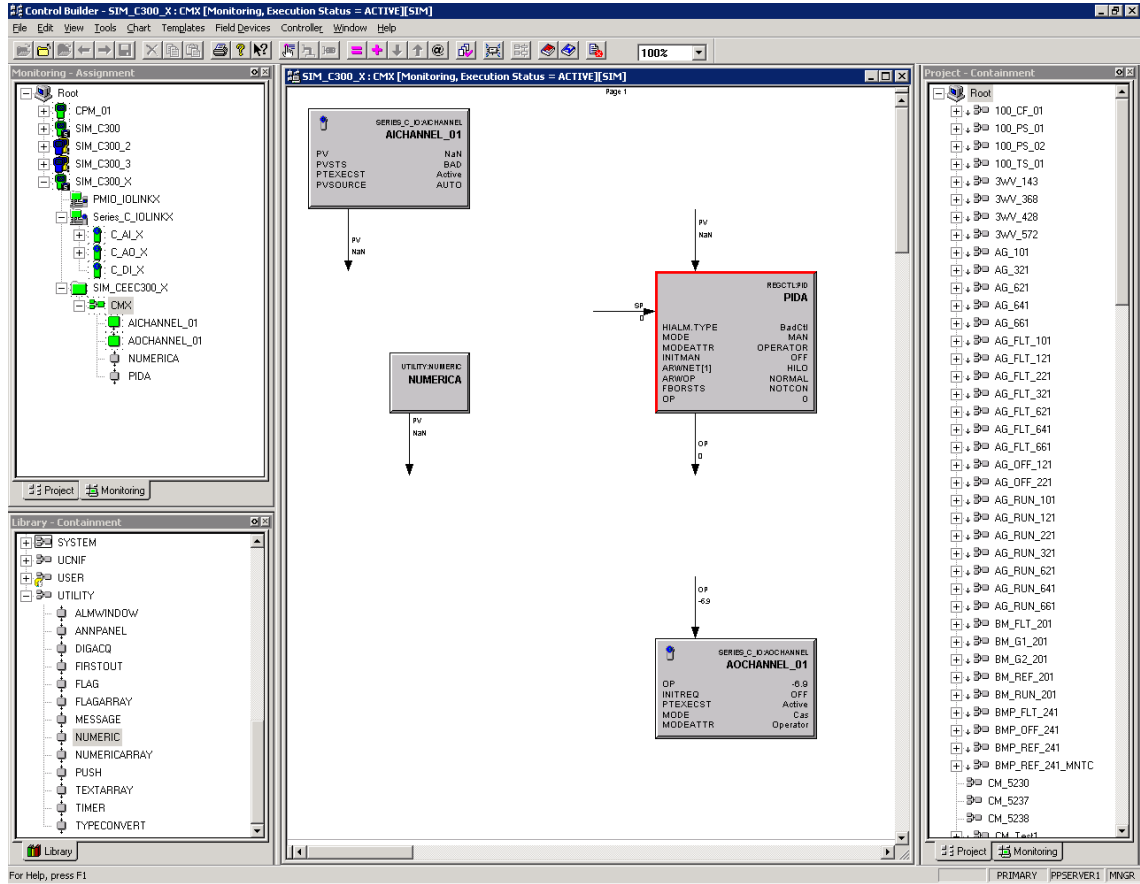
Step 18.

Back in the Request Value Change window, click OK.



Step 19.

Admire the fact that the program has been successfully loaded into the memory of the SIM-C300.



Appendix D

Guide to installing the Microsoft Excel Data Exchange Add-In.



MURDOCH
UNIVERSITY

PERTH, WESTERN AUSTRALIA

Appendix D: Guide to Installing the Microsoft Excel Data Exchange Add-In

This document provides step-by-step instructions to assist in the installation of the Microsoft Excel Data Exchange Add-In onto client computers.

Edwin Lum

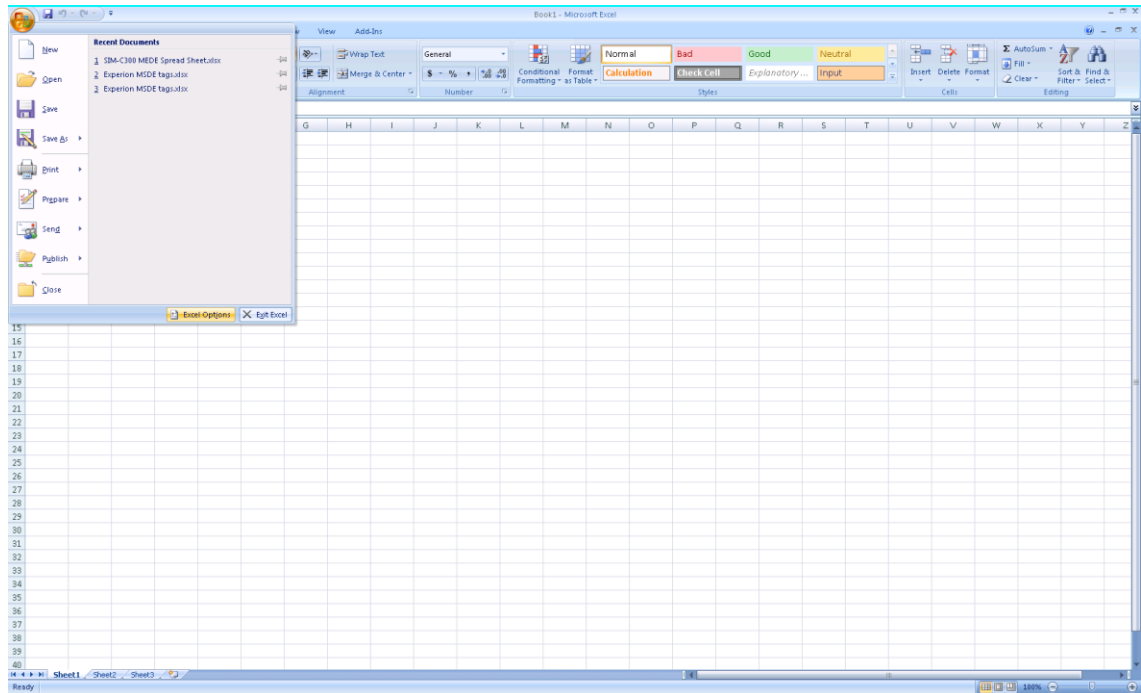
Honeywell Experion for Teaching Purposes Thesis 2011

Instructions

Please read and understand the set of instructions before undertaking any work to avoid any confusion or mishaps.

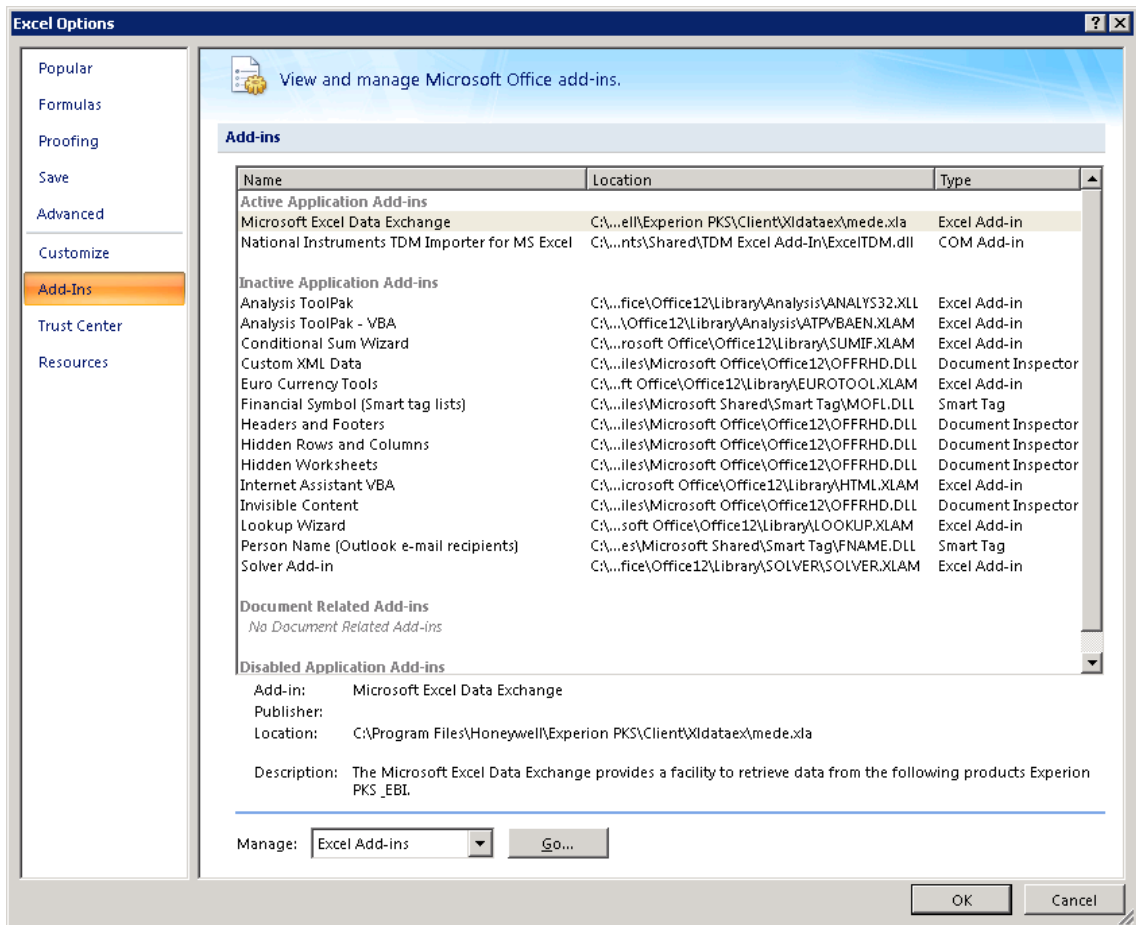
Step 1.

Open Microsoft Excel. Click Start and select Excel Options.



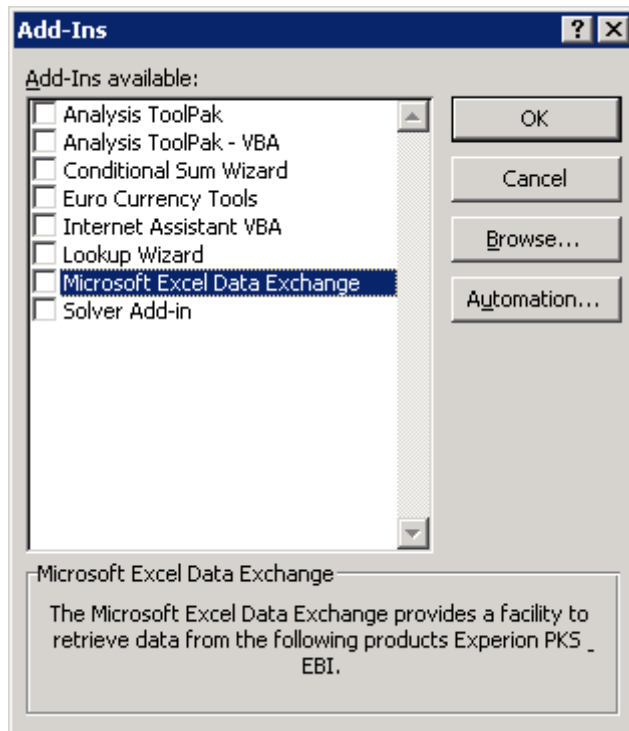
Step 2.

In the Excel Options window, click on Add-Ins. In the Manage drop down menu, select Excel Add-Ins and click Go...



Step 3.

In the prompted Add-Ins dialog box, should Microsoft Excel Data Exchange be listed, check it and click OK. If not, click Browse...

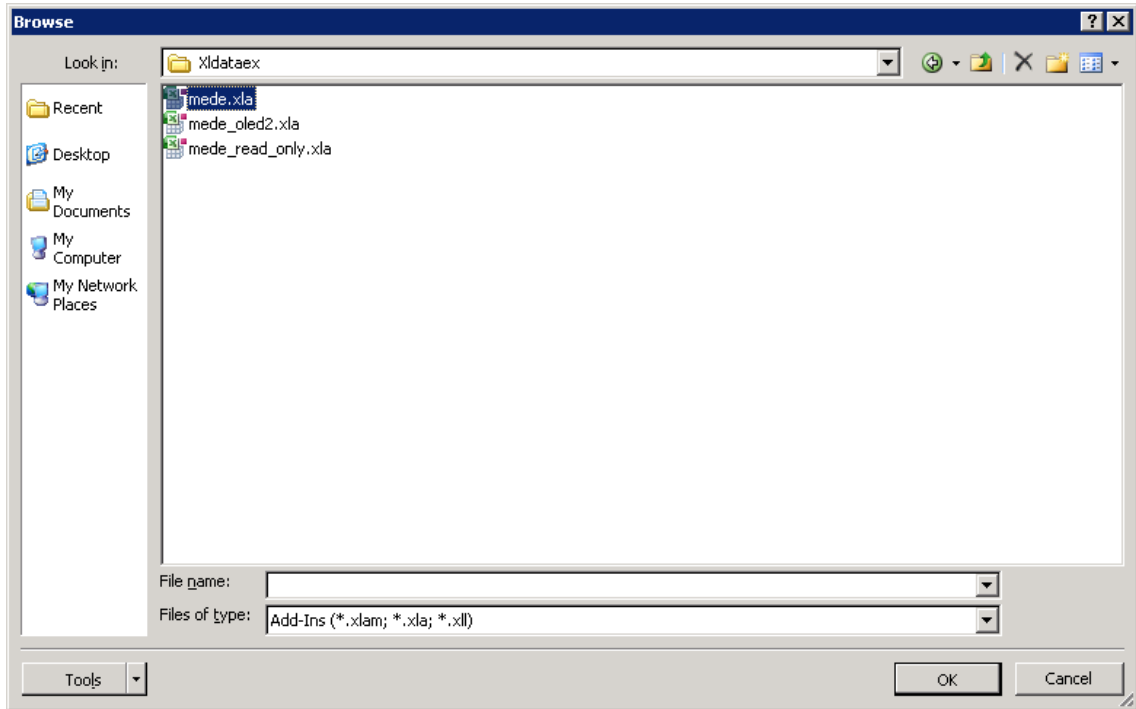


Step 4.

In the Browse window, direct the address to look in in following link.

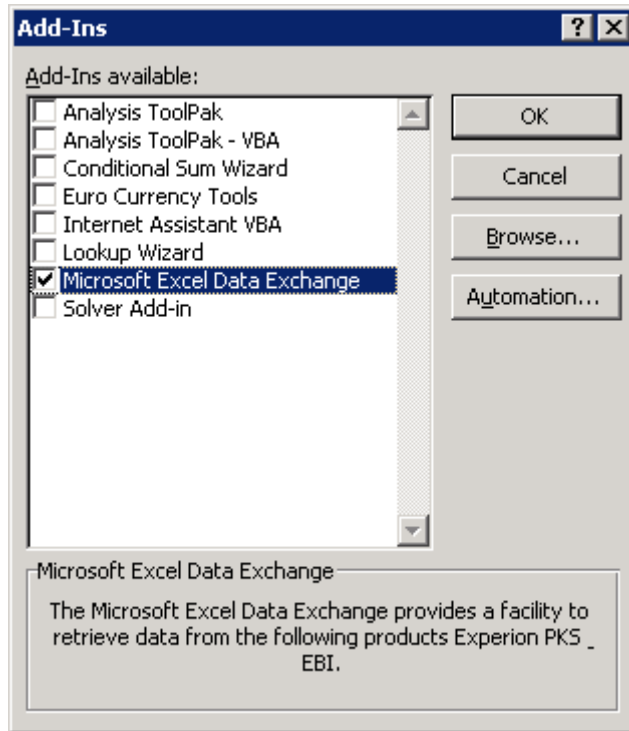
C:\Program Files\Honeywell\Experion PKS\Client\Xldataex

Select mede.xla and click OK.



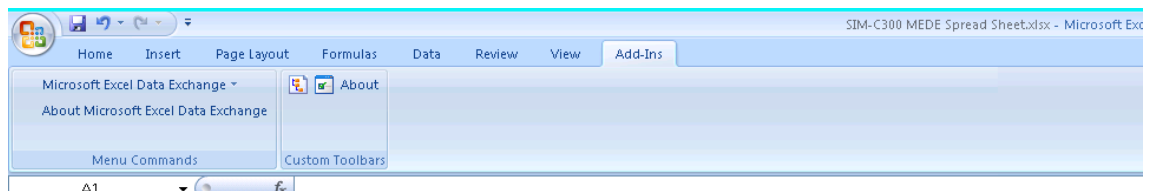
Step 5.

Back in the Add-Ins dialog box, check Microsoft Excel Data Exchange and click OK.



Step 6.

A new ribbon should appear on the Excel toolbar named Add-Ins.



Data values may now be retrieved using the MEDE Wizard or through cell formulas.

Appendix E

A solution guide to fix an Error 7045.



MURDOCH UNIVERSITY

PERTH, WESTERN AUSTRALIA

Appendix E: Solution Guide to Error 7045

This document provides step-by-step instructions to overcome an error associated with using simulated Honeywell C300 controllers.

Edwin Lum

Honeywell Experion for Teaching Purposes Thesis 2011

About this Guide

At the start of each week, all loaded simulated Honeywell C300 controllers appear to go offline. This is assumed to be because of the maintenance operations that execute every Sunday at Murdoch University's Pilot Plant. As a result, each simulated Honeywell C300 controller, commonly known as a SIM-C300, that is required must be reloaded to once again utilise it. In this guide, step-by-step instructions are provided to get the desired SIM-C300 back online.

Instructions

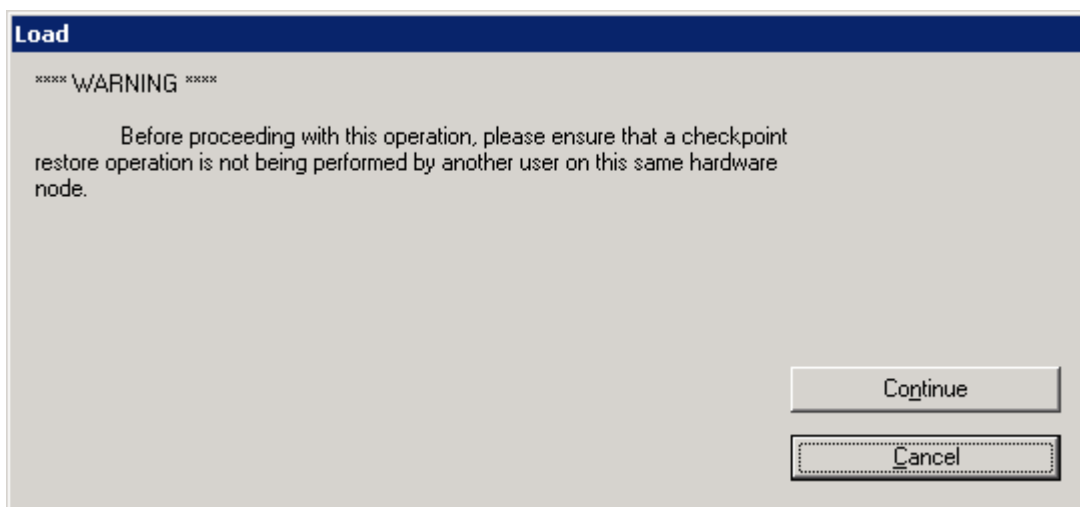
Please read and understand the set of instructions before undertaking any work to avoid any confusion or mishaps.

Step 1.

In Project tab of Control Builder, select the SIM-C300 and click the green downwards pointing arrow on the tool bar to download.

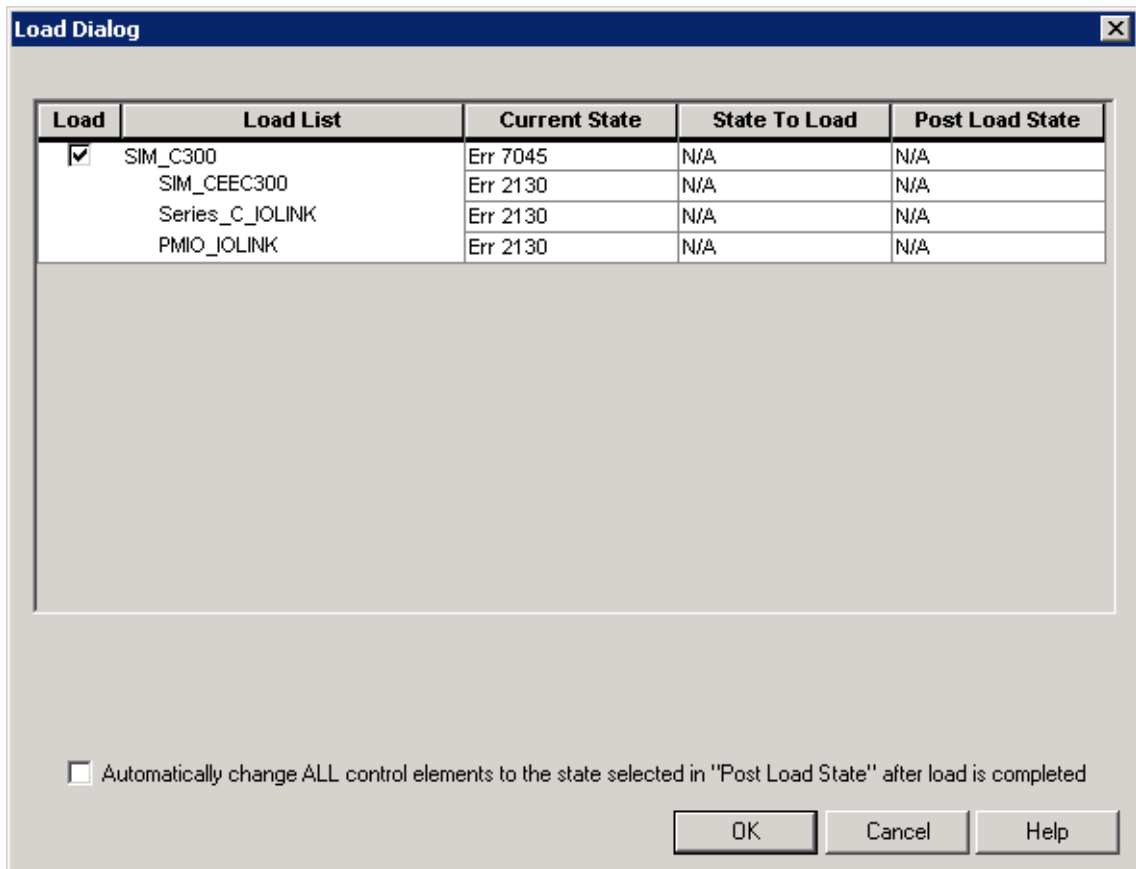
Step 2.

This will then prompt a Load window. Click Continue.



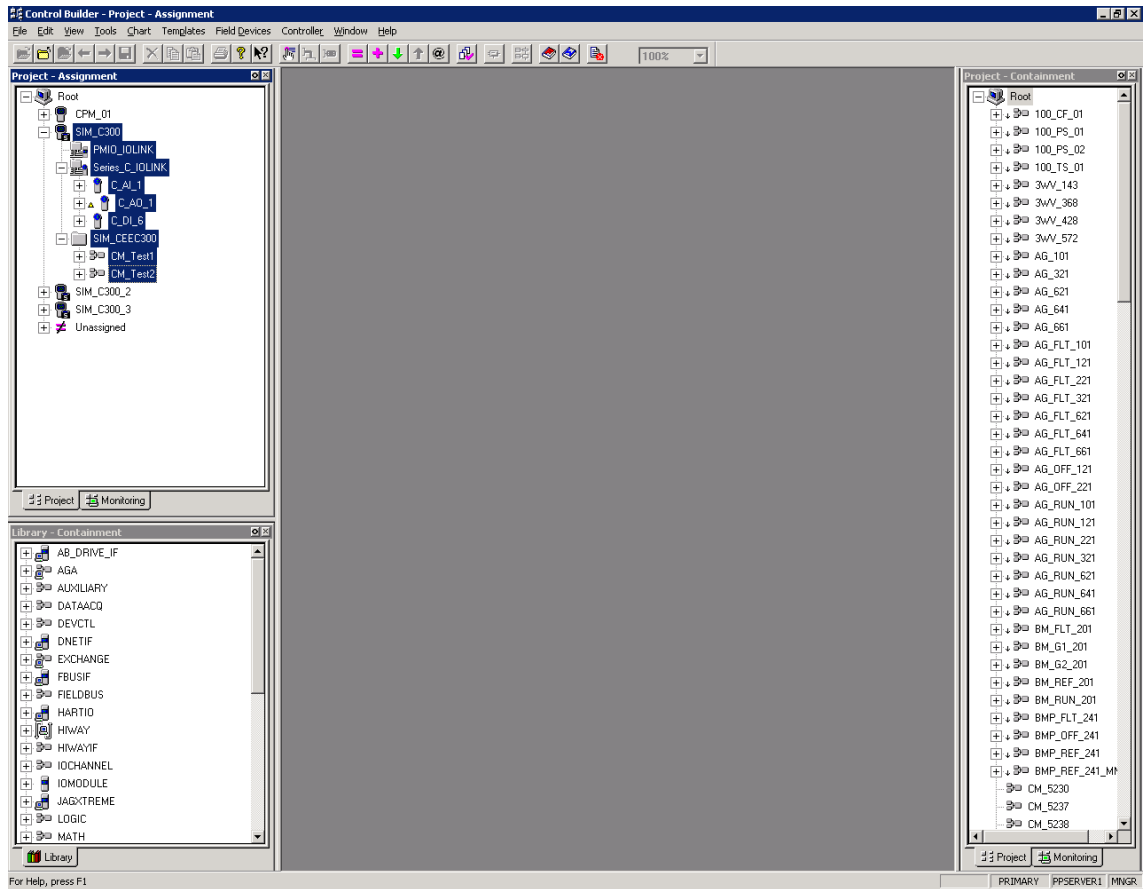
Step 3.

A Load Dialog window will then follow. Check the “Automatically change ALL control elements to the state selected in “Post Load State” after the load is completed” box and click OK.



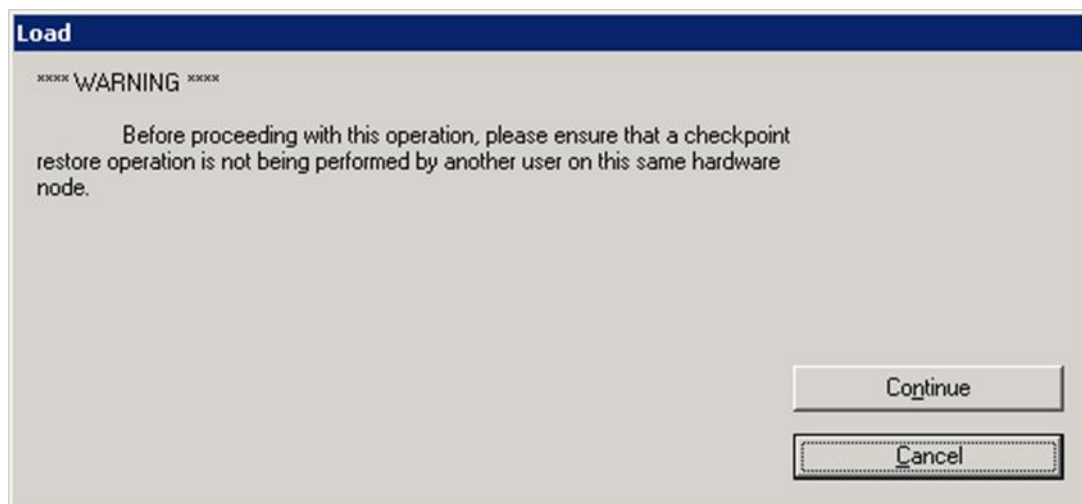
Step 4.

Expand the SIM-C300 in the Project tab and select all items listed beneath its tree. Download these items by clicking the green downwards pointing arrow on the tool bar.



Step 5.

Click Continue when prompted again by a Load window.

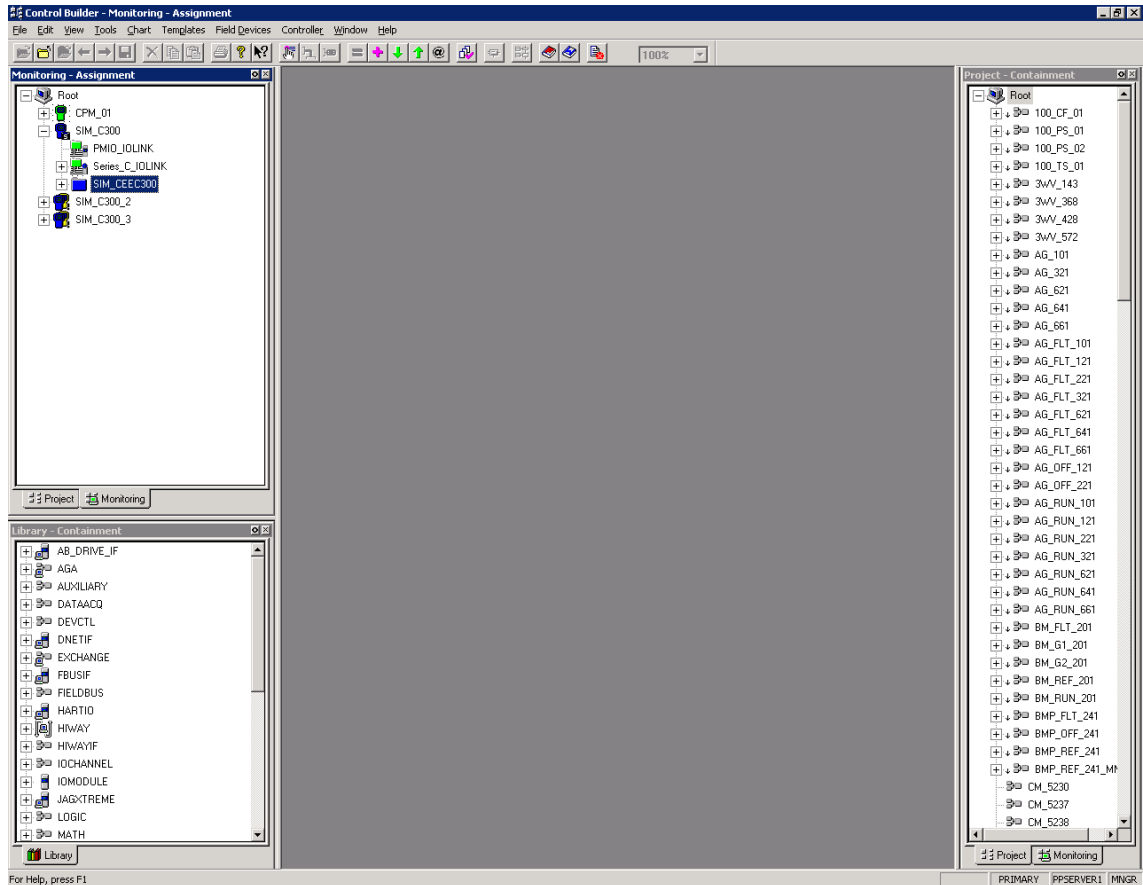


Step 6.

In the Load Dialog window that ensues, check both boxes at the bottom and click OK.

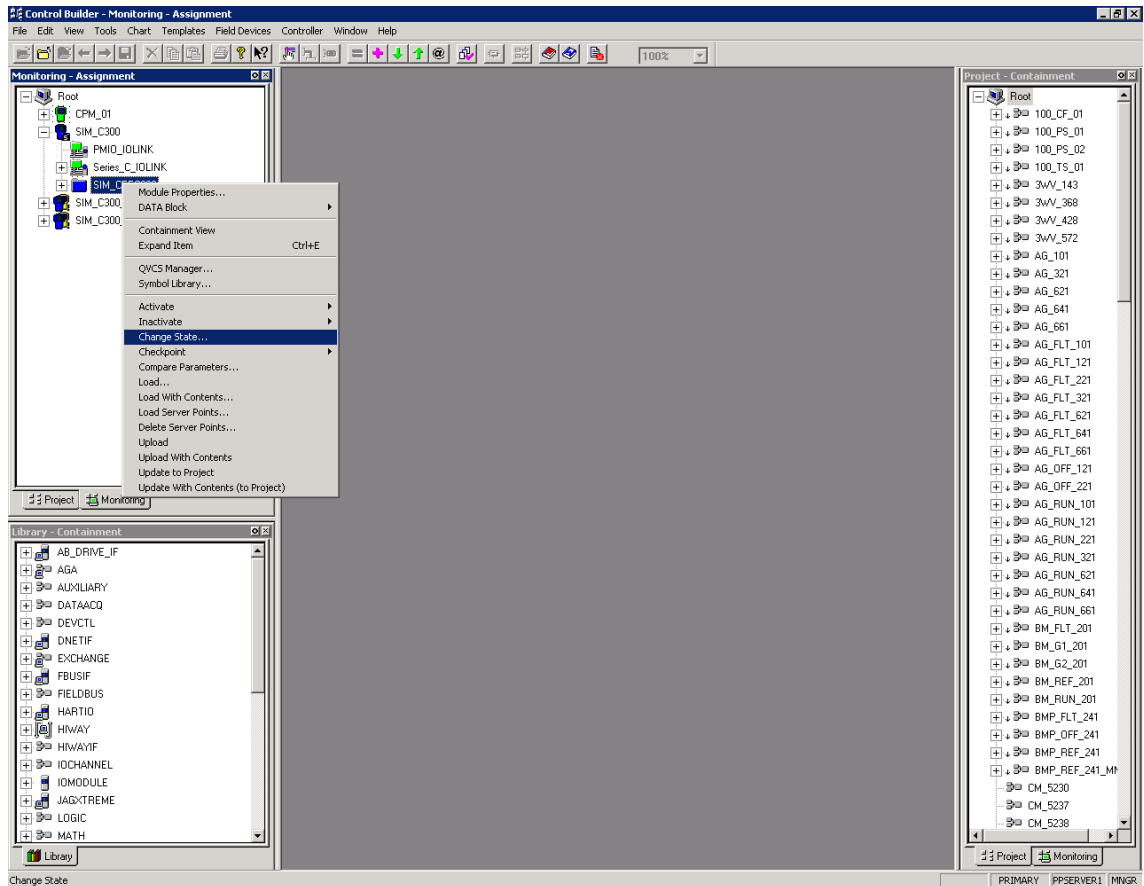
Step 7.

Switch over to the Monitoring tab to perform the following tasks. Expand the SIM-C300 and select its Control Execution Environment (CEE).



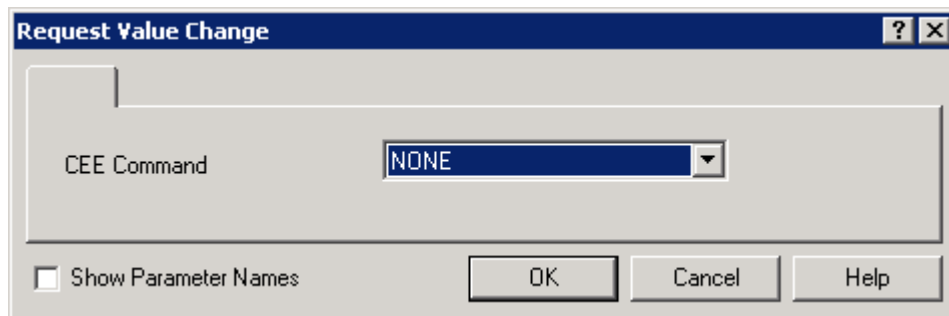
Step 8.

Right click on the CEE and select the option to Change State.



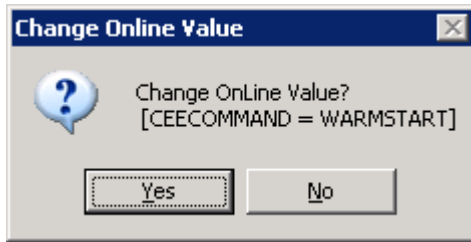
Step 9.

Alter the CEE Command to WARMSTART when the Request Value Change window appears.



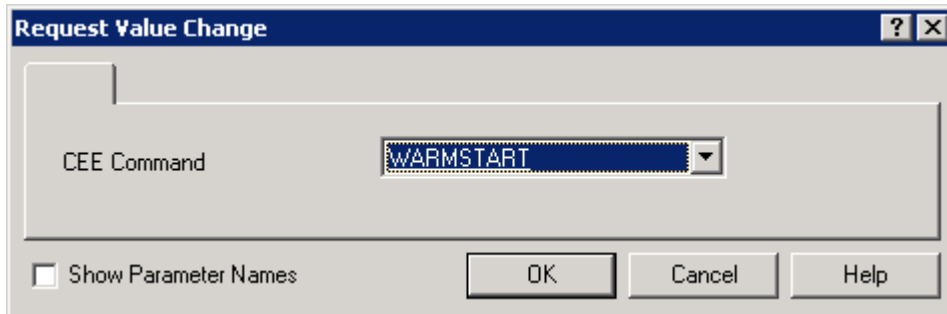
Step 10.

A Change Online Value window will then ask for confirmation. Click Yes.



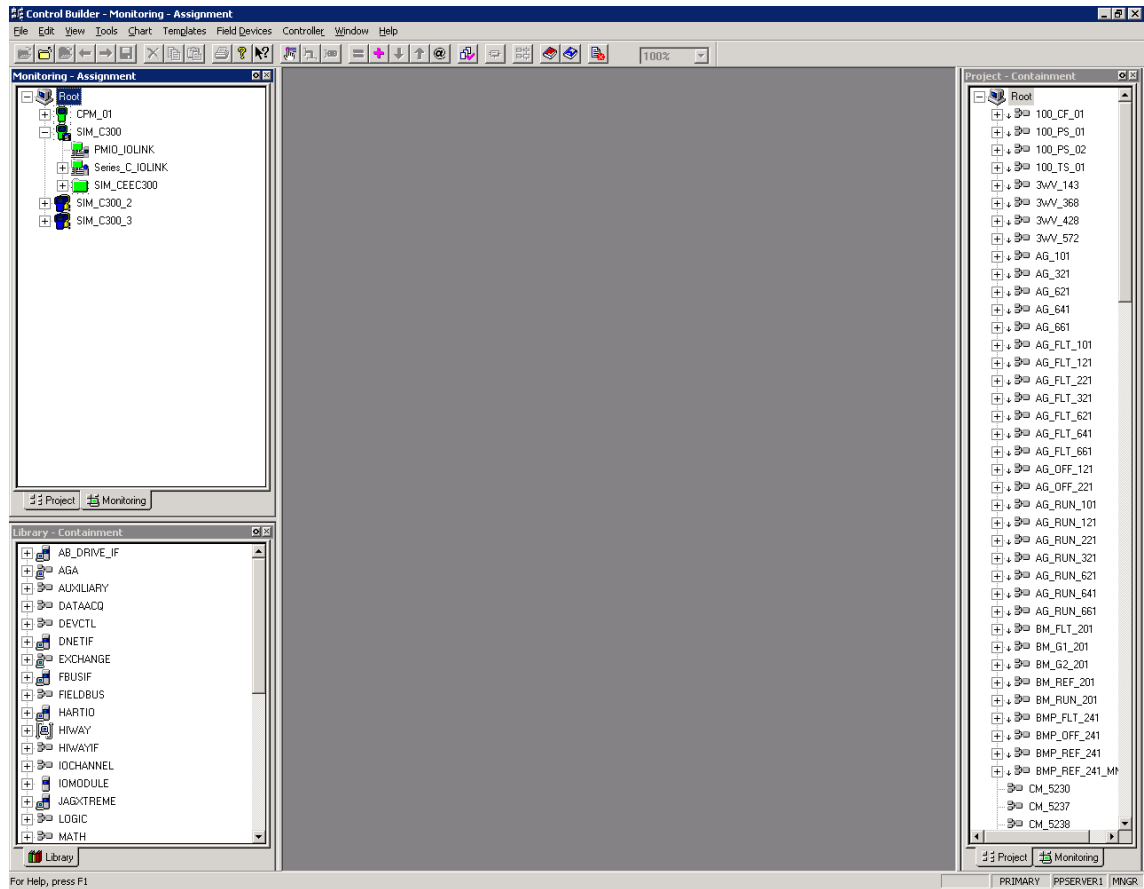
Step 11.

Back on the Request Value Change window, click OK.



Step 12.

Check that the SIM-C300 has been successfully reloaded in the Monitoring tab indicated by green icons.



This page is intentionally left blank.