VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Implementation of Communication Protocol BACnet in AC500

Implementace komunikačního protokolu BACnet v AC500

Ekaterina Adeeva

Diploma Thesis

Supervisor: prof. Ing. Jiří Koziorek, Ph.D.

Ostrava, 2021

Abstrakt

Tato práce se zabývá návrhem a implementací komunikační sítě BMS systému pro datová centra. Pro realizaci sítě byly využity komunikační protokoy BACnet a OPC Classic. Cílem práce bylo vytvořit realizačně efektivní a modulární řešení, které lze výhledově aplikovat pro implementaci komplexejších řešení BMS systémů. Teoretická část obsahuje technický přehled oblasti aplikace a analýzu využití protokolu BACnet v průmyslových řídicích systémech. Výsledek praktické části přináší komunikační systém založený na protokolu BACnet, který slouží k výměně dat mezi dvěma PLC AC500 vyráběnými společností ABB. Součástí komunikačního systému je také distribuovaný řídicí system 800xA výrobce ABB, který zde zastává funkci vzdáleného monitorování a řízení, archivaci naměřených dat a alarmovou diagnostiku monitorovaného systému.

Klíčová slova

Interoperabilita; BACnet; BMS; Automatizace budov; Hierarchie automatizace; ISO/OSI model; Průmyslové řídicí systémy; Komunikační systém; PLC; OPC DA.

Abstract

This work deals with the design and implementation of a BMS system communication network for data centers. BACnet and OPC Classic communication protocols were used for the implementation of the network. The aim of the work was to create an implementation-efficient and modular solution that can be applied in the future for the implementation of more complex solutions of BMS systems. The theoretical part contains a technical overview of the application area and an analysis of the use of BACnet protocol in industrial control systems. The result of the practical part is a communication system based on BACnet protocol, which is used to exchange data between two AC500 PLCs produced by ABB company. The communication system also includes a distributed control system 800xA manufactured by ABB, which has the function of remote monitoring and control, archiving of measured data, and alarm diagnostics of the monitored systems.

Keywords

Interoperability; BACnet; BMS; Building Automation; Automation hierarchy; ISO/OSI model; Industrial control systems; Communication system; PLC; OPC DA.

Acknowledgement

I would like to thank prof. Ing. Jiří Koziorek, Ph.D. for leading this diploma thesis, including useful advice and consultations. I would also like to thank ABB company, specifically Michal Hawranek for help and support with practical part, gained knowledge and valuable experience, and Michal Robenek for help with formal issues and providing necessary equipment.

Table of Contents

List of Symbols and Abbreviations

ATA	Advanced Technology Attachment
AHU	Air-handling unit
ANSI	American National Standards Institute
ASHRAE	American Society of Heating, Refrigerating, and Air-Conditioning Engineers
AIS	Analog input signal
AC	Area controller
ABB	Asea Brown Boveri
APDU	Application protocol data unit
ARCNET	Attached Resource Computer Network
B-ASC	BACnet Application Specific Controller
BVLL	BACnet Virtual Link Layer
BACnet	Building Automation and Control Networks
BACS	Building automation and control system
BAS	Building Automation Systems
BIBB	BACnet Interoperability Building Block
BBMD	BACnet Broadcast Management Devices
BMCS	Building Management and Control Systems
BMS	Building Management Systems
CPU	Central processing unit
COV	Change of value
THS-CA	Cold aisle temperature and humidity sensor
CAN	Control Area Network
DH	Data hall
degC	Degree Celsius
DALI	Digital Addressable Lighting Interface
DIS	Digital input signal
DSS	Digital Signature Standard
DC	Direct current
DDC	Direct digital control
DCS	Distributed control system
EIA	Electronic Industries Alliance
ED	End-device
EWP	Engineering Workplace
ED	Entrance door
EIBA	European Installation Bus Association
FB	Function block
FBD	Function Block Diagram
FF	Foundation Fieldbus
FIB	Factory Instrumentation Protocol
Gbit	Gigabit
HW	Hardware
HVAC	Heating, ventilation, and air-conditioning
ID	Identifier
I/O	Input/Output
IEEE	Institute of Electrical and Electronics Engineers
IEC	International Electrotechnical Commission

ISO	International Organization for Standardization
IP	Internet Protocol
KNX	Konnex
LAN	Local Area Network
MCC	Master coordinating controller
MS/TP	Master-Slave/Token-Passing
MAC	Medium Access Control
Mbit	Megabit
MB	Megabyte
µs	Microsecond
ms	Millisecond
OPC DA	OPC Data Access
OPC	Open Platform Communications
OSI	Open Systems Interconnection
OWP	Operator Workplace
PA	Process Automation
Pa	Pascal
PAN	Personal area network
PC	Personal computer
PPP	Point-to-Point Protocol
POU	Program Organization Unit
PL	Powerline
PLC	Programmable logic controller
%RH	Relative humidity
RNRP	Redundancy network routing protocol
RF	Radio Frequency
s	Second
ST	Structured Text
SW	Software
t	Time
TCP	Transmission Control Protocol
TP	Twisted Pair
UML	Unified Modelling Language
UDP	User Datagram Protocol
VAV	Variable air volume
VLAN	Virtual Local Area Network
V	Voltage
WAN	Wide Area Network

List of Figures

List of Tables

Introduction

Building Management Systems (BMS), also known as Building Automation Systems (BAS), and Building Management and Control Systems (BMCS), are currently installed in many large buildings in order to monitor and control building technical systems and services, such as air conditioning, ventilation, lighting, and access control. The current generation of BMS is based on open communication protocols that allow integration of systems from multiple vendors and remote access from any place.

BACnet protocol is an industry-standard protocol that is widely used in the heating and ventilation industry. BACnet is a communication protocol for Building Automation and Control Networks. It is both ANSI and ISO standard protocol, registered by ASHRAE. The BACnet protocol provides mechanisms for computerized automation devices to exchange information, regardless of the particular building service they perform. The BACnet protocol defines a number of services for communication between different building systems and devices in building automation and control applications.

One of the segments in ABB operations is Data Center Automation. In today's data centers, an integrated building management solution is required in order to provide real-time information to multiple devices, ensure safety and security. ABB controllers provide control of the mechanical infrastructure of a data center. The BACnet protocol allows equipment in the building automation to communicate with the PLC. The biggest advantage of this automation solution is the possibility to integrate various communication systems with BMS.

The implementation of BACnet communication in one of ABB controllers AC500 for the use in Data Center Automation has become the assignment for this thesis. The main purpose of this thesis is to bring a simple and effective engineering solution for BMS BACnet communication between AC500 PLCs and consider the connectivity to the higher monitoring systems. The final solution is to be compared with other widely used solutions and protocols. The result of this thesis brings a compact and feasible project, which serves as practical support for the implementation of a more comprehensive solution.

The work starts with an introduction to building automation, an architecture of the distributed control system, and communication possibilities. The necessary theoretical basis for the execution of this work is an explanation of the basic principles of the BACnet communication protocol. The BACnet communication architecture in accordance with the OSI model is provided and each layer is described. Then, BACnet objects and services, which are used to transport the data between BACnet devices, are mentioned. The following chapter analyzes the use of BACnet in industrial control systems.

At the beginning of the practical part, the design of the entire communication structure is determined. The selection of HW elements and related SW resources is explained here as well as the purpose of the designed communication network. In the following chapter, a design of the logical and physical architecture is analyzed. It also contains settings for the system configuration.

The implementation of the practical part includes the control application for data transfer between two PLCs using the BACnet protocol and implementation of Operator Workplace that allows monitoring of current states and values from the devices participating in the control system.

The final chapter deals with the testing of the developed application. A virtual machine simulates the behavior of the real installation. It replaces the real 800xA System that should be installed on Windows

Server OS. The system sets high requirements for RAM memory and CPU. Besides that, a virtual machine does not affect the real PC in case of failure. Finally, the work is evaluated in terms of engineering complexity and efficiency in comparison with other widely used solutions to bring further development opportunities.

1 Building Automation Concepts

1.1 Overview of the Area of Application

A building automation system (BAS), otherwise known as a building management system (BMS), consists of a computer-based control system installed in buildings that controls and monitors the conditions of an indoor environment. Building automation systems are commonly implemented in large functional buildings for the control of heating, ventilation, and air-conditioning (HVAC) systems, as well as a wide range of mechanical and electrical systems.

Building automation is devoted to the measurement and control of plants, systems, and installations of building services. The historical roots lie in the traditional core domain of HVAC, where BAS was designed to increase both comfort and energy savings in the building. They were based on the concept of direct digital control (DDC). Since then, BASs have constantly developed by covering more building services such as systems used for constant light control, scene management, and shading.

Motivations for the use of BAS are diverse. The first and most important are considerable savings in operational cost since highly energy-intensive areas such as HVAC can be optimized through advanced control strategies. Also, limiting peak energy consumption by coordinating devices provides high savings potential.

With the adoption of BAS, improved building management also becomes possible, where the focus is on central access to all building systems. This allows problems to be detected and fixed more easily and quickly. Both preventive and corrective measures can be planned more efficiently, and thus costs are further decreased. This general approach of transparently accessing all systems is supported by individual user controls. They allow changes to be applied during regular operation and additionally do not require specially trained staff, despite the fact that the equipment to be controlled is technically complex. Additionally, remote access to building systems becomes possible, which is particularly useful when building sites are geographically dispersed. Travel time and costs are avoided and faults might be resolved remotely. Another benefit is the possibility introduced by BAS to collect high quality-data on energy consumption. Last, but not least, increasing comfort for people in the building is a key benefit. Although it cannot be easily estimated, the positive effect is evident. People who feel comfortable in a building will show higher motivation and therefore also be more productive. The use of modern technology also appeals to tenants concerned with their image and allows building owners and tenants to set themselves apart from the competition by offering higher value. Finally, building security, access control, and people safety are other important topics that can be solved with the help of BAS. [1]

1.2 Industrial Automation Hierarchy

A BAS is a distributed system focusing on the computerized control and management of building services, also referred to as a building automation and control system (BACS). The architecture of this distributed system can be structured into three layers. This architecture is depicted in figure 1. On the lowest layer, the field layer, control over actuators, such as fans or valves for hot water and coolant is provided, and an adaption of them in response to data point values for sensors, actuators, and set points is performed. The automation level communicates vertically with the field level by exchanging values with the field devices as well as horizontally to interact with controllers on the same level. Typical functions in this level are controlling functions, alarm generation, and scheduling. The management layer is the top level and the most abstract layer in the BAS, where global parameters are maintained for the automation layer. Typical functions are management, data retrieval and storage with trending, global schedules, global alarm collection and acknowledgment, visualization, and interfaces to other systems.

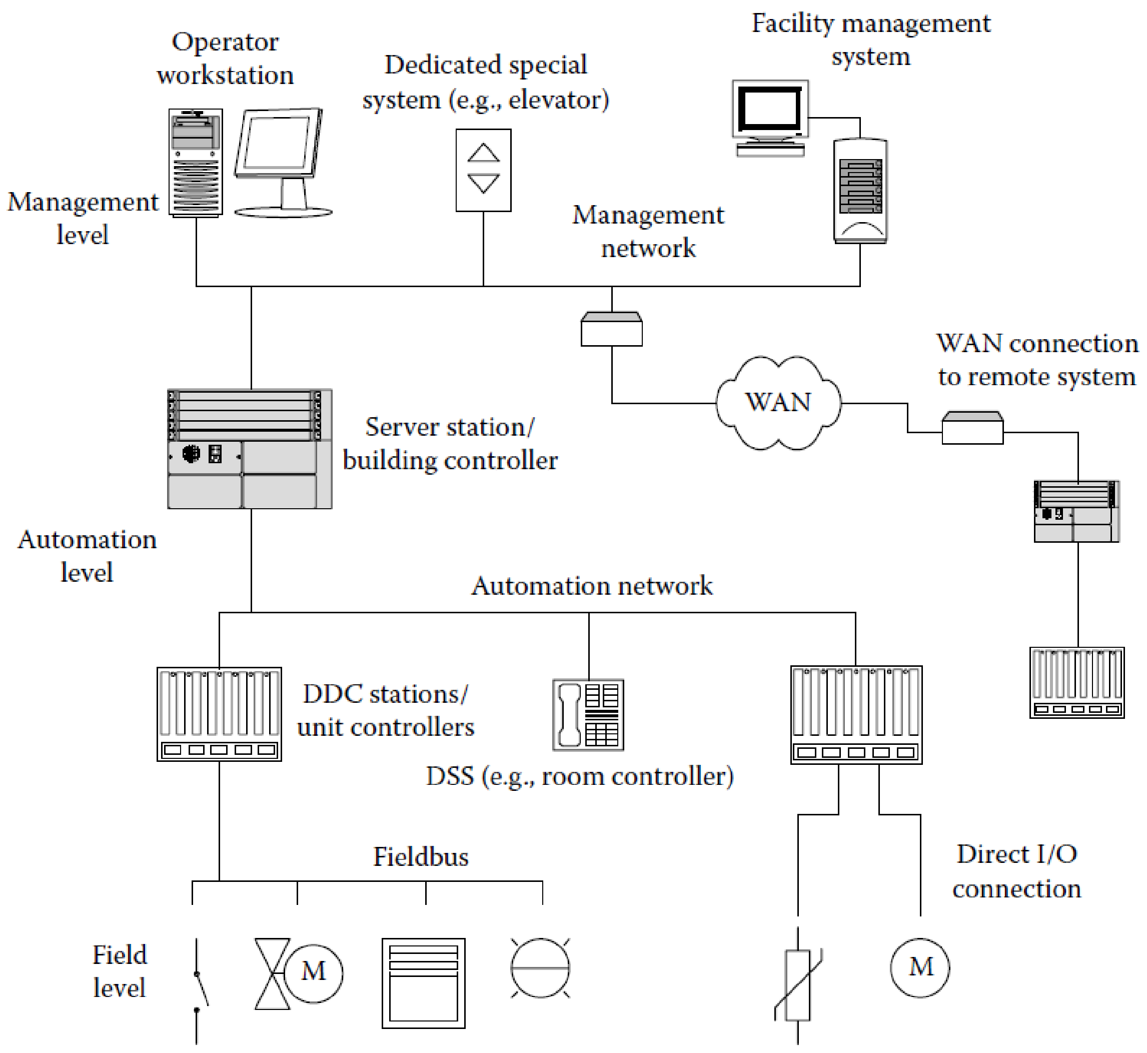

Operator workstation
Dedicated special system (e.g., elevator)
Facility management system
Management level
Management network
WAN connection to remote system
WAN
Server station/ building controller
Automation level
Automation network
DDC stations/ unit controllers
DSS (e.g., room controller)
Fieldbus
Direct I/O connection
Field level
M
M

Figure 1 – Field, automation, and management levels with respected devices [1]

In today's BAS, the different layers are composed of certain types of devices. Industrial communication represents the means of exchanging information between devices in an automated system. Communication inside an automated system falls into horizontal communication, used for the exchange of information within each level, and vertical communication that allows different levels to execute automated functions. Horizontal and vertical communication is carried out by fieldbuses and networks. [2]

1.2.1 Fieldbus Communication

The fieldbus is a serial digital communication that permits the connection of multiple field instruments and processes, located at the field level, and operator stations. Field devices carry out control functions, such as switching, setting, reporting, measuring, and metering, and enable monitoring by means of supervision software. They have built-in microcontrollers, which allow them to send and receive information over a fieldbus, communicating either directly with each other or with control and regulation devices at the automation level above.

The first generation of fieldbus technology was developed within the 1980s. It replaced the parallel transmission technology with a single bus cable. Since 1999, fieldbus systems have been standardized globally with IEC 61158. [3] Today, many different fieldbus systems with different qualities, depending on the scope, are available on the market. Table 1 shows some examples of fieldbuses and their fields of application. [2]

Table 1 – Examples of fieldbuses and their scope

Fieldbus	Scope
Manufacturing Automation Protocol (MAP)	Automation
Manufacturing Message Specification (MMS)	
Modbus	Manufacturing automation
PROFIBUS	
INTERBUS	
Control Area Network (CAN)	
WorldFIP	Process automation
Foundation Fieldbus (FF)	
PROFIBUS-PA	
LonWorks	Building automation
BACnet	

1.2.2 Communication Over Networks

Communication at the automation level is implemented mainly through local area networks (LANs). This is usually a small network, available within a restricted area, such as a building. LAN uses high-speed data transmission over wires or radio waves to allow technical systems to communicate with each other.

Communication is carried out in accordance with protocols, typically based on the ISO/OSI reference model.

The Building Automation Control Network (BACnet) communication protocol has been designed specifically to meet the communication needs of building automation. The BACnet protocol enables computerized equipment to exchange information, regardless of the purpose. [2]

2 Technical Overview of BACnet

2.1 Definition

"BACnet" is an acronym that stands for "Building Automation and Control Networks". It represents a data communication protocol, i.e. a set of rules, which defines how computers exchange information with each other. The rules of BACnet have been developed by the American Society of Heating, Refrigerating, and Air-Conditioning Engineers (ASHRAE) in 1987. In 2003, BACnet was adopted as a global standard by the International Organization for Standardization (ISO) and is currently known as ISO 16484-5. [4]

The basic idea of BACnet is compatibility and interaction between systems of different manufacturers. In the 1980s, if the project included systems by different vendors, programmers had to develop an interface and communication protocol for each particular case. With the increasing number of vendors, the project was becoming impossible to maintain. BACnet was intended to replace proprietary systems and communication protocols. [5]

2.2 Overview of Basic Principles

Three key elements of BACnet are:

1. The network media.
2. Objects to transport the building automation data.
3. Services to transport the data between BACnet devices. [5]

The BACnet protocol determines the method of transferring data frames from one system to another. The messages may contain various information, such as:

- binary input and output values,
- analog input and output values,
- schedule information,
- alarm and event information,
- security files,
- control logic.

BACnet messages can be transported over Ethernet, as well as other local area networks (LANs).

A building automation device must have a data structure for each input, output, and other components. These data structures, so-called objects, enable communication with the individual devices and their functions over BACnet. Each object has certain properties (e.g. name or present value), which can be requested or set. The basic principle of interoperability is that the internal configuration has no role in communication with a particular device. [2]

2.3 The BACnet Communication Architecture

The OSI (Open Systems Interconnection) model is a theoretical structure used to define the functions of a networking system. The model was created in 1984 by the International Organization of Standardization (ISO). According to the OSI model, communications between different systems are divided into 7 layers: Physical, Data Link, Network, Transport, Session, Presentation, and Application.

BACnet is based on this model but uses only layers 1, 2, 3, and 7. The functions of transport, session, and presentation layers (4, 5, 6) are integrated into the application layer. Table 2 shows a comparison of BACnet layers with the OSI model. [6]

Table 2 – BACnet layers compared to the OSI model [6]

BACnet Layers							Equivalent OSI Layers	
BACnet Application layer							Application	7
BACnet Network layer							Network	3
ISO 8802-2 (IEEE 802.2) Type 1		MS/TP	PPP	LonTalk	BVLL	BZLL	Data link	2
ISO 8802-3 (IEEE 802.3)	ARCNET	EIA 485	EIA 232		UDP/IP	ZigBee	Physical	1

Such a four-layer collapsed architecture was implemented for the purpose of reducing overhead costs and minimizing demands related to data transmission.

The physical layer allocates the resources for connecting BACnet devices and data transmission through electronic signals. The data link layer performs such basic BACnet functions, as data organization and access control, addressing, error handling, and flow control.

When only one network is used, most functions of the network layer duplicate data link layer functions or may not be used. The network layer is required when there is a need to differentiate between local and global addresses, and to transmit messages to the relevant networks, for example, when two or more subnetworks are connected by a router using different data link layer options.

The application layer provides functions of the transport layer, most of which are similar to the functions of the data link layer (e.g. data organization and flow control) but are different in scope. The data link layer deals with point-to-point connections between two devices in a single network, while the transport layer handles end-to-end connections across multiple networks.

Most communications in BACnet are very short and the data format does not change. Therefore, restart and interrupt mechanisms, performed by session and presentation layers, are not required.

In conclusion, four-layer collapsed architecture enables the implementation of BACnet to be simple and cheap, both in hardware and software components. [2]

2.4 Network Media

The BACnet standard defines seven network types, which serve as the transport for BACnet messages. The network types incorporate the physical and data link layers, forming the MAC (Medium Access Control) layer. BACnet messages are independent of the MAC layer used for data transfer, therefore, there are no distinctions between messages in different technologies. [6]

The technologies used to transport messages differ in performance, cost, and type of transmission media, for example, twisted-pair, coaxial and fiber-optic cable. BACnet supports several LAN technologies. The choice of a suitable LAN technology for BACnet is based on the following criteria:

- transfer rate,
- response time,
- number of devices (nodes),
- maximum cable length,
- cost.

UDP/IP

BACnet/IP is frequently used with existing Ethernet infrastructure, VLAN, and WAN networks. Devices are connected directly to Ethernet switches or hubs. This type of LAN is fast and high-performance, but the most expensive. BACnet/IP uses UDP/IP for compatibility with an existing IP infrastructure. If BACnet/IP is used with multiple IP subnets, then BACnet Broadcast Management Devices (BBMDs) are required to manage inter-subnet BACnet broadcast messages. [6]

MS/TP

MS/TP stands for Master-Slave/Token-Passing protocol, which is a simple and low-cost technology suitable to control and operate units that do not require a fast transfer rate. It is the most popular type of BACnet LAN for unitary and application-specific controllers. The physical layer is based on the EIA-485 standard (RS-485). The data link layer controls communication between stations by sending a token – "authorization". MS/TP protocol has master and slave stations. Only the master can receive a token and start data exchange. BACnet field devices represent slave stations connected to a master. [2]

Ethernet

The most widespread LAN technology is Ethernet. It was developed in the 1970s and standardized as IEEE 802.3. It was initially used in office intranet systems but now it is widely used in industrial communication systems and wide area networks. Ethernet incorporates physical and data link layers. This MAC type is comparable to BACnet/IP in terms of cost and speed but is limited to a single physical infrastructure that does not utilize IP routers. The technology has evolved over the years, what is particularly noticeable in the significant growth of its data rate – from 10 Mbit/s up to 10 Gbit/s, as well as increased compatibility, availability of transfer media, and implementation of wireless LAN technology. [2]

ARCNET

ARCNET stands for Attached Resource Computer Network, which is a LAN technology developed and widely used in the USA for office intranets. The technology is standardized as ATA/ANSI 878.1. It is much slower than Ethernet now and only a limited number of vendors support BACnet using ARCNET. However, ARCNET is deterministic, and therefore suitable for use in industrial communication technology. The benefits of ARCNET, as a token-passing technology, like MS/TP, are the following:

- real-time functionality, which makes it possible to count response time precisely,
- incorporated physical and data link layers,
- bus, tree, star topologies,
- different media options,
- variable data rates up to 10 Mbit/s,
- transmission rate up to 71% of baud rate. [2]

Point-to-Point

Two BACnet devices (half-routers) can exchange messages over the Point-to-Point Protocol (PPP) based on the EIA-232 standard (RS-232). Remote areas that do not have physical Internet access can be reached over a dial-up connection using a modem. The BACnet standard does not define how the physical connection is established. It only specifies data link layer protocols that enable the reliable transfer of data frames over an established physical layer connection. Although a PPP connection is capable of full-duplex operation, it is usually only temporarily available and has a slower transfer rate. [2]

LonTalk

LonTalk is a LAN technology developed as part of LonWorks protocols. Transmission media is based on coaxial, twisted-pair, fiber-optic cable, and radio systems. The transmission speed can be up to 1.25 Mbit/s. However, BACnet and LonTalk protocols are not interoperable. [2]

ZigBee

ZigBee is a wireless mesh network typically used with very low-cost devices. It is usually used as a gateway to ZigBee devices and not as a native BACnet transport. [6]

The most frequently used technologies in BACnet applications are BACnet/IP and BACnet MS/TP. BACnet/IP is primarily used for operator management stations and automation control systems, such as direct digital controllers (DDCs) and programmable logic controllers (PLCs). In contrast, devices based on MS/TP technology are mostly intelligent field devices, for example, variable frequency drives, VAV boxes, pumps, valves, and room controllers. [5]

2.5 Network Layer

The network layer serves as the basis for the transfer of messages between different networks, regardless of the used technology. The network layer permits global addressing, unlike the data link layer that handles only local addresses. BACnet provides a two-byte network number that allows a maximum of 65,535 available subnetworks, which means that a bigger amount of stations within BACnet are identified by the network address.

A BACnet router is a device that joins multiple network types and passes BACnet messages among the network types without changing the message content. Figure 2 shows the example where two different networks – MS/TP and Ethernet – are connected over a router. [2]

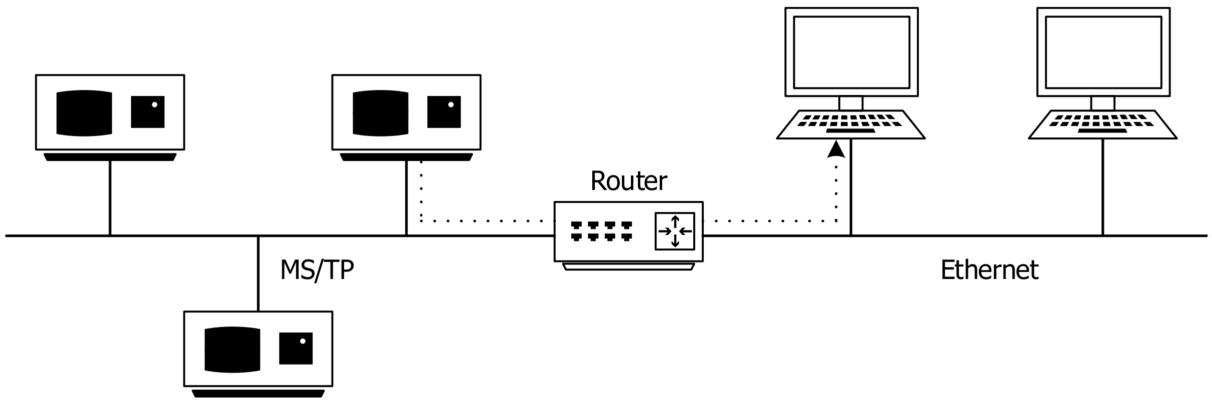


Figure 2 – Two networks connected over a router [2]

A router is a networking device that allows access to computer networks and transfers data packets between them. Based on an internal routing table and the destination address, the router forwards the packet to another network. The Routing Information Protocol defines the rules for the exchange of network topology information between routers. Routers can be standalone devices or may be built into automation devices. [2]

Every BACnet subnetwork connected through routers is uniquely identified by a 16-bit BACnet network number known by the router devices. Routers connecting different BACnet data link layers exchange their routing information on the network layer using special messages, e.g. *Who-Is-Router-To-Network* and *I-Am-Router-To-Network* to discover the active routes in a BACnet internetwork.

Besides the logical addressing using the device-object ID, the physical addressing of BACnet devices is performed using the so-called BACnet MAC address. The BACnet MAC address represents the BACnet network number in the range from 1 to 65,535 plus the MAC address assigned to the device. The length of the destination MAC layer address ranges from one byte (MS/TP) to seven bytes (LonTalk).

As an alternative to the optional manual configuration, the services *Who-Is* and *I-Am* provide a method for dynamic device binding. In response to a *Who-Is* request, the *I-Am* service delivers the logical address (device ID) sent from the physical address. In the event that the message is forwarded by a router, the actual physical address is encoded in the network protocol data unit (NPDU).

Dynamic object binding is provided by the *Who-Has* and *I-Have* services. A device can ask for an object name on the network, and other devices containing this object respond with the relevant object ID. Dynamic device and object binding services are generally used as broadcast messages in the network but can be also transported using unicast addressing. [5]

2.6 Application Layer

The application layer handles the transfer of information related to control, monitoring, requests, and alerts. Application protocol data unit (APDU) consists of a header section and data section. The header defines transport information, e.g. request or acknowledgment. The data section contains coded BAC-net command sequences, e.g. manipulation or request of objects.

The BACnet application layer carries out the tasks connected with the OSI transport layer, which are stored in the header. For instance, "acknowledge" services expect the receiver to reply as soon as the message arrives. The correlation between a client and a server is shown in figure 3. [2]

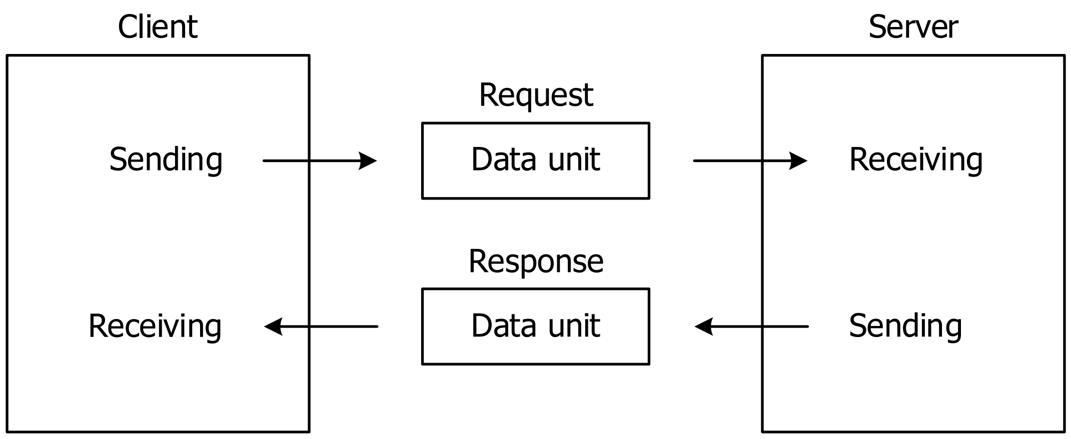


Figure 3 – The data exchange between client and server [2]

When the client sends a request, it expects to receive acknowledgment of receipt from the server within a certain interval. If no response is received, the client sends the message again. After several attempts, the application program is informed. [2]

2.7 BACnet Object Model

The information linked to the physical or virtual data points is decentralized, therefore, BACnet objects provide all the data to the device, unlike the approach used in classical systems, where the management station requested only the present value.

Currently, BACnet supports more than 50 different object types, including analog, binary, and multi-state inputs, outputs, and values. Standard object types are accessed by the *Object_Type* property. Apart from primitive value objects, BACnet also specifies objects with specific functionalities, such as load control, accumulators and pulse converters, structured views, groups, commands, as well as calendars, schedule programs, counter values, lighting output, and channels, etc. For example, life-safety objects may represent fire alarm systems.

All objects contain properties that provide physical and virtual information, like the *Present_Value*, which keeps the current value and is the most important property. Other properties define the engineering unit, alarm limits, state text information, etc. The object properties can be requested using the BACnet services. Each property has a unique ID, assigned property data type, and a conformance code.

The conformance code determines the access permissions and optional presence of the property in a particular BACnet object. Properties are defined as readable (R), writable (W) or optional (O).

BACnet objects are represented in the network as a device-object, where they are stored as an object-list. The object-list contains a list of all objects related to the device. The structure is shown in figure 4. [5]

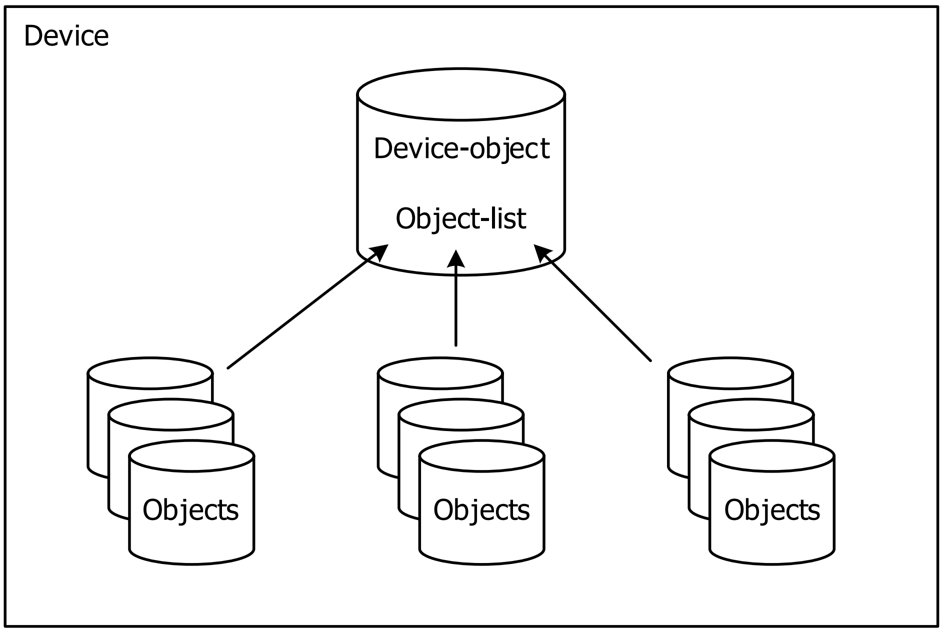


Figure 4 – The structure of storing BACnet objects within the device object [5]

Every object has its unique 32-bit identifier referred to as *Object_Identifier* property, that BACnet uses for addressing the objects. The first 10 bits define an object type, like analog-input, analog-output, analog-value, binary-input, binary-output, binary-value, averaging, device, etc. The rest 22 bits indicate the object instance number.

Numerical *Object_Identifier* properties serve for automatic access to objects. Proper names of objects are more convenient for developers. For this purpose, each BACnet object has an object name referred to as *Object_Name* property. The object name of the device must be unique throughout the whole BACnet network as well.

After sending a message the object identifier is entered in the data portion of the APDU in order to reference the desired object. A device can contain several objects with the same object type. Every BACnet device has exactly one *DEVICE* object representing the device itself with a unique object identifier within the whole network. The *DEVICE* object type defines device properties and the number of available objects. [2]

2.8 BACnet Services

BACnet has 35 application services in total that give access to the devices, objects, and properties. 20 services are used on the network layer for the exchange of information between routers and for secure authentication. The application layer includes services for data access, scheduling, alerts, trend logging, and device and network management.

The most frequently used services are basic ones, like read or write services. More complicated services provide file transfer, device detection in the network, time synchronization, reading range from logging objects, alarm notifications, and some others. Among BACnet services, there are even proprietary services, such as confirmed/unconfirmed private transfer. [5]

Figure 5 illustrates the example of requesting a value from a temperature sensor using the *ANALOG_INPUT* object. The client sends a request to the server using the *"ReadProperty"* service that belongs to the acknowledgment services. *"Property_Identifier"* indicates that the requested item is *"Present_Value"*, and *"Object_Identifier"* is an address of the object of interest. Then the server sends a response with all requested information back to the client. The messages are numbered to guarantee that the responses match the requests. [2]

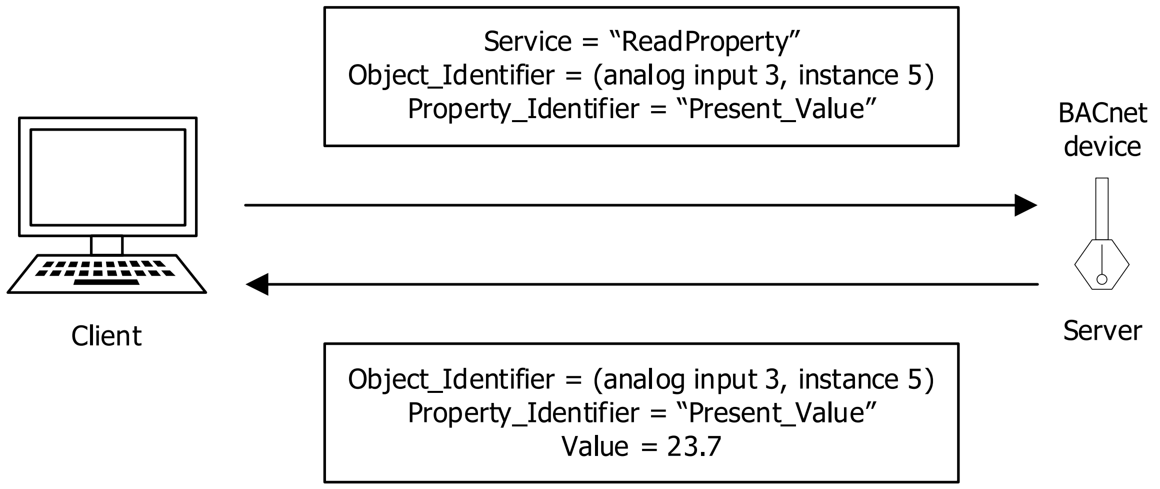

Service = "ReadProperty"
Object_Identifier = (analog input 3, instance 5)
Property_Identifier = "Present_Value"
BACnet device
Client
Server
Object_Identifier = (analog input 3, instance 5)
Property_Identifier = "Present_Value"
Value = 23.7

Figure 5 – Use case of BACnet services to request a value from the device [2]

3 Analysis of Using BACnet in Industrial Control Systems

3.1 System Integration

System integration is a necessity for BAS. The combination of system parts forms the HVAC control systems, and the configuration of devices ensures their proper interaction. Since field devices and management infrastructure are provided by different vendors, automation systems are entirely separate in most buildings. Consequently, cross-domain integration is essential for intelligent buildings.

Room automation is a key area of cross-domain integration. For example, automatic control of window blinds can significantly decrease energy consumption for cooling purposes. A computer-aided management system that has direct access to data from BAS is a beneficial tool for accounting and controlling purposes.

However, examining subsets of system functionality in an integrated system becomes more difficult. This brings additional challenges in fault analysis, debugging, and functionality assessment. Moreover, problems in determining liability may arise in case multiple contractors are working on a single integrated system. These issues need to be approached by selecting the points of interaction between different systems in order to make the flow of control traceable. High-level integration, for which only a limited number of interaction points are established at the highest system level, brings considerable benefits.

A modern building is not achievable relying on one building automation technology. BAS integrates several different systems taking advantage of the specific benefits of every single technology. Vendors of automation equipment support many protocol standards, like BACnet, LonWorks, and KNX. These different technologies join together particularly at the interface between the automation level and management level of BAS. Interoperability has to be reached in order to provide interfacing these technologies by management-level applications. The classical way to achieve this goal is to integrate technology-specific interfaces in the management-level application. However, this approach creates a lock-in to specific technologies and vendors, and future extension is less flexible. A more promising way is to define and agree on a general application model covering the functionality of these underlying systems. [5]

3.2 Applications

Nowadays BASs are not constrained by simple HVAC systems, they have become more complex. Centralized control systems have been superseded by distributed control systems. The implementation of the technical requirements is described by building services. The physical processes that have to be controlled can be very complex and spatially distributed, therefore, a certain division of services is required. Individually controlled units are called zones. Typical zones for building services now range from entire floors to single-window axles. For instance, HVAC controls an entire room, whereas the window axle controls only blinds. A building service may have different requirements for different zones. Areas, such as corridors, elevators, and stairwells, connect several building units and therefore require specific automation functions. For example, the corridor light may depend on room occupancy. A summary of different building service domains can be found in table 3. [1]

Table 3 – Domains of building services [1]

Domain	Typical building services
Climate control	HVAC, humidity, air quality
Visual comfort	Artificial lighting, daylighting (motorized blinds/shutters), constant light control
Safety	Fire alarm, gas alarm, water leak detection, emergency sound system, emergency lighting, closed-circuit television (CCTV)
Security	Instruction alarm, access control, CCTV, audio surveillance
Transportation	Elevators, escalators, conveyor belts
One-way audio	Public address/audio distribution and sound reinforcement systems
Energy management	Peak avoidance
Supply and disposal	Power distribution, waste management, freshwater/domestic hot water, wastewater
Communication and information exchange	IT networks, private branch exchange (PBX), intercom, shared WAN access, wireless access (WLAN)
Miscellaneous special domains	Clock systems, flexitime systems, presentation equipment (e.g. video walls), medical gas, pneumatic structure support systems (for air hoses)

3.3 Building Automation Technologies

Building automation technologies were primarily used for simple control tasks, but they developed into highly complex systems. Today's BASs are crucial for large buildings. They help to save energy and human resources. Additionally, the comfort of facility users is increased, and maintaining safety in buildings is a key asset. The following paragraphs provide an introduction to state-of-the-art technologies used in building automation.

3.3.1 KNX

The installation of lighting control and electrically operated shading systems is usually performed by electrical installation specialists. To ensure the sustainable future of this industry, many European companies have developed system components that allow the automatic implementation of comfort and convenience functions.

In 1990, major producers of electrical installation technology formed the European Installation Bus Association (EIBA), known as the KONNEX Association. The purpose was to design a bus system that would meet these requirements – the European Installation Bus (EIB). EIB is now referred to as KNX. KNX is a standardized bus system that allows data transmission between different devices and systems from several manufacturers.

KNX implements all necessary building control functions and can be programmed and commissioned by qualified engineers. The transmission rate is low but sufficient for the transfer of switching and controlling commands.

KNX was not designed for controlling operational systems and, as a result, cannot be used for transferring many analog signals. However, it has been very successful in Europe due to the fact that it is easy to install and maintain. The installation of KNX for lighting control is depicted in figure 6. [2]

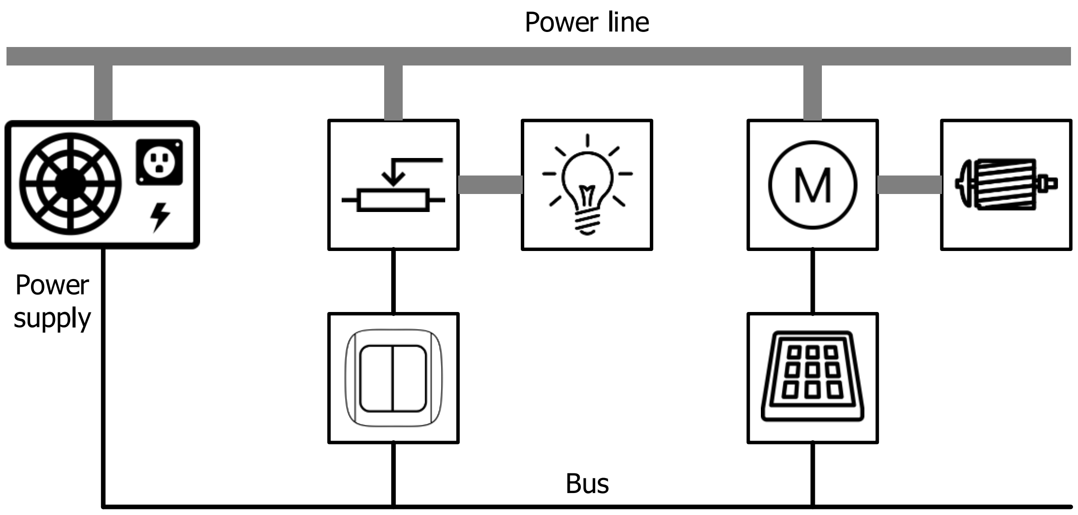


Figure 6 – KNX used for lighting control [2]

3.3.2 LonWorks

In order to control operational systems, a broad range of measured values, setpoints, and other parameters need to be managed. The software applications set higher demands on the processor and the software engineer. The LonWorks technology (LON) was developed to meet these requirements. Its structure is presented in figure 7.

LON is a universal system for automation designed by the American company Echelon. The scope declares that it can be used in centralized DDCs and in decentralized building control components. The LonMark Interoperability Association certification guarantees the compatibility of devices from different vendors.

In the field of lighting control, LON is the main competitor of KNX in Europe. LON has a great benefit over KNX in applications when there is a need to connect room automation to the operational systems through the same bus system. [2]

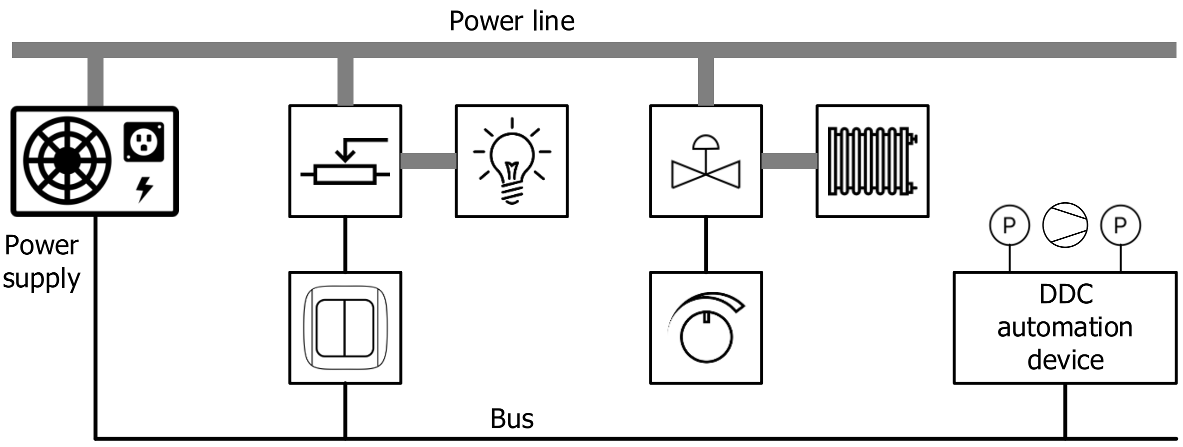


Figure 7 – System structure of LonWorks [2]

3.3.3 BACnet

Extensive building automation systems in hospitals, universities, or administration buildings, often have several operating stations. In case of the need to extend the building by constructing an additional wing that also has an extensive building automation system, it would require connecting these systems to the existing control computer. To do this, American building control engineers developed a standardized communication system called Building Automation and Control Network (BACnet). The way remote buildings exchange information is shown in figure 8. BACnet has an object-oriented structure with a large number of functions. BACnet is customized for building applications and can be used at all levels of building automation. [2]

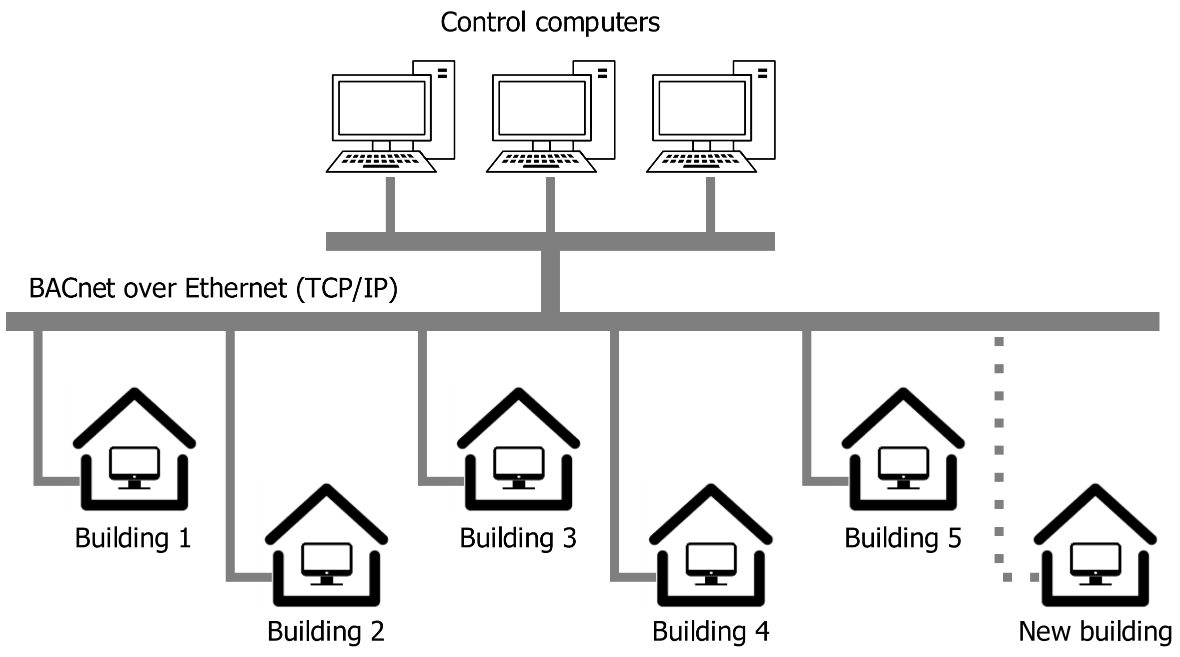


Figure 8 – Data exchange between remote buildings [2]

3.3.4 Other technologies

Modbus

Modbus is an open standardized communication protocol designed by Modicon in 1979. It was previously used for the communication among programmable logic controllers (PLCs) in industrial applications but became essential in the building automation area, especially for HVAC systems, DDCs, and room control. Modbus is a simple application layer protocol that supports different media. [5]

ZigBee

ZigBee is an open wireless standard for low-rate and lossy networks. It is primarily used for home and building automation, including smart energy applications. The ZigBee standard is defined and maintained by the ZigBee Alliance. ZigBee provides a secure and flexible communication infrastructure and determines a high-level information modeling scheme. It originally supports multi-hop communication and has self-healing capabilities. In comparison with other wireless technologies, ZigBee has fewer hardware requirements and requires less power. [5]

DALI

A significant part of building automation concerns lighting. Apart from simple switching, demand-specific and energy-efficient lighting is an important issue in modern buildings. Digital Addressable Lighting Interface (DALI) as adopted in IEC 60929 is an open interface specification that addresses this particular scope. This interface provides a manufacturer-independent integration of lighting systems. [5] [7]

OPC

In the 1990s, different manufacturers of HMI and SCADA software (later called *OPC Foundation*) defined the OPC standard. This standard provides services for reading, writing, and monitoring of control data in automation systems. The initial OPC standard was subsequently renamed in OPC Data Access (OPC DA). In addition to the first publication, many other OPC specifications have been specified, such as OPC Alarms&Events (OPC A&E), OPC Historical Data Access (OPC HDA), OPC Batch, OPC Security, OPC Data Exchange (OPC DX), OPC Compliance Tests, and OPC and XML. [5] [8]

3.4 The origin of BACnet

Before the advent of BACnet, many proprietary solutions from different vendors prevailed in building automation. As a result, for example, a heating controller from vendor A could not communicate with a control center software from vendor B. The communication with air conditioning, ventilation, lighting, and alarm systems was not possible since they were designed separately and did not have interfaces.

Building developers had to solve this problem. They either had to abandon the centralized control of automation systems or purchase a complete solution from one manufacturer. With the further development of an installation, they could not substitute for potentially cheaper and better components.

For this reason, there was an urgent need for a standardized data communication protocol that would provide the opportunity for diverse automation and control components in a building to communicate among themselves, providing interoperability and independence from manufacturers.

BACnet, as an open multi-vendor communication protocol, allows components from different vendors to interact, thereby contributing to the functional network, increasing comfort, saving energy, improving security, and reducing costs.

A similar situation was observed in computer networks with the development of the Internet. Despite the fact that other proprietary protocols exist, open protocols such as the Transmission Control Protocol/Internet Protocol (TCP/IP) have been standardized. Nowadays, various hardware and software components are available from different manufacturers. [2]

3.5 Areas of Use in BAS Layers

Commercial buildings are usually very large. Individual parts of a commercial building have different requirements in terms of heating, ventilation, and air conditioning. Therefore, the systems in different parts of the building are typically decentralized and controlled by remote stations. The control center records the data from the distributed stations, such as measurement values, operating states, alerts. It allows consistent cross-system access to all building data and control functions. This distributed system in building automation can be represented by a three-tier automation hierarchy, as shown in figure 9. [2]

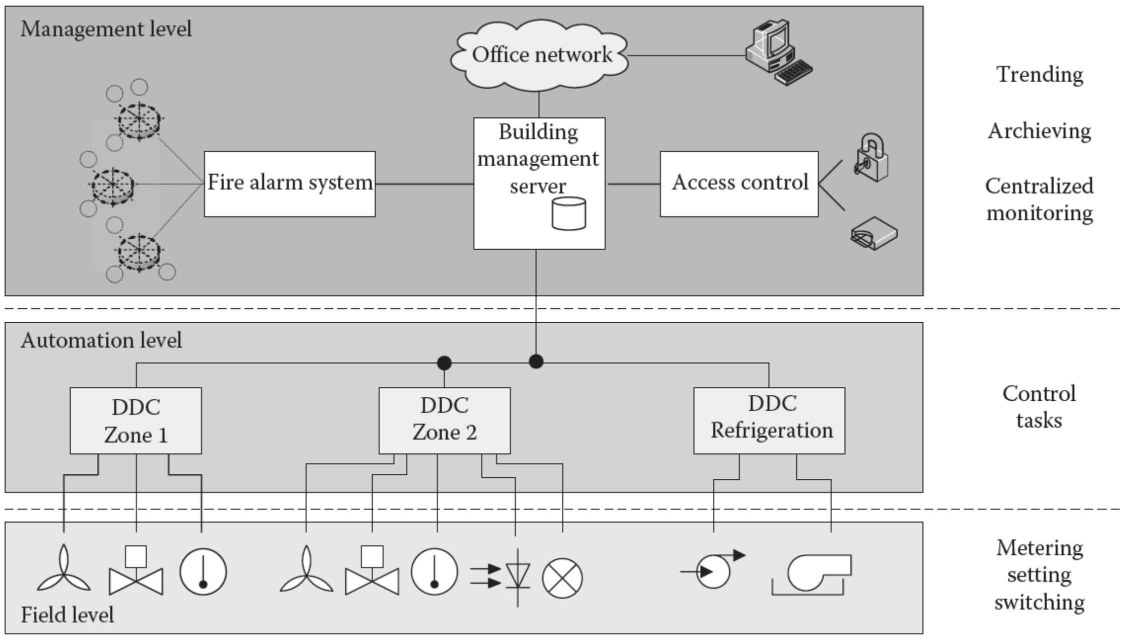


Figure 9 – Three-tier model in building automation [5]

The field level consists of individual sensors (e.g. temperature sensor, switch sensor), actuators (e.g. control valves, drives, relays), and control panels. At the automation level, direct digital controllers (DDCs) connect the devices from the field level. Conversely, the DDCs are connected to a building management system server at the management level, which serves to monitor all sections of the network, manage the individual systems, and analyze faults. For instance, users in a network or office intranet can access the building management system server.

Generally, the transmission rate at the management level must be considerably faster than at the field level, to transmit process data collected from all systems. On the other hand, response time must be quicker at the field level. It should not exceed ten milliseconds for a signal coming, for example, from a light switch to a lamp to turn it on or off.

BACnet was initially designed to be used at the automation and management levels of the three-tier model. Nowadays, it has found its use in a wide variety of automation applications.

BACnet may be used at all levels of building automation but it is particularly suitable for management functions. It is primarily used in larger installations as a superordinate system with LonWorks and KNX present at the field level. [2]

4 Specification and Design of the Communication System

4.1 Selection of the AC500 PLC

The range of AC500 modular PLCs provides solutions for small, medium, and high-end applications. It includes: [9]

- AC500-eCo – flexible and economical configuration for automation solutions in smaller applications,
- AC500 and AC500 V3 – designed for complex, high-speed machinery and networking solutions,
- AC500-XC and AC500-XC V3 – variant for extreme conditions with extended operation temperature, immunity to vibration and hazardous gases, use at high altitudes and in humid environments
- AC500-S and AC500-S-XC – safety PLC (SIL3, PL e) designed for safety applications involved in factory, machinery, or process automation area.

The AC500 PLC was chosen for this project, which has the following characteristics: [10]

- High performance (time per binary instruction) 0.0006 … 0.06 µs
- Large program memory up to 160 MB
- From 8 to 80 000 I/Os
- Many communication possibilities (Ethernet, OPC DA, BACnet, EtherCAT, etc.)

AC500-S, AC500-XC, and AC500-S-XC are also possible for use in this project since they support the BACnet B-ASC profile and OPC DA, which are required for the implementation.

Processor module series PM583-ETH was chosen for this project, which has the following characteristics:

- CPU performance (time per binary instruction) 50 ns
- Program memory 1024 kB
- Powered with 24 V DC
- BACnet B-ASC server
- OPC DA server
- Process control objects library (PCO) for DCS integration

The overview of the AC500 PLC with processor module PM583-ETH is shown in figure 10.

Figure 10 – The overview of the AC500 PLC with processor module PM583-ETH [11]

Terminal base TB541-ETH with 4 coupler slots is used in the configuration, but any other series is possible since the implementation of BACnet does not require an additional communication coupler.

I/O modules are not used in the laboratory installation described in the project, because it is created for testing of BACnet communication. Thus, the selection of I/O modules is not included.

System configuration of the AC500 PLC is realized in the Automation Builder engineering suite. Implementation of the SW application is performed in the CoDeSys programming tool, which is integrated into Automation Builder.

4.2 Selection of the Network Type for BACnet Communication

The slot for the processor module of the AC500 PLC includes an Ethernet port that allows easy integration of the BACnet/IP protocol. BACnet/IP uses Ethernet for communication between servers and clients. BACnet UDP port is configured to 47808 (0xBAC0).

In order for a device to be classified as a standard BACnet device, it must comply with a set of BIBBs defined for this specific type of device. The listing of required BIBBs for each device classification is called a Device Profile. The AC500 PLC is classified as a device that must comply with the B-ASC profile. BIBBs required for the B-ASC device profile are listed in table 4. [6]

Table 4 – BIBBs required for B-ASC device profile

Type	BIBB	Description
Data Sharing	DS-RP-B	Executes a ReadProperty request
	DS-WP-B	Executes a WriteProperty request
Device & Network Management	DM-DDB-B	Executes a Who-Is request
	DM-DOB-B	Initiates an I-Am response
	DM-DCC-B	Executes a communications control request

The “ReadProperty” and “WriteProperty” BIBBs are straightforward since they involve data sharing, but in both cases, the device only responds to a request read or write. The Device & Network Management BIBBs are also limited in that the device only responds to requests. Therefore, it means that the device functions as a server and not as a client.

Since the AC500 PLC can be configured as a BACnet server only, a client is required to request the data from PLC. All BACnet servers and devices have to be connected to the same network. For this purpose, the switch MOXA EDS-408A-SS-SC with 6 Ethernet ports is used, which is shown in figure 11.

Figure 11 – The overview of the MOXA Ethernet switch EDS-408A-SS-SC [12]

4.3 Selection of the BACnet Client

According to the BACnet B-ASC profile, both PLCs can be configured as a BACnet server only. In order to design and implement BACnet communication between two PLCs AC500, a BACnet client is required. For this purpose, the CP600 panel was selected to act as a gateway in the communication between two PLCs.

The range of CP600 control panels provides HMI functions for a wide range of applications. The type of panel selected for the project is CP661-WEB, which has the following characteristics: [9]

- Display size 12.1”
- Resolution 800x600 pixels
- Processor type ARM Cortex A8: 1 GHz
- Operating system Microsoft Windows CE 6.0 Core
- Application memory up to 60 MB
- Supported interfaces – Ethernet, USB Host, RS-232, RS-485, RS-422
- Power supply 24 V DC

The panel is shown in figure 12.

Figure 12 – The overview of the CP661-WEB panel [13]

The panel configuration is realized in the engineering tool PB610 Panel Builder, which is integrated into the Automation Builder engineering suite.

4.4 Selection of the HMI/SCADA System

The 800xA System was chosen as a development environment to create an operator workplace including a user interface. It is a Distributed Control System that allows the operation and configuration of continuous and batch control applications. The control environment supports such functionalities as graphic displays, alarm management, history trending, redundancy, reports, and many others. [14]

The Engineering Environment includes Engineering Workplace used for the system configuration, Operator Workplace that allows the operator to watch the system in real-time, Graphics Builder for creating graphical representation, and many others.

4.5 Selection of the Protocol for Communication with the Operator Workplace

The 800xA System consists of several key server nodes, which communicate with each other through a backbone network using the RNRP protocol. Servers run software that provides system functionality, Workplaces run software that provides various forms of user interaction. The Aspect Server provides services related to object management, names, security, etc. The Connectivity Server provides access to the controllers and other data sources throughout the network.

In this project, as a small application, a Single Node System is implemented. It combines Aspect Server, Connectivity Server, and Workplace, which all reside on a single PC. The HW configuration of this system is presented in figure 13. [15]

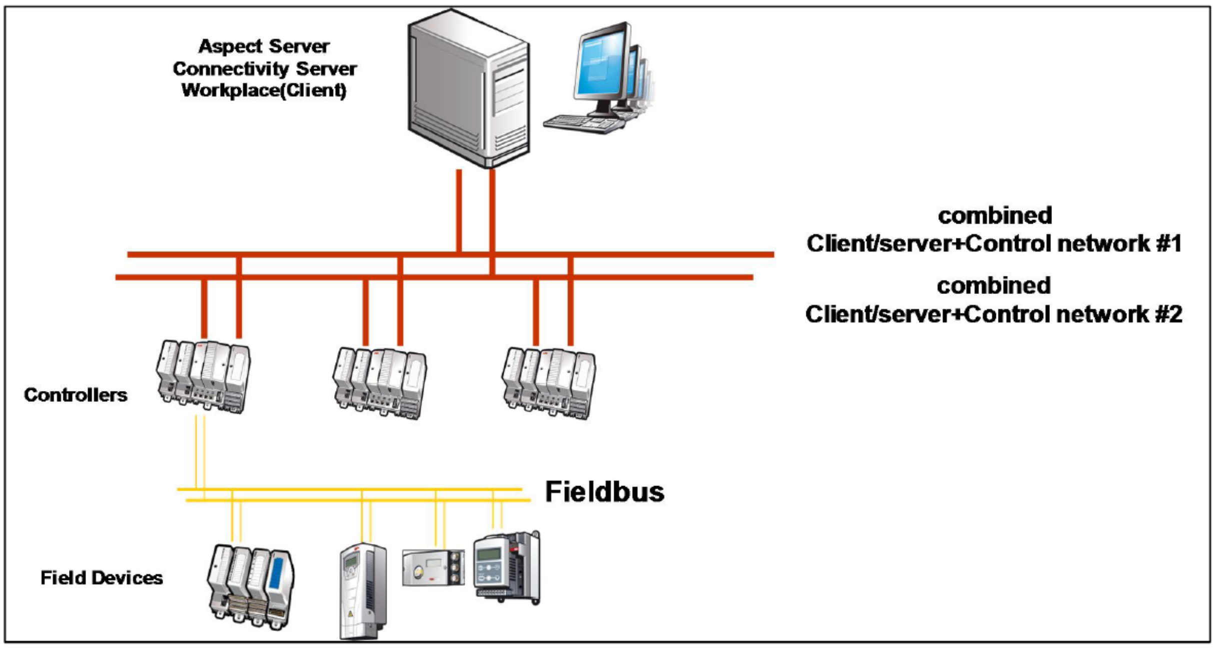


Figure 13 – Single Node System 800xA

OPC Data Access protocol was chosen for the communication between the 800xA System and AC500 PLC, where the 800xA System acts as a client and the AC500 PLC is a server.

4.6 Required Libraries and Additional Tools

Automation Builder environment contains libraries and additional tools useful for easy integration. In this project, the PS565 BACnet-ASC library package is used. The library allows integration of the AC500 CPU into a BACnet network and the data exchange between the PLC and other devices connected to the BACnet network. The library provides the BACnet B-ASC profile that enables the AC500 CPU to act as a server. The CPU receives client requests, performs them, and reports back the results. It supports the use of BACnet/IP communication protocol for a device with onboard Ethernet.

Figure 14 provides a general overview of how B-ASC library elements will be connected to the application and I/O channels. [16]

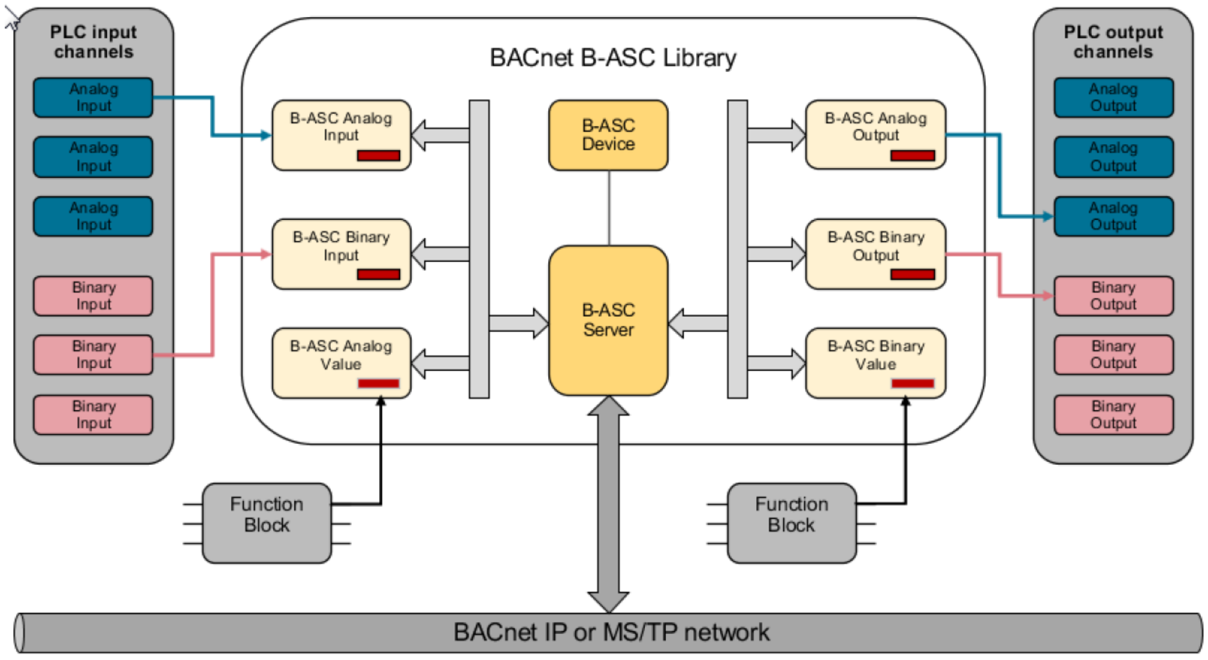


Figure 14 – Integration of the B-ASC library elements to the application and I/O channels

The library contains the following function blocks:

- BACnet server
- BACnet device
- Analog input
- Analog output
- Analog value
- Binary input
- Binary output
- Binary value

The B-ASC Server FB encapsulates the BACnet protocol stack, which is responsible for receiving and transmitting BACnet messages via the network. Servicing the request messages in detail – providing access to the data – will be part of the B-ASC library BACnet objects.

B-ASC Device FB describes the BACnet device representation at the network.

Physical inputs and outputs could be connected to BACnet objects for inputs and outputs using B-ASC library function blocks. Function blocks or other kinds of variables of the PLC program could be connected to BACnet value objects.

To comply with the B-ASC profile, the following services are supported:

- Who-Is / I-Am
- Who-Has / I-Have
- ReadProperty
- WriteProperty
- DeviceCommunicationControl

The software package CoDeSys OPC Server 3.5 supports OPC connection to the AC500 PLC. It contains an OPC Configurator tool that allows configuration of the PLC as an OPC server.

4.7 Purpose of the Communication

Two AC500 PLCs participate in the BACnet communication. One of the PLCs represents the Master Coordinating Controller (MCC), which is responsible for the control of all related areas. Each area has a local dedicated controller called Area Controller (AC), which collects the data from connected devices and sends them to the MCC.

Controlled and monitored devices are located in the data hall. In this project, they comprise eight air handling units, eight cold aisle temperature and humidity sensors, and three entrance doors. The devices are listed in table 5.

Table 5 – Device List

Raw Name	Full Name	Type	Location
AHU-A1	DTB001_DH1_AHU-A1	Air handling unit	Data Hall 1
AHU-A2	DTB001_DH1_AHU-A2	Air handling unit	Data Hall 1
AHU-A3	DTB001_DH1_AHU-A3	Air handling unit	Data Hall 1
AHU-A4	DTB001_DH1_AHU-A4	Air handling unit	Data Hall 1
AHU-B1	DTB001_DH1_AHU-B1	Air handling unit	Data Hall 1
AHU-B2	DTB001_DH1_AHU-B2	Air handling unit	Data Hall 1
AHU-B3	DTB001_DH1_AHU-B3	Air handling unit	Data Hall 1
AHU-B4	DTB001_DH1_AHU-B4	Air handling unit	Data Hall 1
ED-01	DTB001_DH1_ED-01	Entrance door	Data Hall 1
ED-02	DTB001_DH1_ED-02	Entrance door	Data Hall 1
ED-03	DTB001_DH1_ED-03	Entrance door	Data Hall 1
THS-CA-A1	DTB001_DH1_THS-CA-A1	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-A2	DTB001_DH1_THS-CA-A2	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-A3	DTB001_DH1_THS-CA-A3	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-A4	DTB001_DH1_THS-CA-A4	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-B1	DTB001_DH1_THS-CA-B1	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-B2	DTB001_DH1_THS-CA-B2	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-B3	DTB001_DH1_THS-CA-B3	Cold aisle temperature and humidity sensor	Data Hall 1
THS-CA-B4	DTB001_DH1_THS-CA-B4	Cold aisle temperature and humidity sensor	Data Hall 1

Table 6 provides a summary of all used device types with a detailed description of all available signals. Type AIS stands for analog input signal, DIS – digital input signal.

Table 6 – Signal List

Device Type	Signal			
	Signal Name	**Tag Name**	**Type**	**Unit**
AHU	Start/Stop	Start	DIS	
	Mode of Operation	OpMode	AIS	
	General Alarm	GenAlm	DIS	
	Supply Fan Status	SFStatus	DIS	
	Off	Off	DIS	
	In Auto Mode	InAuto	DIS	
	Supply Air Temperature	SATemp	AIS	degC
	Supply Air Humidity	SAHumid	AIS	%RH
	Return Air Temperature	RATemp	AIS	degC
	Return Air Humidity	RAHumid	AIS	%RH
	Supply Fan Pressure Alarm	SFPressAlm	DIS	
	Leak Detection Alarm	LeakDetAlm	DIS	
	Supply Air Damper Status	SADStatus	DIS	
	Return Air Damper Status	OutADStatus	DIS	
	Hot Aisle Differential Pressure	HADPVal	AIS	Pa
Door	Status	Status	DIS	
THS	Temperature	TempVal	AIS	degC
	Temperature Status	TempStatus	DIS	
	Humidity	HumidVal	AIS	%RH
	Humidity Status	HumidStatus	DIS	

5 Configuration of the Installation for Testing of the Communication System

5.1 Logical Diagram of Communication System Architecture

A logical diagram of communication system architecture is presented in figure 15. In the provided communication system, field devices are represented by air handling unit (AHU), temperature and humidity sensor (THS-CA), and electric door sensor (ED). These field devices are simulated by a signal simulation on Area Controller. The area controller (AC) handles the communication with field devices and provides the information to the master coordinating controller (MCC) through a gateway represented by the control panel (CP661). For the purpose of testing this project as a small application, Engineering Workplace (EWP) and Operator Workplace (OWP) were merged to a Single Node System. Upload of configuration and application files is provided by Engineering Workplace. Operator Workplace serves as OPC client, which is running on a virtual machine.

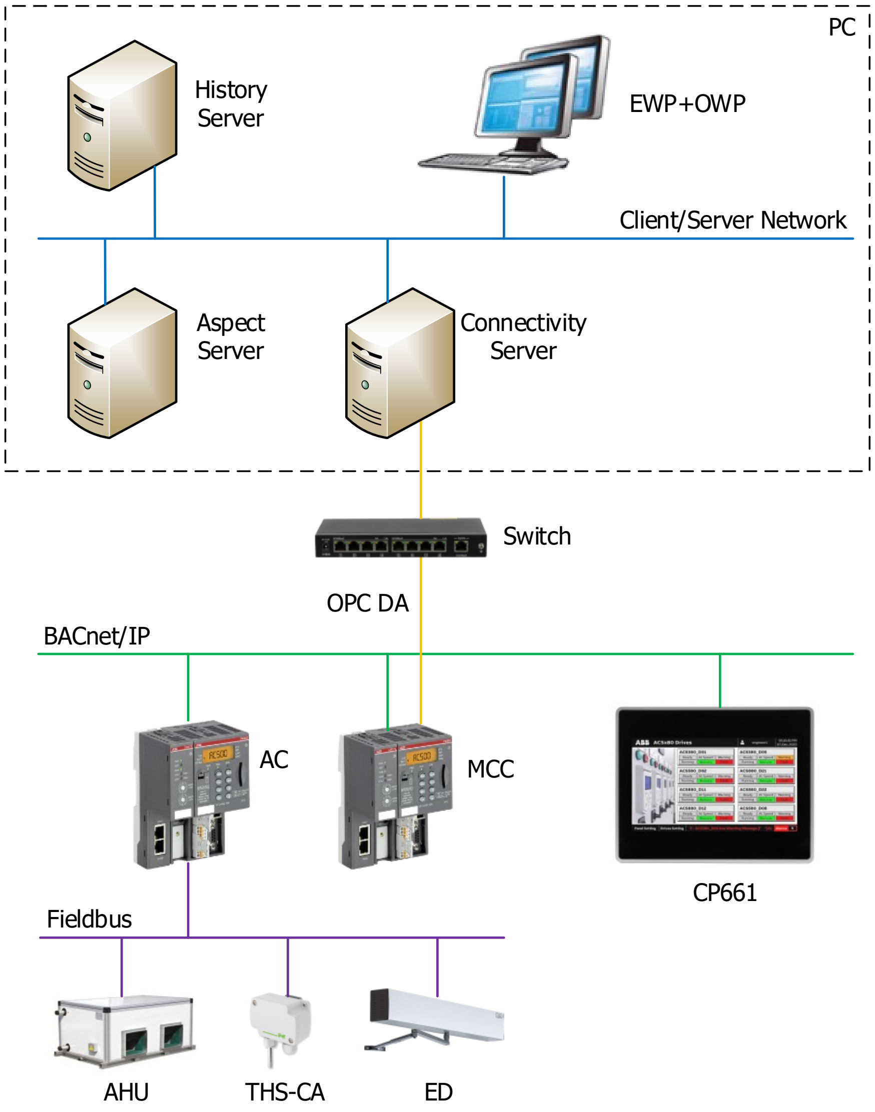


Figure 15 – Logical diagram of system architecture

5.2 Physical Diagram of Communication System Architecture

A physical diagram of communication system architecture is presented in figure 16. The physical devices are connected to the communication switch forming a star topology. Field devices are not included in the physical diagram because their behavior is provided by signal simulation inside the Area Controller program. A single PC contains a virtual machine that comprises the functionality of all servers of the 800xA System and all applications required for engineering and monitoring of the controlled system.

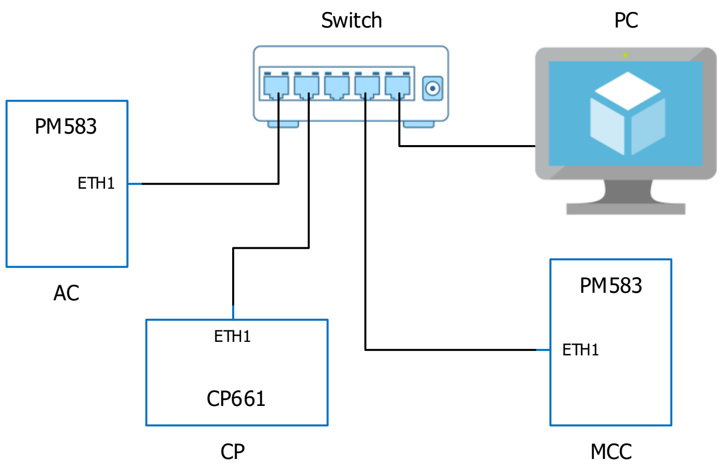


Figure 16 – Physical diagram of system architecture

5.3 IP Addresses of the Devices in the Network

Table 7 provides the list of IP addresses of connected devices in the network.

Table 7 – The list of IP addresses of connected devices in the network

Device name	Type of device	IP address	Subnet mask
PM583-ETH	AC500 MCC – BACnet server	172.16.8.29	255.255.255.0
PM583-ETH	AC500 AC – BACnet server	172.16.8.30	255.255.255.0
CP661-WEB	Control panel – BACnet device	172.16.8.31	255.255.255.0
PC	Physical PC	172.16.8.201	255.255.255.0
EWP+OWP 800xA	Engineering Workplace and Operator Workplace	172.16.8.202	255.255.255.0

5.4 Laboratory Installation

Figure 17 shows laboratory installation for testing of the communication system between two programmable controllers using BACnet. The demo stand contains the control panel CP661-WEB located at the top, below are placed two PLCs with I/O modules and communication couplers, the MOXA switch is installed at the bottom of the stand.

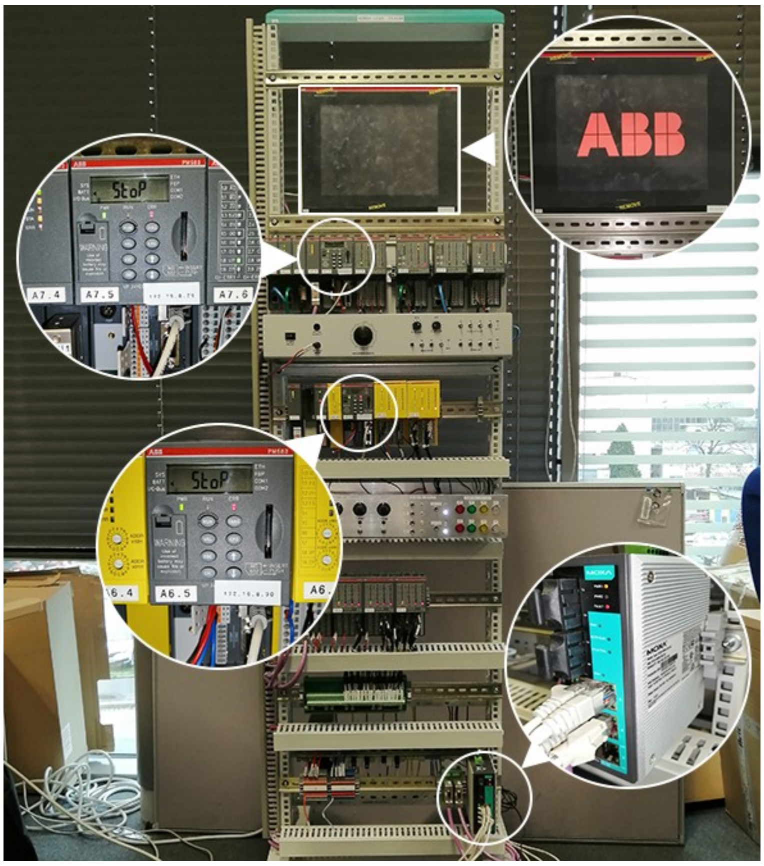


Figure 17 – Laboratory installation for testing of the communication system

5.5 BACnet Objects IDs

The maximum number of BACnet objects per CPU is limited by

- the CPU load or cycle time of the task, in which the BACnet is handled
- the speed of the CPU itself

For the PM583 used in the project, the values for the maximum number of BACnet object are the following:

- 450 objects with the task time of 100 ms
- 600 objects with the task time of 150 ms
- 750 objects with the task time of 200 ms
- 850 objects with the task time of 250 ms

The BACnet configuration requires each object to have a unique ID number. The ID of CP661-WEB panel is configured to 40500. The PLCs AC and MCC have IDs 40100 and 40300 respectively. The information about 155 signals is transferred from AC to MCC. These objects which belong to the AC have IDs from 40101 to 40255, the objects on the side of the MCC have IDs from 40201 to 40455.

6 Implementation of the Control Application for Data Transfer

6.1 Requirements of the Application

The first step in designing the control application was to analyze the functional requirements imposed on the device.

The BACnet protocol determines several functional requirements, which the control application must comply with: [6]

- Every BACnet device shall contain a device object that defines certain device information, including the device object identifier or instance number.
- An object identifier must be a 32-bit binary number containing a code for the object type and the object instance number.
- An object identifier must be field-configurable to be unique across the entire BACnet network.
- Each standard object type must have a collection of properties that define the object. Each property must include at least a name and a value.
- BACnet services shall be used as formal requests for interconnection between two BACnet devices.
- In situations where there is a need to synchronize the actions of multiple clients potentially writing to the same property, a command priority from 1 to 16 must be provided with write property, where 1 is the most important and 16 is least important.

The control application conforms to the B-ASC device profile. The library provided by ABB, which is used for the implementation, corresponds to this profile. In order to fulfill the communication requirements, the following conditions must be met by the control application:

- BASC_SERVER FB, which enables the BACnet server functionality on the AC500 PLC must be turned on the whole time and do exist only once in the whole project.
- The number of BACnet objects should not exceed the amount where 75% of the CPU load limit is reached.

All B-ASC blocks may be used in a single POU assigned to a single IEC task. However, server and device function blocks do not need to run with a fast cycle time. Their role is to initialize the library and update outputs with static information. For this reason, function blocks BASC_SERVER and BASC_DEVICE are placed in one task with low cycle time.

The function blocks for analog and binary values update their present values while being called. Thus, these function blocks are called by a different task with faster cycle time.

The values for signals coming from connected devices are simulated in the main program. Since the values don’t change very frequently, the main POU is called by a task with a low cycle time. In order to avoid continuous updating of values that haven’t changed since the last time they were sent, a Change of Value (COV) scheme is used. With COV, the client subscribes to the server device that contains the value of interest and indicated a minimum amount of change that is required for notification. After the subscription is accepted, the server device itself watches the object property. If the value changes by more than the trigger amount since the last reported value, then the server issues a COV notification message that reports the change. In this way, only changing values get reported, thus saving a lot of network traffic.

6.2 Algorithm of the Application

The structure of the control application for data transfer from AC to MCC via BACnet was designed in several POUs based on the analysis of functional requirements. A block diagram of the structure of the control application is shown in figure 18.

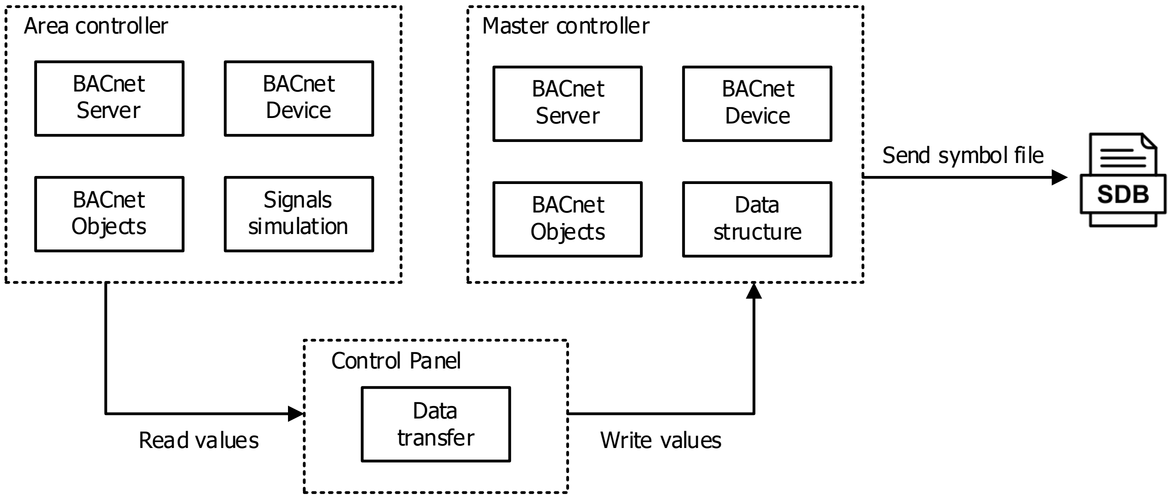


Figure 18 – Block diagram of the structure of the control application

The concept of dividing a control application into several tasks can be demonstrated by the UML timing diagram (figure 19).

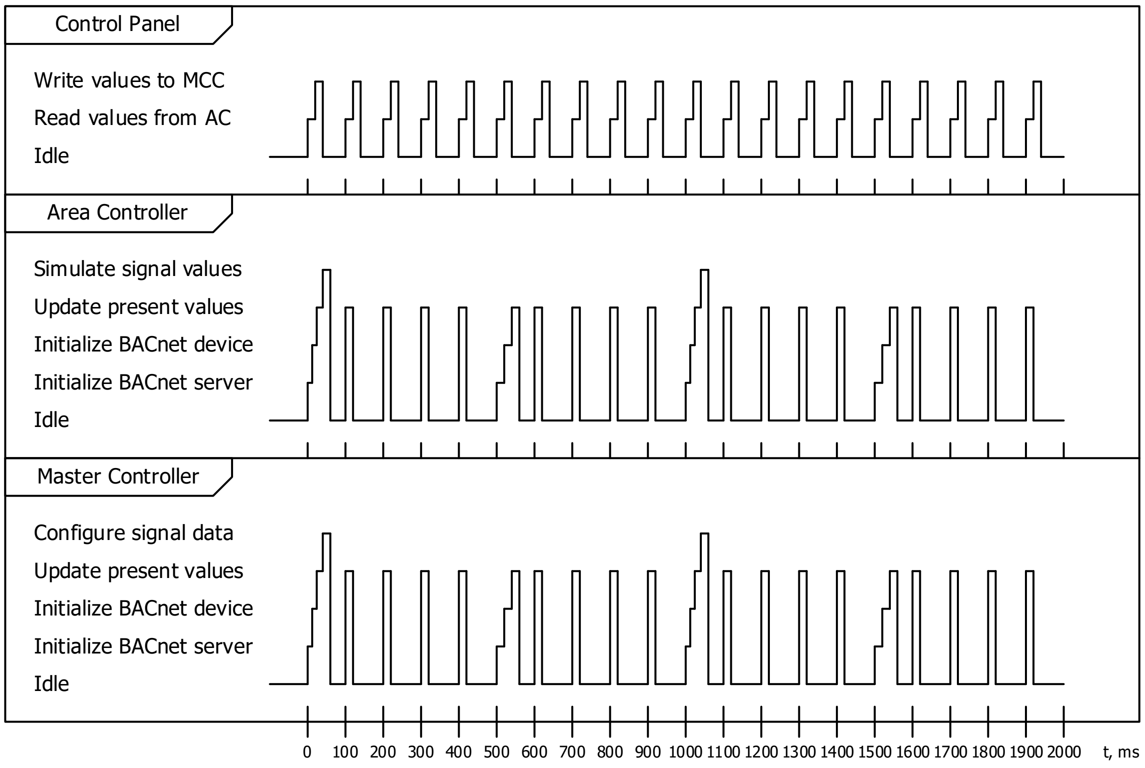
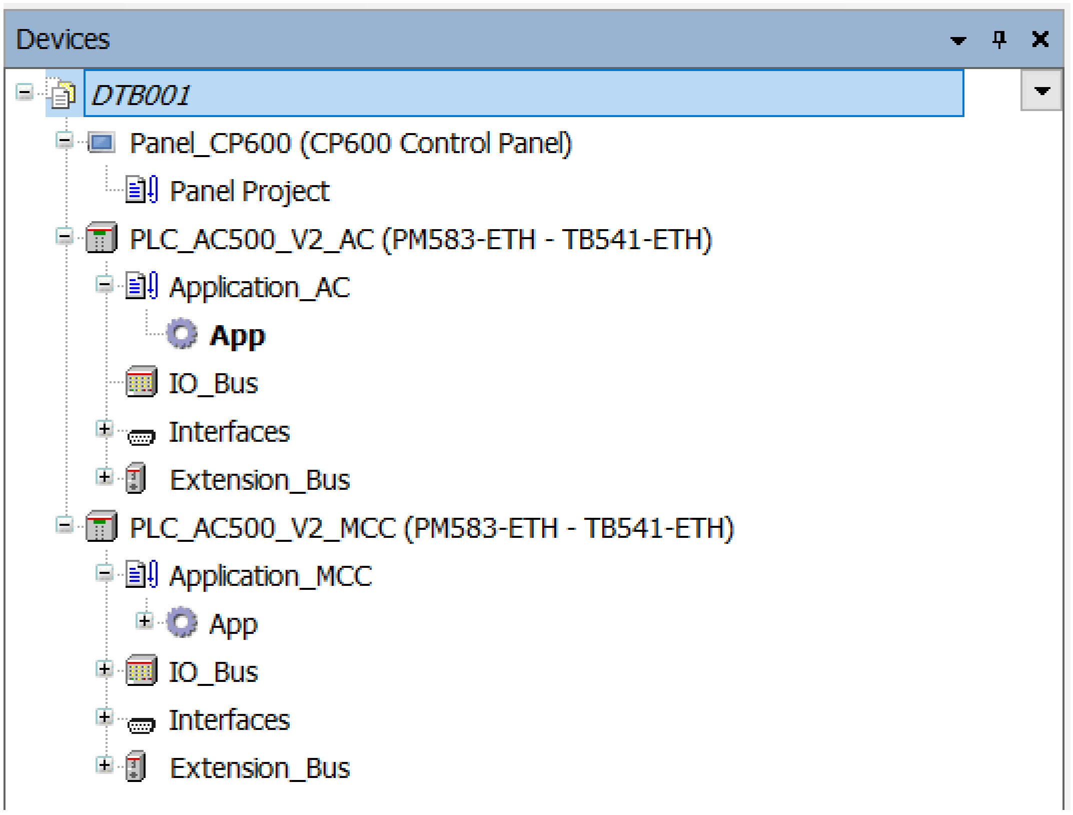


Figure 19 – UML timing diagram of communication between PLCs and control panel

6.3 Configuration of two PLCs AC500 and CP600 Panel

The first step in the implementation of the control application was to create a configuration of all devices participating in the BACnet communication. The specified devices were selected in Automation Builder as presented in figure 20. It is necessary to specify the correct IP addresses of the devices in the configuration.



Devices

- DTB001
  - Panel_CP600 (CP600 Control Panel)
    - Panel Project
  - PLC_AC500_V2_AC (PM583-ETH - TB541-ETH)
    - Application_AC
      - **App**
    - IO_Bus
    - Interfaces
    - Extension_Bus
  - PLC_AC500_V2_MCC (PM583-ETH - TB541-ETH)
    - Application_MCC
      - App
    - IO_Bus
    - Interfaces
    - Extension_Bus

Figure 20 – List of devices in Automation Builder

Explanatory notes to the structure in figure 20:

- DTB001 is the name of the project
- Panel_CP600 is the name of the panel
- Panel Project is the name of the application for the panel
- PLC_AC500_V2_AC is the name of the area controller
- Application_AC is the name of the application for the area controller
- PLC_AC500_V2_AC is the name of the master controller
- Application_MCC is the name of the application for the master controller

I/O modules, interfaces, and other communication modules are not added to the configuration, since they are not used in the project.

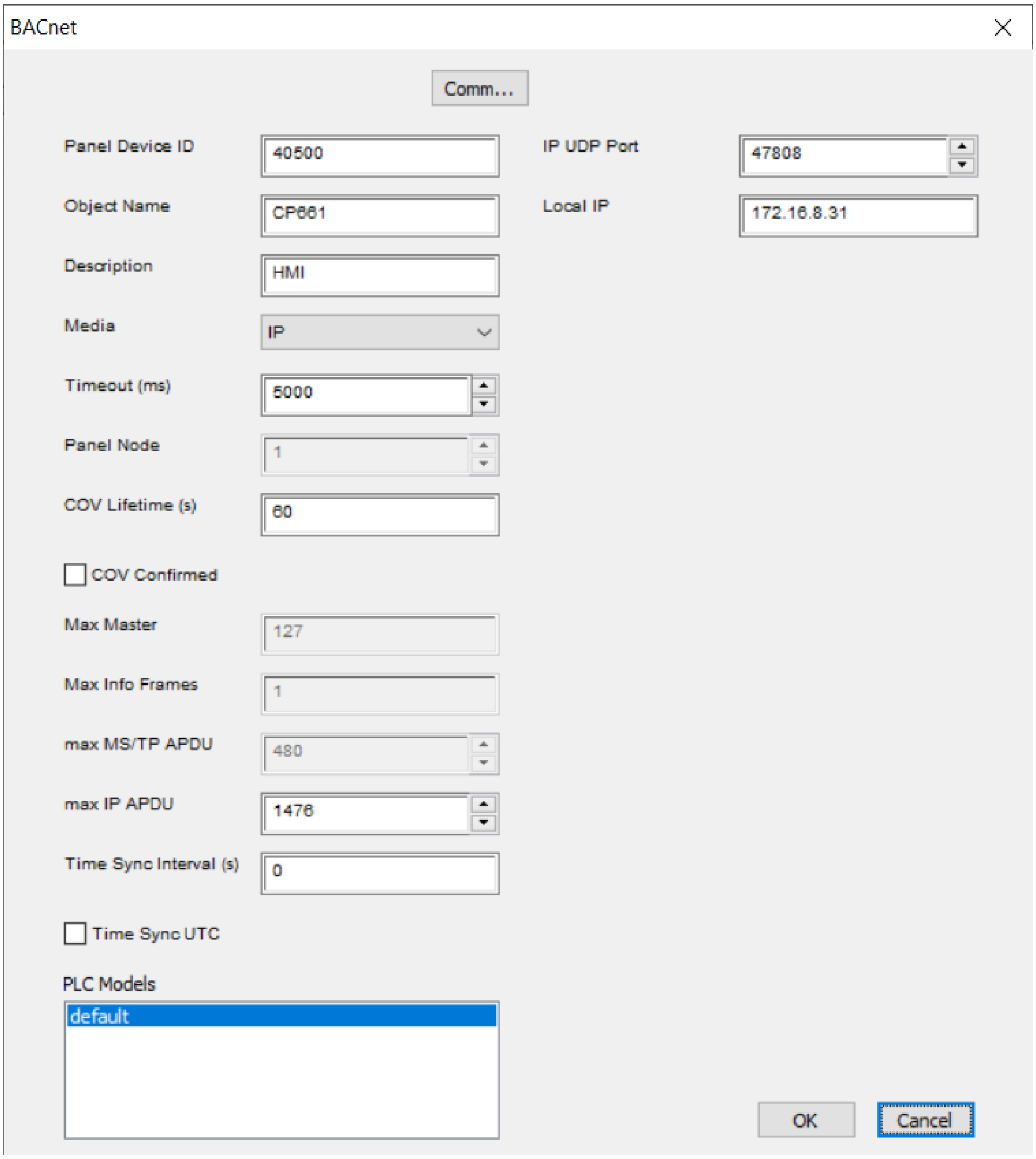
6.4 Application for the Panel

The configuration of the CP661 control panel was prepared in the engineering tool PB610 Panel Builder, which is integrated into the Automation Builder engineering suite. In this environment, the

BACnet protocol was configured, all BACnet objects were defined, and also the information for data transfer was prepared.

6.4.1 BACnet Protocol Settings

The BACnet protocol is configured according to the required parameters, as presented in figure 21. The configured properties are explained below.



Panel Device ID	40500
Object Name	CP661
Description	HMI
Media	IP
Timeout (ms)	5000
Panel Node	1
COV Lifetime (s)	60
COV Confirmed	☐
Max Master	127
Max Info Frames	1
max MS/TP APDU	480
max IP APDU	1476
Time Sync Interval (s)	0
Time Sync UTC	☐
PLC Models	default
IP UDP Port	47808
Local IP	172.16.8.31

Figure 21 – Configuration of BACnet protocol in the panel project

Panel Device ID: A unique BACnet device ID for the panel. The CP661-WEB panel has ID 40500.

Object name: A unique name of the panel.

Description: Additional information about the device.

Media: The media type is set according to the network type.

Timeout (ms): This property specifies the time that the panel will wait for an expected response from the device before retrying or continuing to the next request. The valid range is from 100 to 9999 milli-seconds.

COV Lifetime (s): Specifies the lifetime of COV subscriptions. The valid range is 10 to 3600 seconds. The default setting is 60.

Max IP APDU: This property specifies the total length or number of bytes of message segments that the panel will accept. The panel attempts to read the maximum APDU length allowed by the target device on startup and use the smallest of the local or remote limits when sending requests. The default value is 1476 bytes, which is the largest length allowed for BACnet/IP.

Time Sync Interval (s): Specifies the period of the automatic time synchronization.

IP UDP Port: Specifies the local UDP port, as a decimal value, that the panel binds for all communica-tions on the channel. It is also the remote port to which all messages sent to devices on this channel are addressed. The default setting is 47808 (0xBAC0).

Local IP: The IP address in the local network that is assigned to the panel.

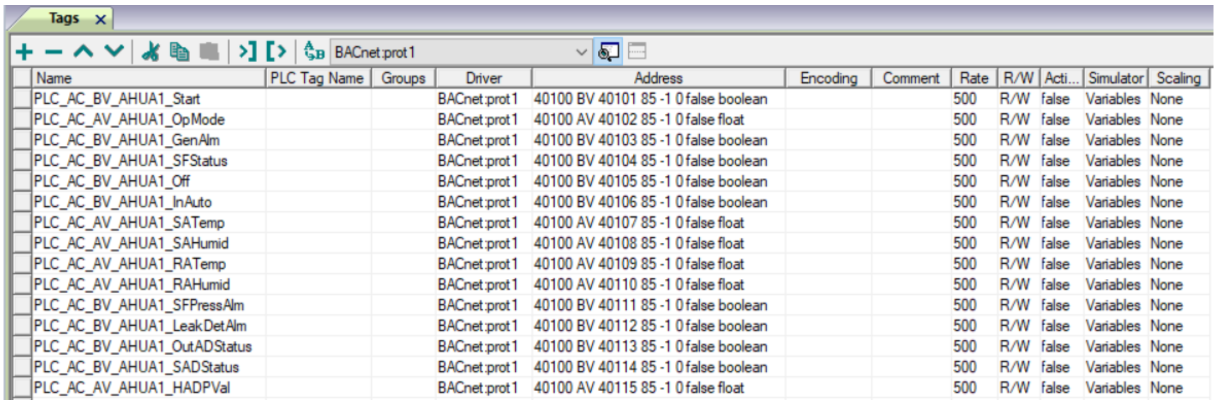
6.4.2 List of Tags

All the signals, which will be transferring between PLCs, need to be defined by tags. Each tag has the following properties:

- **Name:** A unique name for each signal.
- **Driver:** Protocol that is used for transferring the signal.
- **Address:** A set of properties that define BACnet object as follows:
  a) **Object type:** The type of object according to the BACnet specification. In this project, ana-log and binary values are used.
  b) **Device ID:** An identifier of the BACnet device that contains the current object. The PLCs AC and MCC have IDs 40100 and 40300 respectively.
  c) **Data type:** The type of data of the current object. In this project, float is used for analog values and boolean – for binary values.
  d) **Object instance:** A unique ID of this BACnet object. Objects that belong to the AC have IDs from 40101 to 40255, objects on the side of the MCC have IDs from 40201 to 40455.
  e) **Object property:** Defines which property will be read by the panel. In this project, value 85 is used for all signal that allows reading of property "Present_Value".
  f) **Write priority:** Defines priority for the cases when there is a need to synchronize the ac-tions of multiple clients potentially writing to the same property. Values are in a range from 1 to 16 where 1 is the most important and 16 is least important. In this project, write priority is not used and so it is set to 0.
- **Rate:** The frequency of updating the value.
- **Read/Write:** Read or write access to the signal value.

The most important point is to specify a unique tag for each signal, which will coincide with the object ID created in the application for the corresponding PLC. An object instance is a direct connection from the panel to the PLC.

The beginning of the list of tags is provided in figure 22.



Tags

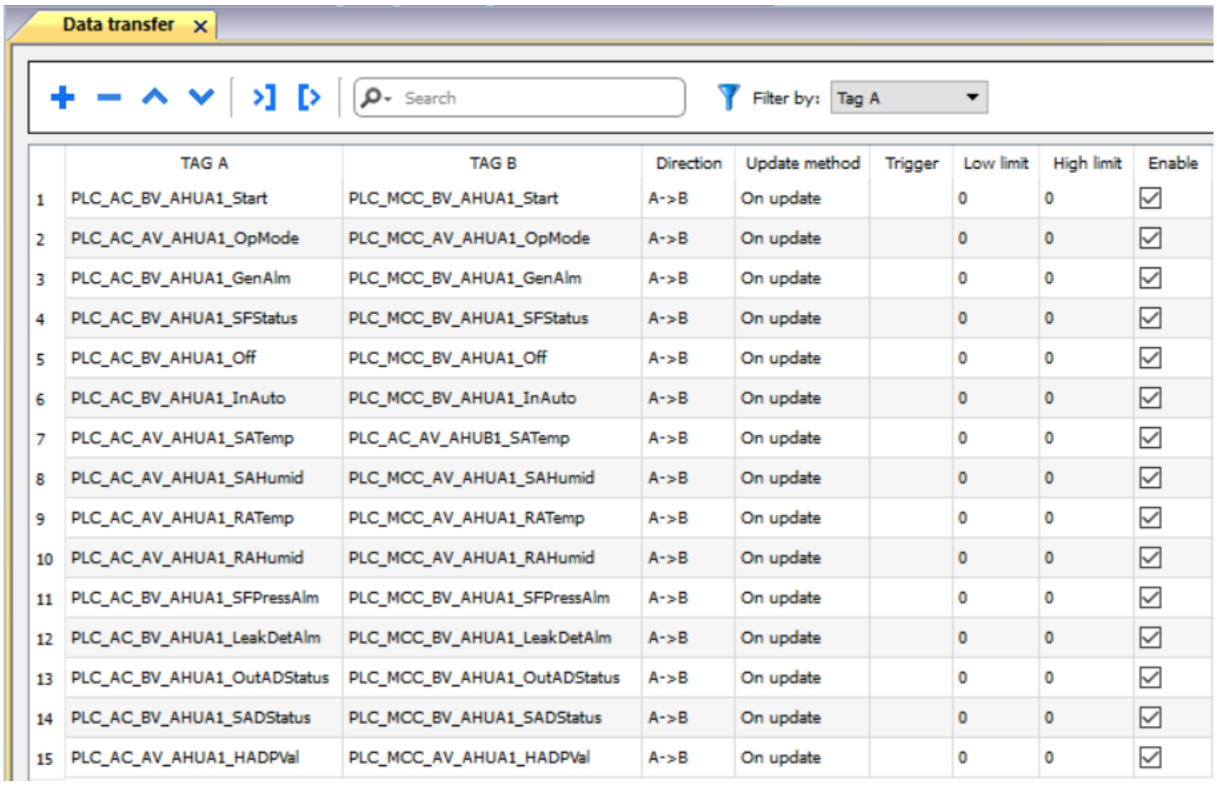
BACnet:prot1

Name	PLC Tag Name	Groups	Driver	Address	Encoding	Comment	Rate	R/W	Acti...	Simulator	Scaling
PLC_AC_BV_AHUA1_Start			BACnet:prot1	40100 BV 40101 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_OpMode			BACnet:prot1	40100 AV 40102 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_GenAlm			BACnet:prot1	40100 BV 40103 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SFStatus			BACnet:prot1	40100 BV 40104 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_Off			BACnet:prot1	40100 BV 40105 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_InAuto			BACnet:prot1	40100 BV 40106 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_SATemp			BACnet:prot1	40100 AV 40107 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_SAHumid			BACnet:prot1	40100 AV 40108 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_RATemp			BACnet:prot1	40100 AV 40109 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_RAHumid			BACnet:prot1	40100 AV 40110 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SFPressAlm			BACnet:prot1	40100 BV 40111 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_LeakDetAlm			BACnet:prot1	40100 BV 40112 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_OutADStatus			BACnet:prot1	40100 BV 40113 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SADStatus			BACnet:prot1	40100 BV 40114 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_HADPVal			BACnet:prot1	40100 AV 40115 85 -1 0 false float			500	R/W	false	Variables	None

Figure 22 – Part of the list of tags

6.4.3 Data Transfer

In order to send the values from the AC to the MCC, corresponding tags and the direction of communication should be defined in Data transfer. Here all created tags are listed. Figure 23 shows the beginning of this list.



Data transfer

Search

Filter by: Tag A

	TAG A	TAG B	Direction	Update method	Trigger	Low limit	High limit	Enable
1	PLC_AC_BV_AHUA1_Start	PLC_MCC_BV_AHUA1_Start	A->B	On update		0	0	✓
2	PLC_AC_AV_AHUA1_OpMode	PLC_MCC_AV_AHUA1_OpMode	A->B	On update		0	0	✓
3	PLC_AC_BV_AHUA1_GenAlm	PLC_MCC_BV_AHUA1_GenAlm	A->B	On update		0	0	✓
4	PLC_AC_BV_AHUA1_SFStatus	PLC_MCC_BV_AHUA1_SFStatus	A->B	On update		0	0	✓
5	PLC_AC_BV_AHUA1_Off	PLC_MCC_BV_AHUA1_Off	A->B	On update		0	0	✓
6	PLC_AC_BV_AHUA1_InAuto	PLC_MCC_BV_AHUA1_InAuto	A->B	On update		0	0	✓
7	PLC_AC_AV_AHUA1_SATemp	PLC_AC_AV_AHUB1_SATemp	A->B	On update		0	0	✓
8	PLC_AC_AV_AHUA1_SAHumid	PLC_MCC_AV_AHUA1_SAHumid	A->B	On update		0	0	✓
9	PLC_AC_AV_AHUA1_RATemp	PLC_MCC_AV_AHUA1_RATemp	A->B	On update		0	0	✓
10	PLC_AC_AV_AHUA1_RAHumid	PLC_MCC_AV_AHUA1_RAHumid	A->B	On update		0	0	✓
11	PLC_AC_BV_AHUA1_SFPressAlm	PLC_MCC_BV_AHUA1_SFPressAlm	A->B	On update		0	0	✓
12	PLC_AC_BV_AHUA1_LeakDetAlm	PLC_MCC_BV_AHUA1_LeakDetAlm	A->B	On update		0	0	✓
13	PLC_AC_BV_AHUA1_OutADStatus	PLC_MCC_BV_AHUA1_OutADStatus	A->B	On update		0	0	✓
14	PLC_AC_BV_AHUA1_SADStatus	PLC_MCC_BV_AHUA1_SADStatus	A->B	On update		0	0	✓
15	PLC_AC_AV_AHUA1_HADPVal	PLC_MCC_AV_AHUA1_HADPVal	A->B	On update		0	0	✓

Figure 23 – Part of data transfer configuration

“Tag A” specifies the tag of the AC signal from which the value is read, “Tag B” specifies the tag of the MCC signal where this particular value should be written to. Direction indicates whether the data are transferred from A to B or from B to A. Update method “On update” is selected, which means that the value will be sent as soon as it is updated.

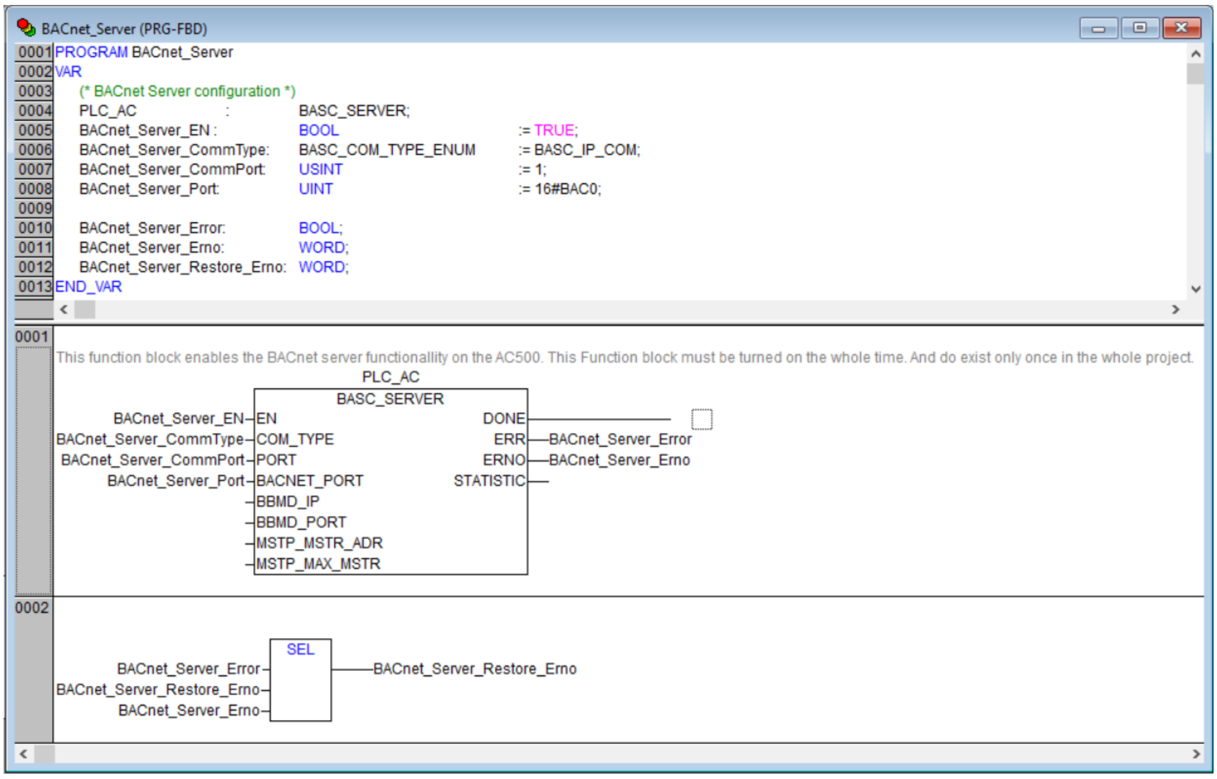
After the configuration is completed, the Panel Builder project is downloaded to the panel.

6.5 Application for Area Controller

The application for the Area Controller is prepared in CoDeSys, which is integrated into Automation Builder. The BACnet server and BACnet device are configured for the AC, all BACnet objects related to the AC are defined, and also the signals coming from the field devices are simulated in the application. It consists of four POUs: BACnet_Server, BACnet_Device, BACnet_Objects, and PLC_PRG.

6.5.1 BACnet Server

The program BACnet_Server is written using function block diagram (FBD) language. All variables used in this program unit are defined in the top area of variable declarations. The bottom part contains the executed program. In this program unit, communication via BACnet protocol is enabled, BACnet network type and communication port are defined. Used variables and the executed program are provided in figure 24.


BACnet_Server (PRG-FBD)

```
PROGRAM BACnet_Server
VAR
    (* BACnet Server configuration *)
    PLC_AC                      :     BASC_SERVER;
    BACnet_Server_EN :                BOOL                          := TRUE;
    BACnet_Server_CommType:           BASC_COM_TYPE_ENUM            := BASC_IP_COM;
    BACnet_Server_CommPort:           USINT                         := 1;
    BACnet_Server_Port:               UINT                          := 16#BAC0;

    BACnet_Server_Error:              BOOL;
    BACnet_Server_Erno:               WORD;
    BACnet_Server_Restore_Erno:       WORD;
END_VAR
```

0001

This function block enables the BACnet server functionallity on the AC500. This Function block must be turned on the whole time. And do exist only once in the whole project.

```
                                  PLC_AC
                               BASC_SERVER
        BACnet_Server_EN-EN                     DONE-
  BACnet_Server_CommType-COM_TYPE                ERR-BACnet_Server_Error
  BACnet_Server_CommPort-PORT                   ERNO-BACnet_Server_Erno
      BACnet_Server_Port-BACNET_PORT       STATISTIC-
                        -BBMD_IP
                        -BBMD_PORT
                        -MSTP_MSTR_ADR
                        -MSTP_MAX_MSTR
```

0002

```
                              SEL
         BACnet_Server_Error-      -BACnet_Server_Restore_Erno
  BACnet_Server_Restore_Erno-
          BACnet_Server_Erno-
```

Figure 24 – Configuration of BACnet server for the AC

Network 1 contains the function block BASC_SERVER that enables the BACnet server functionality on the AC500. This function block must be turned on the whole time and exist only once in the whole application. Inputs for BBMD and MS/TP are not used and outputs are only informative for this project. Network 2 contains a function block for binary selection that returns an error code if an error occurs.

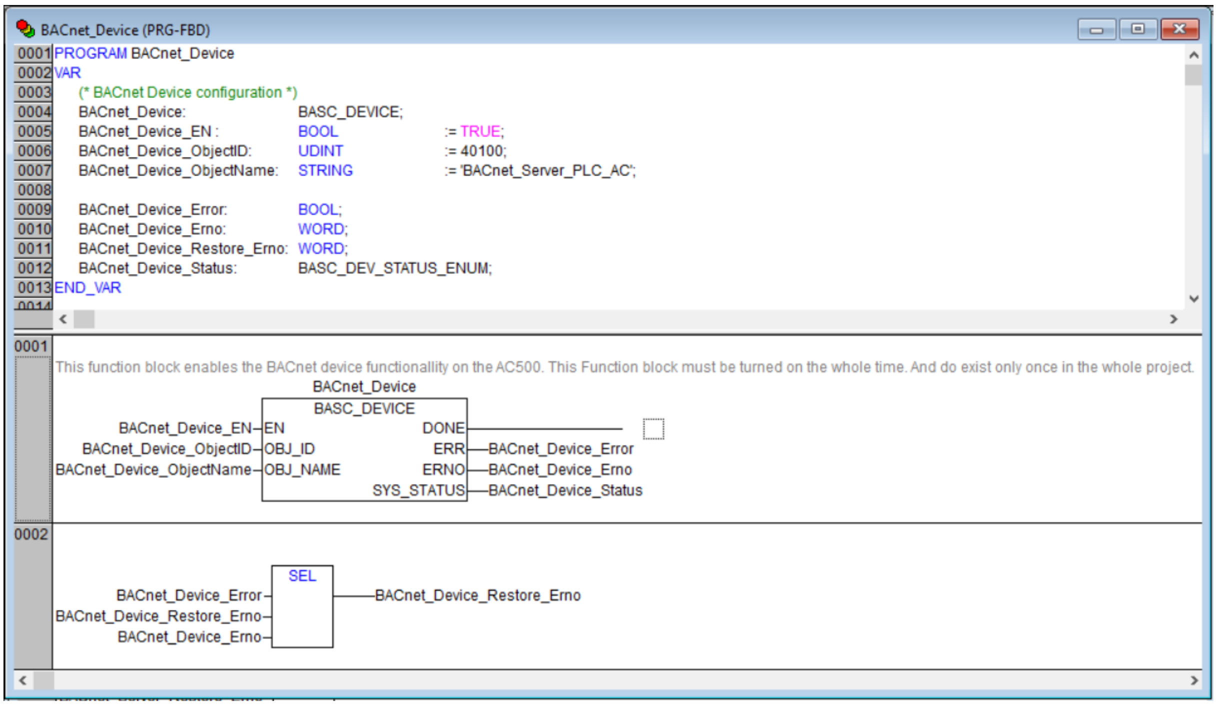
Description of inputs and outputs of BASC_SERVER function block is provided in table 8.

Table 8 – Description of inputs and outputs of BASC_SERVER function block

Name	Type	Description
INPUTS		
EN	BOOL	Enables function block by TRUE level
COM_TYPE	BASC_COM_TYPE_ENUM	Defines BACnet protocol type – IP or MSTP
PORT	BYTE	Index of Ethernet interface or serial port depending on COMM_TYPE
BACNET_PORT	WORD	Defines BACnet port in case of IP
BBMD_IP	STRING	The IP address of BBMD, used for register-foreign-device
BBMD_PORT	WORD	BACnet port of BBMD, used for register-foreign-device
MSTP_MSTR_ADR	BYTE	MS/TP master address
MSTP_MAX_MSTR	BYTE	Maximum master address for protocol MS/TP
OUTPUTS		
DONE	BOOL	Execution finished when the value is TRUE
ERR	BOOL	An error occurred during execution if the value is TRUE
ERNO	WORD	Error code according to BACnet documentation
STATISTIC	BASC_STATISTIC_COUNTER_TYPE	BACnet statistics – number of received requests, number of replies with success, and number of replies with failure

6.5.2 BACnet Device

The program BACnet_Device is written using function block diagram (FBD) language. All variables used in this program unit are defined in the top area of variable declarations. The bottom part contains the executed program. In this program unit, the operation of the AC that represents the BACnet device is enabled, and its required properties – "Object_Identifier" and "Object_Name" – are defined. The object ID is set to 40100 – the same value was assigned to the AC inside the panel project. Used variables and the executed program are provided in figure 25.


BACnet_Device (PRG-FBD)

```
PROGRAM BACnet_Device
VAR
    (* BACnet Device configuration *)
    BACnet_Device:                  BASC_DEVICE;
    BACnet_Device_EN :              BOOL            := TRUE;
    BACnet_Device_ObjectID:         UDINT           := 40100;
    BACnet_Device_ObjectName:       STRING          := 'BACnet_Server_PLC_AC';

    BACnet_Device_Error:            BOOL;
    BACnet_Device_Erno:             WORD;
    BACnet_Device_Restore_Erno:     WORD;
    BACnet_Device_Status:           BASC_DEV_STATUS_ENUM;
END_VAR
```

0001

This function block enables the BACnet device functionallity on the AC500. This Function block must be turned on the whole time. And do exist only once in the whole project.

0002

Figure 25 – Configuration of BACnet device for the AC

Network 1 contains the function block BASC_DEVICE that enables the BACnet device functionality on the AC500. This function block must be turned on the whole time and exist only once in the whole application. Outputs of this block are only informative for this project. Network 2 contains a function block for binary selection that returns an error code if an error occurs.

Description of inputs and outputs of BASC_DEVICE function block is provided in table 9.

Table 9 – Description of inputs and outputs of BASC_DEVICE function block

Name	Type	Description
		INPUTS
EN	BOOL	Enables function block by TRUE level
OBJ_ID	DWORD	Numeric code that is used to identify the object
OBJ_NAME	STRING	Represents a name for the object that is unique network-wide
		OUTPUTS
DONE	BOOL	Execution finished when the value is TRUE
ERR	BOOL	An error occurred during execution if the value is TRUE
ERNO	WORD	Error code according to BACnet documentation
SYS_STATUS	BASC_DEV_STATUS_ENUM	Reflects the current physical and logical status of the BACnet Device

6.5.3 BACnet Objects

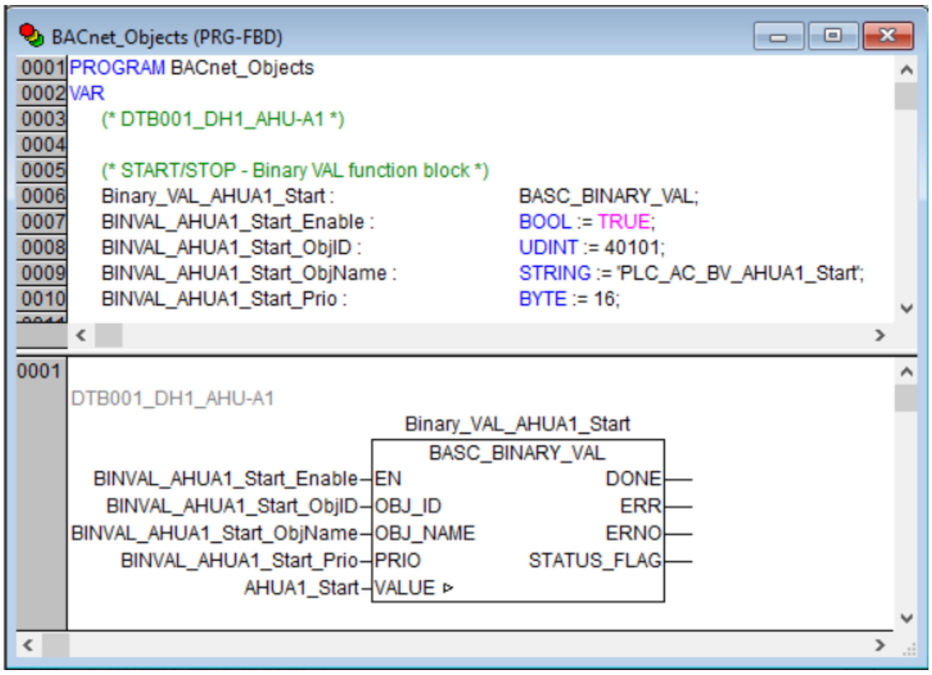
The program BACnet_Objects is written using function block diagram (FBD) language. All variables used in this program unit are defined in the top area of variable declarations. The bottom part contains the executed program.

The PS565 BACnet-ASC library contains the following function blocks that represent BACnet objects:

- BASC_ANALOG_IN – used to connect physical analog input to BACnet object of type analog input
- BASC_ANALOG_OUT – used to connect the physical analog output to BACnet object of type analog output
- BASC_ANALOG_VAL – used to connect IEC variable of type real to BACnet object of type analog value
- BASC_BINARY_IN – used to connect physical binary input to BACnet object of type binary input
- BASC_BINARY_OUT – used to connect the physical binary output to BACnet object of type binary output
- BASC_BINARY_VAL – used to connect IEC variable of type boolean to BACnet object of type binary value

Since the signals are not coming from the real inputs of the PLC but simulated inside of the application, function blocks BASC_BINARY_VAL and BASC_ANALOG_VAL are used for binary and analog values respectively.

The function block BASC_BINARY_VAL represents the BACnet object functionality on the AC500. It keeps the binary value that can be both input and output values. The configuration of inputs and outputs is shown in figure 26.



```
PROGRAM BACnet_Objects
VAR
    (* DTB001_DH1_AHU-A1 *)

    (* START/STOP - Binary VAL function block *)
    Binary_VAL_AHUA1_Start :            BASC_BINARY_VAL;
    BINVAL_AHUA1_Start_Enable :         BOOL := TRUE;
    BINVAL_AHUA1_Start_ObjID :          UDINT := 40101;
    BINVAL_AHUA1_Start_ObjName :        STRING := 'PLC_AC_BV_AHUA1_Start';
    BINVAL_AHUA1_Start_Prio :           BYTE := 16;
```

```
DTB001_DH1_AHU-A1
                                   Binary_VAL_AHUA1_Start
                                  +----------------------------+
                                  |      BASC_BINARY_VAL       |
     BINVAL_AHUA1_Start_Enable----|EN                      DONE|----
      BINVAL_AHUA1_Start_ObjID----|OBJ_ID                   ERR|----
    BINVAL_AHUA1_Start_ObjName----|OBJ_NAME                ERNO|----
        BINVAL_AHUA1_Start_Prio---|PRIO             STATUS_FLAG|----
                   AHUA1_Start----|VALUE ▹                     |
                                  +----------------------------+
```

Figure 26 – Configuration of BASC_BINARY_VAL function blocks

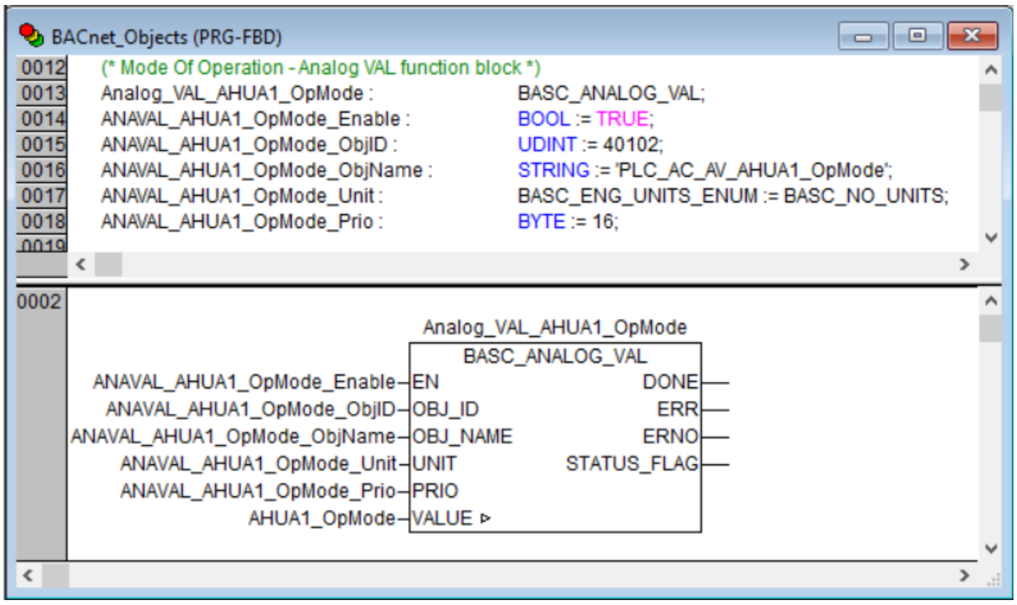
Network 1 contains the BACnet object that defines binary value indicating signal “START/STOP” from AHU-A1 device. In the variable declarations area, the operation of this block is enabled, its required properties – “Object_Identifier” and “Object_Name” – are defined, and the priority is given. The object ID is set to 40101 – the same value was assigned to this signal inside the panel project. The property “Present_Value” contains the value of the global variable “AHUA1_Start”, which is assigned in the main program.

Description of inputs and outputs of BASC_BINARY_VAL is provided in table 10.

Table 10 – Description of inputs and outputs of BASC_BINARY_VAL function block

Name	Type	Description
		INPUTS
EN	BOOL	Enables function block by TRUE level
OBJ_ID	DWORD	Numeric code that is used to identify the object
OBJ_NAME	STRING	A unique name for the object
PRIO	BYTE	Indicates the priority for the value
		INPUT/OUTPUT
VALUE	BOOL	Indicates the current value of the binary value
		OUTPUTS
DONE	BOOL	Execution finished when the value is TRUE
ERR	BOOL	An error occurred during execution if the value is TRUE
ERNO	WORD	Error code according to BACnet documentation
STATUS_FLAG	BYTE	Status flags according to BACnet documentation

The function block BASC_ANALOG_VAL represents the BACnet object functionality on the AC500. It keeps the analog value that can be both input and output values. The configuration of inputs and outputs is shown in figure 27.



```
(* Mode Of Operation - Analog VAL function block *)
Analog_VAL_AHUA1_OpMode :            BASC_ANALOG_VAL;
ANAVAL_AHUA1_OpMode_Enable :         BOOL := TRUE;
ANAVAL_AHUA1_OpMode_ObjID :          UDINT := 40102;
ANAVAL_AHUA1_OpMode_ObjName :        STRING := 'PLC_AC_AV_AHUA1_OpMode';
ANAVAL_AHUA1_OpMode_Unit :           BASC_ENG_UNITS_ENUM := BASC_NO_UNITS;
ANAVAL_AHUA1_OpMode_Prio :           BYTE := 16;
```

Figure 27 – Configuration of BASC_ANALOG_VAL function blocks

Network 2 contains the BACnet object that defines analog value indicating signal “Mode of operation” from AHU-A1 device. In the variable declarations area, the operation of this block is enabled, its required properties – “Object_Identifier”, “Object_Name”, and “Units” – are defined, and the priority is given. The object ID is set to 40102 – the same value was assigned to this signal inside the panel project. The property “Present_Value” contains the value of the global variable “AHUA1_OpMode”, which is assigned in the main program.

Description of inputs and outputs of the BASC_ANALOG_VAL function block is provided in table 11.

Table 11 – Description of inputs and outputs of BASC_ANALOG_VAL function block

Name	Type	Description
INPUTS		
EN	BOOL	Enables function block by TRUE level
OBJ_ID	DWORD	Numeric code that is used to identify the object
OBJ_NAME	STRING	A unique name for the object
UNIT	BASC_ENG_UNITS_ENUM	Indicates the engineering units of this object
PRIO	BYTE	Indicates the priority for the value
INPUT/OUTPUT		
VALUE	REAL	Indicates the current value, in engineering units, of the analog value
OUTPUTS		
DONE	BOOL	Execution finished when the value is TRUE
ERR	BOOL	An error occurred during execution if the value is TRUE
ERNO	WORD	Error code according to BACnet documentation
STATUS_FLAG	BYTE	Status flags according to BACnet documentation

All other signals from the field devices are prepared in the same way as BACnet objects of analog and binary values. The object IDs for the objects that belong to the AC range from 40101 to 40255.

6.5.4 Main Program PLC_PRG

The main program is written using structured text (ST) language. This program unit represents a simulation of the values received from the field devices. The variables that are used for keeping the received values are defined in the “Global Variables” area. Values from analog signals have type “REAL”, values from digital signals have type “BOOLEAN”. The simulated values are assigned to the global variables in the main program “PLC_PRG”.

6.6 Application for Master Controller

The application for the Master Controller is prepared in CoDeSys, which is integrated into Automation Builder. The BACnet server and BACnet device are configured for the MCC, all BACnet objects related to the MCC are defined, and also the structure of analog and digital signals according to the representation in the Engineering Workplace is prepared. It consists of five POUs: BACnet_Server, BACnet_Device, BACnet_Objects, PLC_PRG, and MCC function block.

6.6.1 BACnet_Server

The program BACnet_Server is written using function block diagram (FBD) language. It duplicates the program BACnet_Server from the AC application, except for the name given to the BASC_SERVER FB. The purpose of this program unit is to enable the BACnet server functionality on the MCC.

6.6.2 BACnet_Device

The program BACnet_Device is written using function block diagram (FBD) language. It is equivalent to the program BACnet_Device from the AC application. But in this instance, required properties – "Object_Identifier" and "Object_Name" – are different. The object ID is set to 40300 – the same value that was assigned to the MCC inside the panel project, and the object name is unique for the MCC. The purpose of this program unit is to enable the BACnet device functionality on the MCC.

6.6.3 BACnet_Objects

The program BACnet_Objects is written using function block diagram (FBD) language. It is equivalent to the program BACnet_Objects from the AC application. This program unit defines analog and binary values that are transferred from the AC as BACnet objects. But in this instance, required properties – "Object_Identifier" and "Object_Name" – are different. The object IDs for the objects that belong to the MCC range from 40301 to 40455. The property "Present_Value" of each BACnet object contains the value of the corresponding global variable, which is assigned in the main program.

6.6.4 MCC Data Type

MCC data type defines analog and digital input signals of field devices in the format required by the Engineering Workplace. This data type is defined in the "Global Variables" area, which is later written to the symbol file Application_MCC.sdb and sent via OPC service. Figure 28 shows the beginning of this structure.

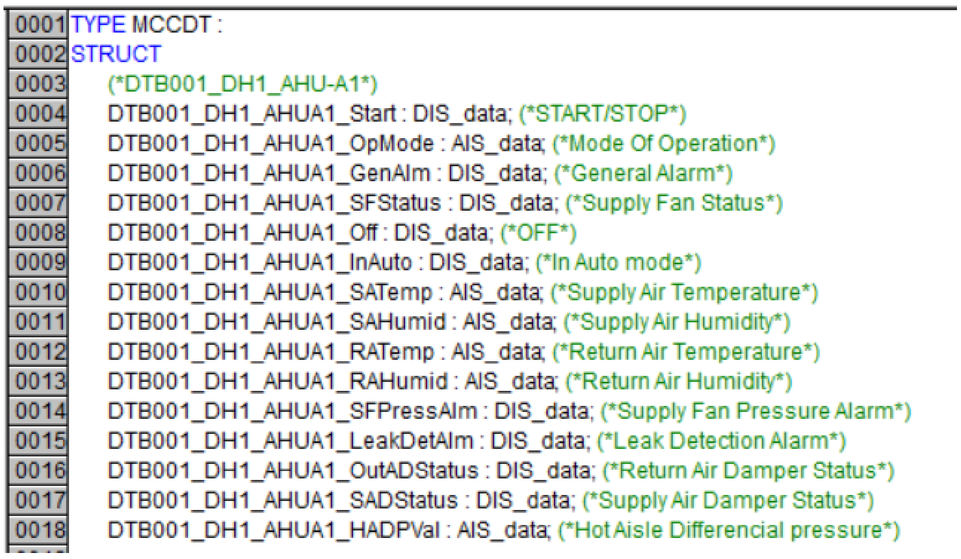

```
TYPE MCCDT :
STRUCT
    (*DTB001_DH1_AHU-A1*)
    DTB001_DH1_AHUA1_Start : DIS_data; (*START/STOP*)
    DTB001_DH1_AHUA1_OpMode : AIS_data; (*Mode Of Operation*)
    DTB001_DH1_AHUA1_GenAlm : DIS_data; (*General Alarm*)
    DTB001_DH1_AHUA1_SFStatus : DIS_data; (*Supply Fan Status*)
    DTB001_DH1_AHUA1_Off : DIS_data; (*OFF*)
    DTB001_DH1_AHUA1_InAuto : DIS_data; (*In Auto mode*)
    DTB001_DH1_AHUA1_SATemp : AIS_data; (*Supply Air Temperature*)
    DTB001_DH1_AHUA1_SAHumid : AIS_data; (*Supply Air Humidity*)
    DTB001_DH1_AHUA1_RATemp : AIS_data; (*Return Air Temperature*)
    DTB001_DH1_AHUA1_RAHumid : AIS_data; (*Return Air Humidity*)
    DTB001_DH1_AHUA1_SFPressAlm : DIS_data; (*Supply Fan Pressure Alarm*)
    DTB001_DH1_AHUA1_LeakDetAlm : DIS_data; (*Leak Detection Alarm*)
    DTB001_DH1_AHUA1_OutADStatus : DIS_data; (*Return Air Damper Status*)
    DTB001_DH1_AHUA1_SADStatus : DIS_data; (*Supply Air Damper Status*)
    DTB001_DH1_AHUA1_HADPVal : AIS_data; (*Hot Aisle Differencial pressure*)
```

Figure 28 – MCC data type

The structure lists all the signals coming from the field devices. Analog signals have custom type AIS_data provided in table 12, digital signals have custom type DIS_data provided in table 13.

Table 12 – Description of custom type AIS_data

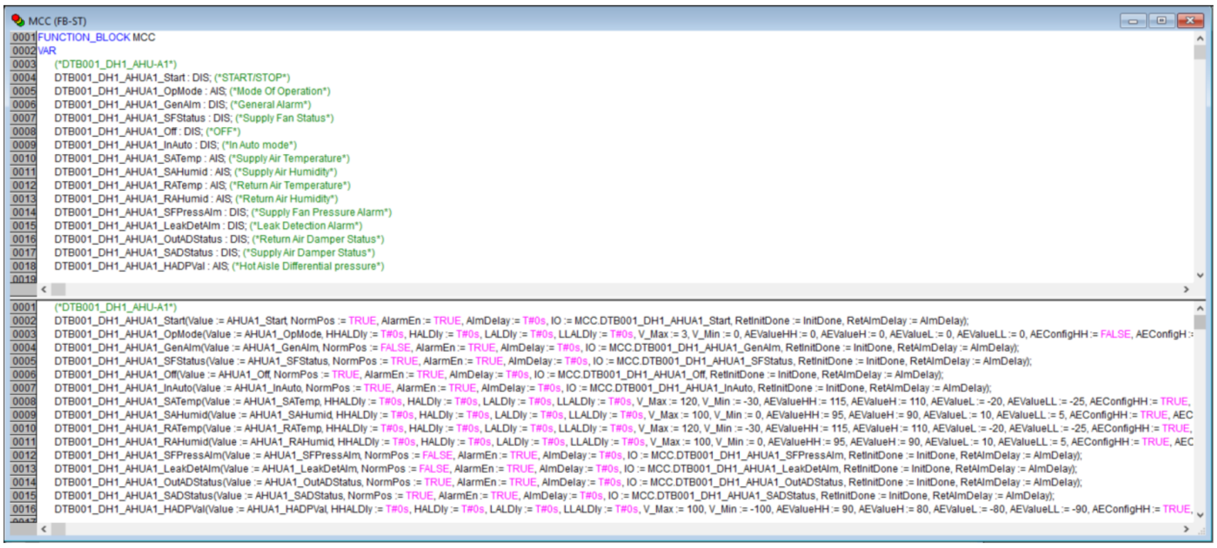
Name	Type	Description
INPUTS		
IN_Value	REAL	Process value
IN_InhAlm	BOOL	Inhibits all alarms
IN_AlmHHAck	BOOL	Acknowledge HIGH-HIGH alarm on the rising edge
IN_AlmHAck	BOOL	Acknowledge HIGH alarm on the rising edge
IN_AlmLAck	BOOL	Acknowledge LOW alarm on the rising edge
IN_AlmLLAck	BOOL	Acknowledge LOW-LOW alarm on the rising edge
IN_AckAlm	BOOL	Acknowledge alarm on the rising edge
IN_AEValueHH	REAL	Input value of HIGH-HIGH alarm
IN_AEValueH	REAL	Input value of HIGH alarm
IN_AEValueL	REAL	Input value of LOW alarm
IN_AEValueLL	REAL	Input value of LOW-LOW alarm
OUTPUTS		
OUT_Value	REAL	Process value
OUT_AlmActHH	BOOL	HIGH-HIGH alarm is active
OUT_AlmActH	BOOL	HIGH alarm is active
OUT_AlmActL	BOOL	LOW alarm is active
OUT_AlmActLL	BOOL	LOW-LOW alarm is active
OUT_AlmHHNorm	BOOL	HIGH-HIGH alarm is normal
OUT_AlmHNorm	BOOL	HIGH alarm is normal
OUT_AlmLNorm	BOOL	LOW alarm is normal
OUT_AlmLLNorm	BOOL	LOW-LOW alarm is normal
OUT_AlmHHUnack	BOOL	HIGH-HIGH alarm is not acknowledged
OUT_AlmHUnack	BOOL	HIGH alarm is not acknowledged
OUT_AlmLUnack	BOOL	LOW alarm is not acknowledged
OUT_AlmLLUnack	BOOL	LOW-LOW alarm is not acknowledged
OUT_AlmActERR	BOOL	Signal error state
OUT_AlmAct	BOOL	Any alarm is active
OUT_AlmSigErr	BOOL	Alarm Status or Signal Error, e.g. electrical signal less than 4 mA
OUT_Forced	BOOL	Tells if the input is forced or not
HW Value		
HW_Value	REAL	Value in the application

Table 13 – Description of custom type DIS_data

Name	Type	Description
		INPUTS
IN_InhAlm	BOOL	Inhibits alarm treatment
IN_AlmDelay	TIME	Alarm delay
		OUTPUTS
OUT_Value	BOOL	Output value
OUT_AlmActDIFF	BOOL	DIFF alarm is active
OUT_AlmNorm	BOOL	An alarm is not active and acknowledged
OUT_AlmUnAck	BOOL	Pulse: Acknowledge alarm on the rising edge
OUT_AlmAct	BOOL	Any alarm is active
OUT_Forced	BOOL	Tells if the input is forced or not

6.6.5 MCC Function Block

MCC function block is written using structured text (ST) language. All variables used in this function block are defined in the top area of variable declarations. The bottom part contains the definition of these variables according to their data type. MCC FB includes the definition of analog and digital input signals from field devices configured according to the Engineering Workplace. Figures 29, 30, 31 show the configuration for one AHU unit, one electric door, and one THS-CA sensor.

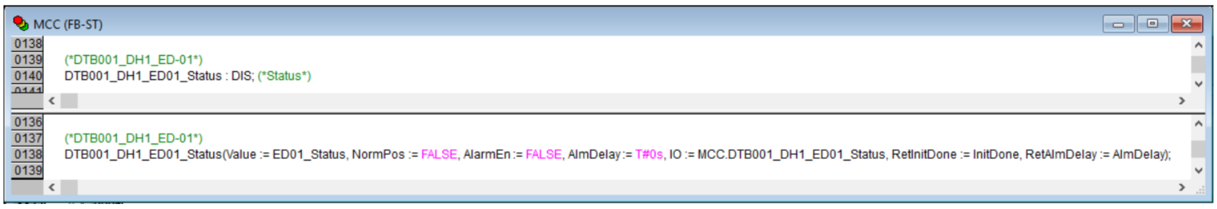


MCC (FB-ST)

```
FUNCTION_BLOCK MCC
VAR
    (*DTB001_DH1_AHU-A1*)
    DTB001_DH1_AHUA1_Start : DIS; (*START/STOP*)
    DTB001_DH1_AHUA1_OpMode : AIS; (*Mode Of Operation*)
    DTB001_DH1_AHUA1_GenAlm : DIS; (*General Alarm*)
    DTB001_DH1_AHUA1_SFStatus : DIS; (*Supply Fan Status*)
    DTB001_DH1_AHUA1_Off : DIS; (*OFF*)
    DTB001_DH1_AHUA1_InAuto : DIS; (*In Auto mode*)
    DTB001_DH1_AHUA1_SATemp : AIS; (*Supply Air Temperature*)
    DTB001_DH1_AHUA1_SAHumid : AIS; (*Supply Air Humidity*)
    DTB001_DH1_AHUA1_RATemp : AIS; (*Return Air Temperature*)
    DTB001_DH1_AHUA1_RAHumid : AIS; (*Return Air Humidity*)
    DTB001_DH1_AHUA1_SFPressAlm : DIS; (*Supply Fan Pressure Alarm*)
    DTB001_DH1_AHUA1_LeakDetAlm : DIS; (*Leak Detection Alarm*)
    DTB001_DH1_AHUA1_OutADStatus : DIS; (*Return Air Damper Status*)
    DTB001_DH1_AHUA1_SADStatus : DIS; (*Supply Air Damper Status*)
    DTB001_DH1_AHUA1_HADPVal : AIS; (*Hot Aisle Differential pressure*)
```

```
    (*DTB001_DH1_AHU-A1*)
    DTB001_DH1_AHUA1_Start(Value := AHUA1_Start, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_Start, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_OpMode(Value := AHUA1_OpMode, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 3, V_Min := 0, AEValueHH := 0, AEValueH := 0, AEValueL := 0, AEValueLL := 0, AEConfigHH := FALSE, AEConfigH :=
    DTB001_DH1_AHUA1_GenAlm(Value := AHUA1_GenAlm, NormPos := FALSE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_GenAlm, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_SFStatus(Value := AHUA1_SFStatus, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_SFStatus, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_Off(Value := AHUA1_Off, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_Off, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_InAuto(Value := AHUA1_InAuto, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_InAuto, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_SATemp(Value := AHUA1_SATemp, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 120, V_Min := -30, AEValueHH := 115, AEValueH := 110, AEValueL := -20, AEValueLL := -25, AEConfigHH := TRUE,
    DTB001_DH1_AHUA1_SAHumid(Value := AHUA1_SAHumid, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 100, V_Min := 0, AEValueHH := 95, AEValueH := 90, AEValueL := 10, AEValueLL := 5, AEConfigHH := TRUE, AEC
    DTB001_DH1_AHUA1_RATemp(Value := AHUA1_RATemp, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 120, V_Min := -30, AEValueHH := 115, AEValueH := 110, AEValueL := -20, AEValueLL := -25, AEConfigHH := TRUE,
    DTB001_DH1_AHUA1_RAHumid(Value := AHUA1_RAHumid, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 100, V_Min := 0, AEValueHH := 95, AEValueH := 90, AEValueL := 10, AEValueLL := 5, AEConfigHH := TRUE, AEC
    DTB001_DH1_AHUA1_SFPressAlm(Value := AHUA1_SFPressAlm, NormPos := FALSE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_SFPressAlm, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_LeakDetAlm(Value := AHUA1_LeakDetAlm, NormPos := FALSE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_LeakDetAlm, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_OutADStatus(Value := AHUA1_OutADStatus, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_OutADStatus, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_SADStatus(Value := AHUA1_SADStatus, NormPos := TRUE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_AHUA1_SADStatus, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
    DTB001_DH1_AHUA1_HADPVal(Value := AHUA1_HADPVal, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 100, V_Min := -100, AEValueHH := 90, AEValueH := 80, AEValueL := -80, AEValueLL := -90, AEConfigHH := TRUE,
```

Figure 29 – Configuration of analog and digital signals of one AHU



MCC (FB-ST)

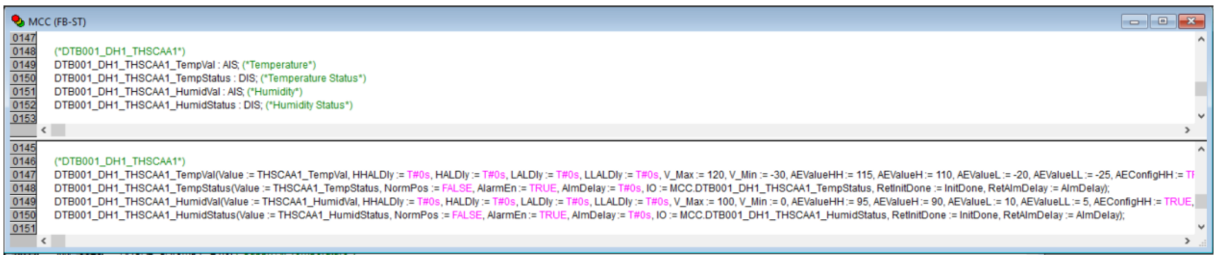
```

    (*DTB001_DH1_ED-01*)
    DTB001_DH1_ED01_Status : DIS; (*Status*)
```

```

    (*DTB001_DH1_ED-01*)
    DTB001_DH1_ED01_Status(Value := ED01_Status, NormPos := FALSE, AlarmEn := FALSE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_ED01_Status, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
```

Figure 30 – Configuration of the status of one electric door



```
(*DTB001_DH1_THSCAA1*)
DTB001_DH1_THSCAA1_TempVal : AIS; (*Temperature*)
DTB001_DH1_THSCAA1_TempStatus : DIS; (*Temperature Status*)
DTB001_DH1_THSCAA1_HumidVal : AIS; (*Humidity*)
DTB001_DH1_THSCAA1_HumidStatus : DIS; (*Humidity Status*)
```

```
(*DTB001_DH1_THSCAA1*)
DTB001_DH1_THSCAA1_TempVal(Value := THSCAA1_TempVal, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 120, V_Min := -30, AEValueHH := 115, AEValueH := 110, AEValueL := -20, AEValueLL := -25, AEConfigHH := TR
DTB001_DH1_THSCAA1_TempStatus(Value := THSCAA1_TempStatus, NormPos := FALSE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_THSCAA1_TempStatus, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
DTB001_DH1_THSCAA1_HumidVal(Value := THSCAA1_HumidVal, HHALDly := T#0s, HALDly := T#0s, LALDly := T#0s, LLALDly := T#0s, V_Max := 100, V_Min := 0, AEValueHH := 95, AEValueH := 90, AEValueL := 10, AEValueLL := 5, AEConfigHH := TRUE,
DTB001_DH1_THSCAA1_HumidStatus(Value := THSCAA1_HumidStatus, NormPos := FALSE, AlarmEn := TRUE, AlmDelay := T#0s, IO := MCC.DTB001_DH1_THSCAA1_HumidStatus, RetInitDone := InitDone, RetAlmDelay := AlmDelay);
```

Figure 31 – Configuration of analog and digital signals of one THS-CA

Analog signals are represented by AIS FB provided in table 14, digital signals are represented by DIS FB provided in table 15.

Table 14 – Description of function block AIS for analog input signal

Name	Type	Description
	INPUTS	
Value	REAL	Value coming from the application
IsCalc	BOOL	IN signal is calculated
Neg_Range	BOOL	Range from -10 to +10
Hysteresis	REAL	IN signal hysteresis in % of the range
HHALDly	TIME	IN signal filter time in seconds for HIGH-HIGH alarm
HALDly	TIME	IN signal filter time in seconds for HIGH alarm
LALDly	TIME	IN signal filter time in seconds for LOW alarm
LLALDly	TIME	IN signal filter time in seconds for LOW-LOW alarm
V_Max	REAL	IN Max value – used only if IsCalc = TRUE
V_Min	REAL	IN Min value – used only if IsCalc = TRUE
AEValueHH	REAL	Input value of HIGH-HIGH alarm
AEValueH	REAL	Input value of HIGH alarm
AEValueL	REAL	Input value of LOW alarm
AEValueLL	REAL	Input value of LOW-LOW alarm
AEConfigHH	BOOL	Configuration of the input value of HIGH-HIGH alarm
AEConfigH	BOOL	Configuration of the input value of HIGH alarm
AEConfigL	BOOL	Configuration of the input value of LOW alarm
AEConfigLL	BOOL	Configuration of the input value of LOW-LOW alarm
	INPUTS/OUTPUTS	
IO	AIS_data	Input/output logic interface
retInitDone	BOOL	The return value of initialization status
retAEValueHH	REAL	The severity of input value of HIGH-HIGH alarm from 1 to 4
retAEValueH	REAL	The severity of input value of HIGH alarm from 1 to 4
retAEValueL	REAL	The severity of input value of LOW alarm from 1 to 4
retAEValueLL	REAL	The severity of input value of LOW-LOW alarm from 1 to 4

Table 15 – Description of function block DIS for digital input signal

Name	Type	Description
		INPUTS
Value	BOOL	The value coming from the application
NormPos	BOOL	Signal of normal position used for alarm and event treatment
AlarmEn	BOOL	Enable alarm treatment
AlmDelay	TIME	Alarm delay
		INPUTS/OUTPUTS
IO	DIS_data	Input/output logic interface
ReInitDone	BOOL	The return value of initialization status
RetAlmDelay	TIME	The return value of alarm delay

The main program “PLC_PRG” starts the execution of the MCC function block.

7 Implementation of the Operator Workplace

7.1 Configuration of AC500 OPC Server

Communication of AC500 to 800xA is done through OPC DA. Therefore, it is required to install the AC500 OPC server and to configure it accordingly.

Basic configuration workflow for AC500 OPC Server:

The definition of the items (symbols) to be exchanged over OPC DA is stored in the symbol file "*.sdb". The symbol file is generated by the AC500 engineering tool Automation Builder which is running on the AC500 engineering node. When downloading the application to the AC500 PLC, the "*.sdb" file is stored on the used disk of the PLC. Finally, the OPC Server running on the 800xA engineering node uploads the "*.sdb" file in order to configure the tags accordingly (figure 32). [14]

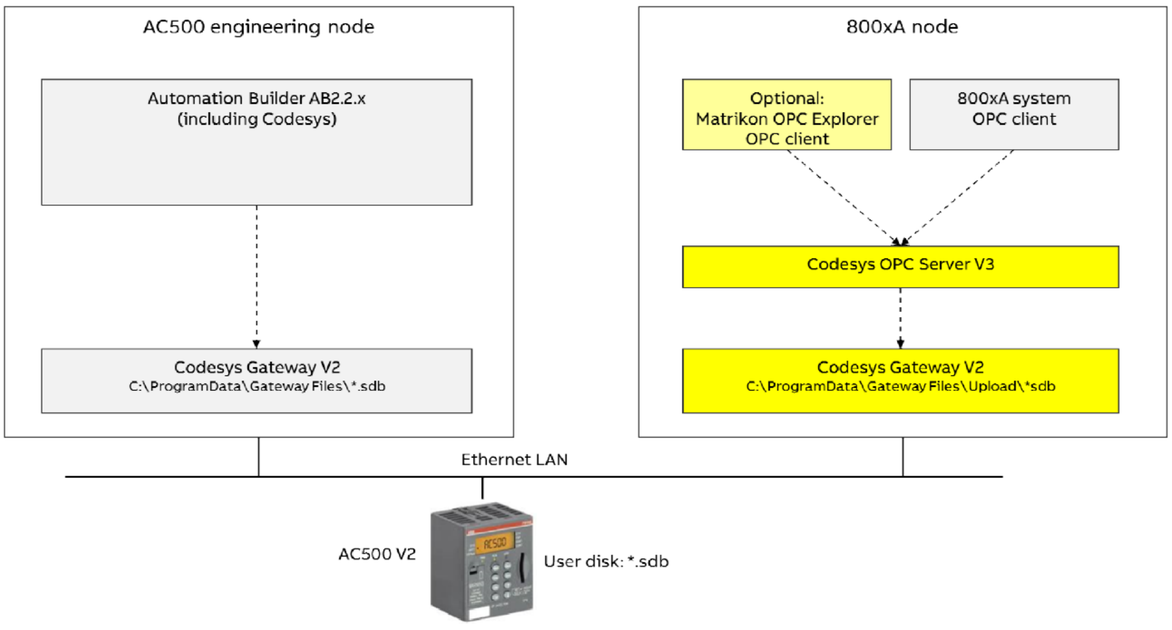

AC500 engineering node
Automation Builder AB2.2.x (including Codesys)
Codesys Gateway V2
C:\ProgramData\Gateway Files*.sdb
800xA node
Optional: Matrikon OPC Explorer OPC client
800xA system OPC client
Codesys OPC Server V3
Codesys Gateway V2
C:\ProgramData\Gateway Files\Upload\*sdb
Ethernet LAN
AC500 V2
User disk: *.sdb

Figure 32 – Connection from AC500 to 800xA engineering node

The symbol file is configured inside the CoDeSys application for the MCC and the configuration of global variables is sent to the PLC.

OPC Configurator tool is used for defining the OPC server. Description of configured settings for the OPC server:

Update rate (200 ms): This is the cycle time according to which the variable values are read from the controller. The data are written to the cache with which the client communicates according to a separately defined update rate.

Sync init (selected): If the option synchronous initialization is selected, the OPC server is available during start-up after the symbol configuration has been loaded.

Logging (selected): If the option Enable logging (default events) is selected, actions and errors detected on the OPC server are written to the OPCServer.log file in the installation directory.

Description of the configured settings for controller:

Interface (GATEWAY): The interface that is used for communication between the OPC server and the controller.

Timeout (10000 ms): If the OPC server has not received a response from the controller within this time, it is closed automatically.

Number of tries (3): Number of attempts to transfer a data block. As soon as the configured number of attempts has been unsuccessful, a message indicating communication interruption is generated.

Buffer size (4800 bytes): Size of the communication buffer on the target device.

Wait time (10 s): Time the OPC server waits until communication to the controller is available after an automatic start of the controller.

Reconnect time (15 s): Interval the OPC server waits to retry to connect to the controller via the gateway.

Active (selected): If this option is selected, the controller is taken into account by the OPC server.

Motorola byte order (selected): Motorola byte order for controllers based on the Motorola chipset.

No login service (selected): Option for target systems that require a login service.

Logging (selected): Options for tracing results in a .log file.

In the settings for connection to PLC MC, gateway TCP/IP is chosen, and the IP address of the PLC is specified.

After saving the configuration, the “OPCServer.ini” file is saved on the client side, which will allow connection to the server.

7.2 Application for the 800xA System

The application for the 800xA system is prepared in Engineering Workplace. The structure of the project in the 800xA development environment is divided according to functionality. There are 21 structures by default. In this application, only three of them are used. An overview of the project structure is provided below.

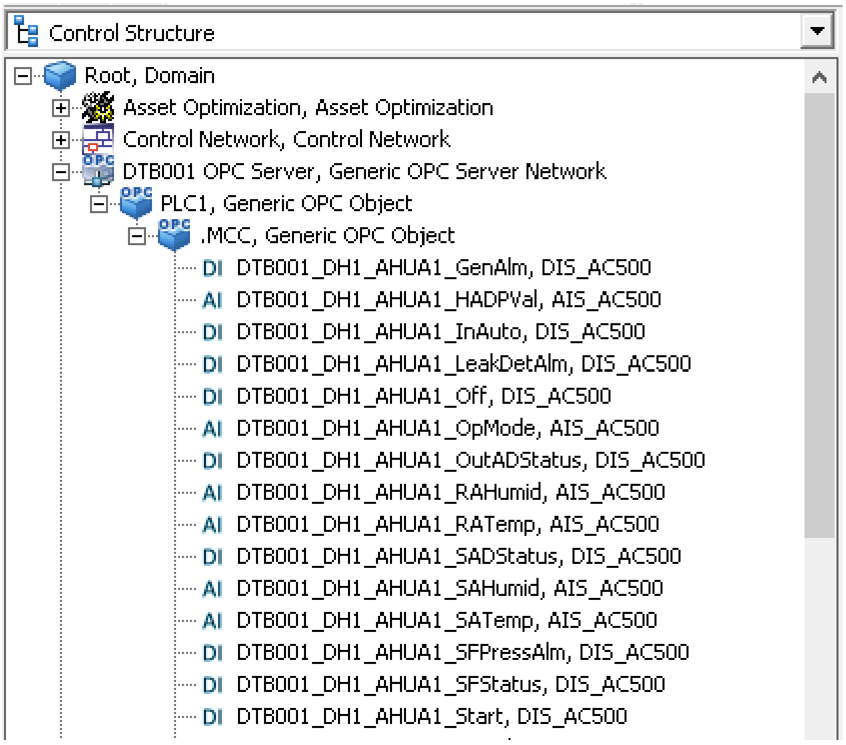
- Control structure: Access to the internal variables of the AC500 PLC.
- Object type structure: Implementation of templates for user interface elements, mapping of signals to the filed devices in general.
- Functional structure: Implementation of the user interface, mapping of signals to the field devices.

7.2.1 Control Structure

Any structure inside the Engineering Workplace consists of aspect objects. An aspect object is a computer representation of a real-world object. Each aspect object has several aspects, i.e. characteristics associated with an aspect object, such as drawings, maintenance records, trends, and faceplates.

The Control Structure is used to organize the process control environment in the system, i.e. to define where the different parts of the control application execute together with the controller hardware, and also for consistency check and monitoring the system status.

The Control Structure of the application is shown in figure 33. For this application, a generic OPC server network named "DTB001 OPC Server" is created to manage the connection to the master controller via OPC. It contains an aspect "Uploader" that allows the import of objects provided in the network. After the configuration is done, the OPC server named "PLC1" appears under the network with all containing objects. In this case, the object "MCC", which was created in Automation Builder as a function block, is reflected in this structure. All function blocks representing analog and digital signals, that were assigned to the MCC FB and exported into a symbol file, also appear under the MCC object.



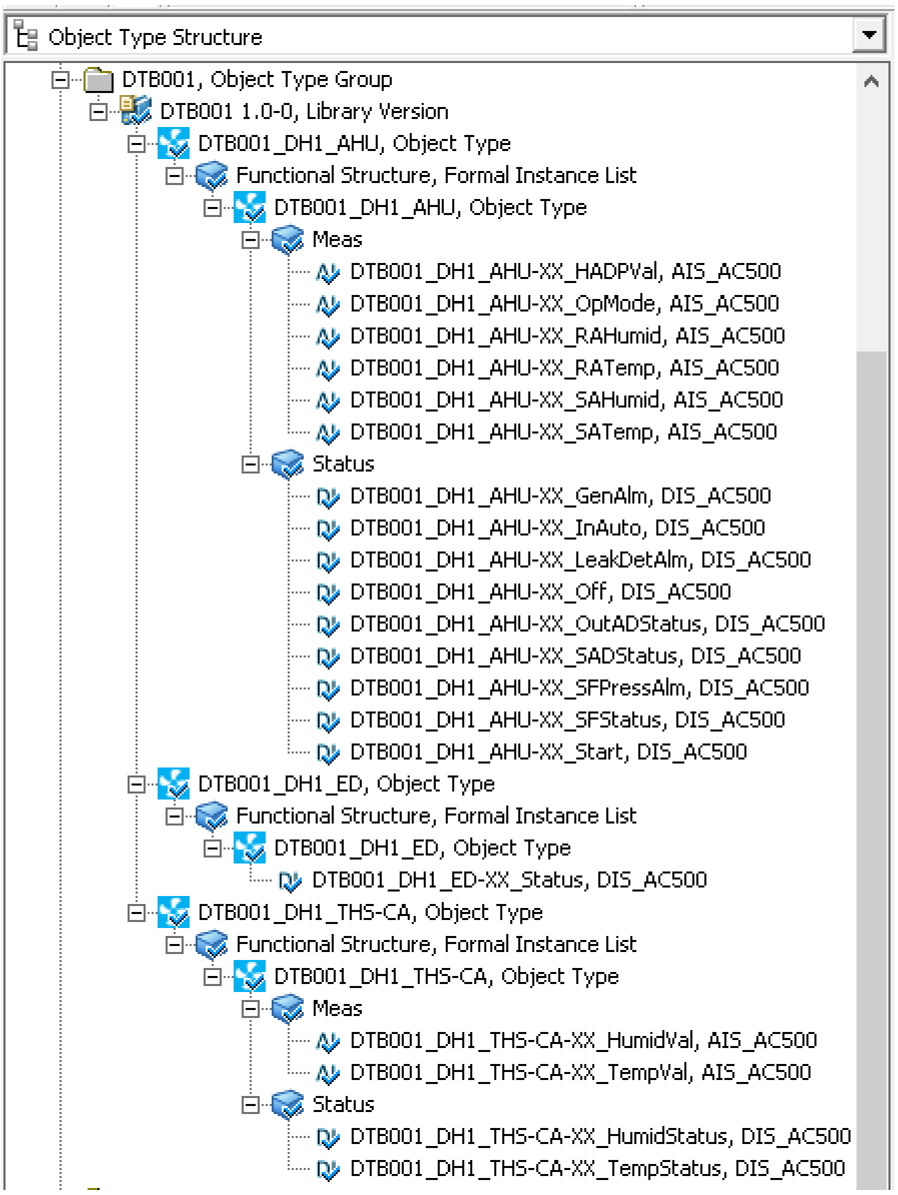
```
Control Structure
⊟ Root, Domain
  ⊞ Asset Optimization, Asset Optimization
  ⊞ Control Network, Control Network
  ⊟ DTB001 OPC Server, Generic OPC Server Network
    ⊟ PLC1, Generic OPC Object
      ⊟ .MCC, Generic OPC Object
          DI DTB001_DH1_AHUA1_GenAlm, DIS_AC500
          AI DTB001_DH1_AHUA1_HADPVal, AIS_AC500
          DI DTB001_DH1_AHUA1_InAuto, DIS_AC500
          DI DTB001_DH1_AHUA1_LeakDetAlm, DIS_AC500
          DI DTB001_DH1_AHUA1_Off, DIS_AC500
          AI DTB001_DH1_AHUA1_OpMode, AIS_AC500
          DI DTB001_DH1_AHUA1_OutADStatus, DIS_AC500
          AI DTB001_DH1_AHUA1_RAHumid, AIS_AC500
          AI DTB001_DH1_AHUA1_RATemp, AIS_AC500
          DI DTB001_DH1_AHUA1_SADStatus, DIS_AC500
          AI DTB001_DH1_AHUA1_SAHumid, AIS_AC500
          AI DTB001_DH1_AHUA1_SATemp, AIS_AC500
          DI DTB001_DH1_AHUA1_SFPressAlm, DIS_AC500
          DI DTB001_DH1_AHUA1_SFStatus, DIS_AC500
          DI DTB001_DH1_AHUA1_Start, DIS_AC500
```

Figure 33 – Control Structure

7.2.2 Object Type Structure

Almost all aspect objects are instances of object types defined in the Object Type Structure. This structure allows creating standardized solutions for recurring problems to efficiently re-use them later. When an instance of some object type is created, the predefined aspects for that object type are automatically instantiated and associated with the new instance.

The Object Type Structure of the application is shown in figure 34. For this application, an object type group named “DTB001” is created. It contains object types for field devices: “DTB001_DH1_AHU” for AHU units, “DTB001_DH1_ED” for electric doors, and “DTB001_DH1_THS-CA” for THS-CA sensors. These object types contain all associated analog and digital signals, according to the structure prepared for the MCC application in Automation Builder.



```
Object Type Structure
DTB001, Object Type Group
  DTB001 1.0-0, Library Version
    DTB001_DH1_AHU, Object Type
      Functional Structure, Formal Instance List
        DTB001_DH1_AHU, Object Type
          Meas
            DTB001_DH1_AHU-XX_HADPVal, AIS_AC500
            DTB001_DH1_AHU-XX_OpMode, AIS_AC500
            DTB001_DH1_AHU-XX_RAHumid, AIS_AC500
            DTB001_DH1_AHU-XX_RATemp, AIS_AC500
            DTB001_DH1_AHU-XX_SAHumid, AIS_AC500
            DTB001_DH1_AHU-XX_SATemp, AIS_AC500
          Status
            DTB001_DH1_AHU-XX_GenAlm, DIS_AC500
            DTB001_DH1_AHU-XX_InAuto, DIS_AC500
            DTB001_DH1_AHU-XX_LeakDetAlm, DIS_AC500
            DTB001_DH1_AHU-XX_Off, DIS_AC500
            DTB001_DH1_AHU-XX_OutADStatus, DIS_AC500
            DTB001_DH1_AHU-XX_SADStatus, DIS_AC500
            DTB001_DH1_AHU-XX_SFPressAlm, DIS_AC500
            DTB001_DH1_AHU-XX_SFStatus, DIS_AC500
            DTB001_DH1_AHU-XX_Start, DIS_AC500
    DTB001_DH1_ED, Object Type
      Functional Structure, Formal Instance List
        DTB001_DH1_ED, Object Type
          DTB001_DH1_ED-XX_Status, DIS_AC500
    DTB001_DH1_THS-CA, Object Type
      Functional Structure, Formal Instance List
        DTB001_DH1_THS-CA, Object Type
          Meas
            DTB001_DH1_THS-CA-XX_HumidVal, AIS_AC500
            DTB001_DH1_THS-CA-XX_TempVal, AIS_AC500
          Status
            DTB001_DH1_THS-CA-XX_HumidStatus, DIS_AC500
            DTB001_DH1_THS-CA-XX_TempStatus, DIS_AC500
```

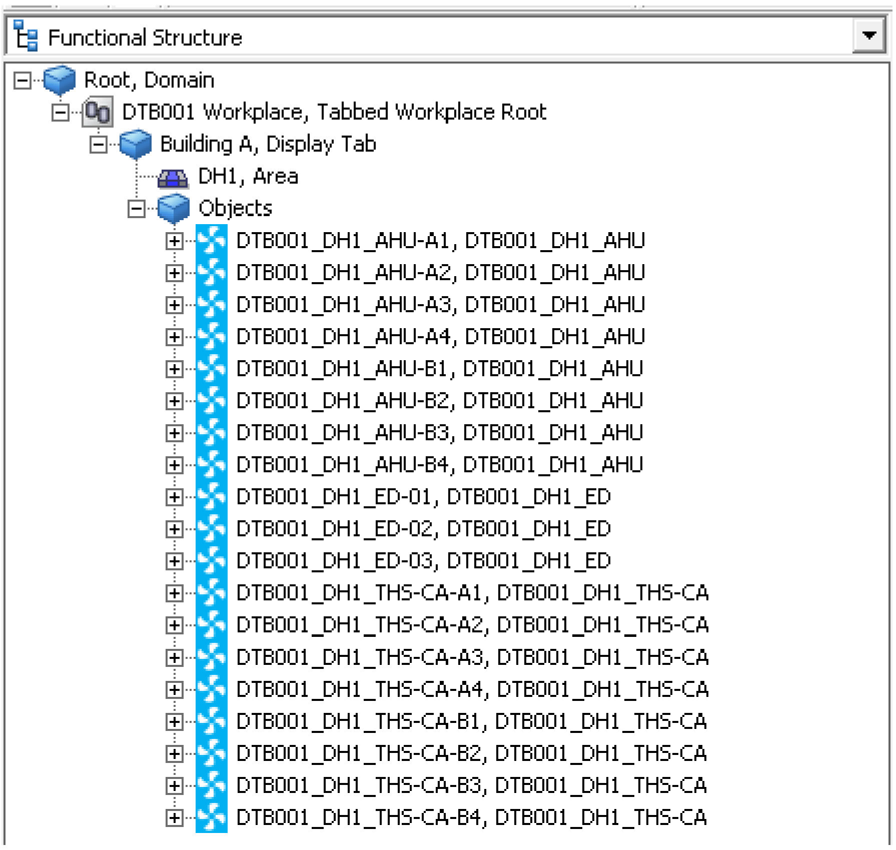
Figure 34 – Object Type Structure

Representations of analog and digital signals were also defined as object types “AIS_AC500” and “DIS_AC500” respectively. The aspects for graphical representation, history trend, control connection, alarms, and events were created for these signals.

7.2.3 Functional Structure

The Functional Structure describes the functionality of the domain. It is used for division into systems and subsystems, organizing displays, alarm sectioning, and other related functions.

The Functional Structure of the application is shown in figure 35. For this application, a workplace root named “DTB001 Workplace” is created. A display tab named “Building A” was created inside of this root, which represents the controlled building. An area named “DH1” represents a data hall, which is the subject of this application, where all field devices are located. All objects corresponding to field devices were instantiated as early created object types for each device. The names of all object instances and their analog and digital signals must coincide with the relevant names defined in the MCC application in Automation Builder to allow mapping to the signals.



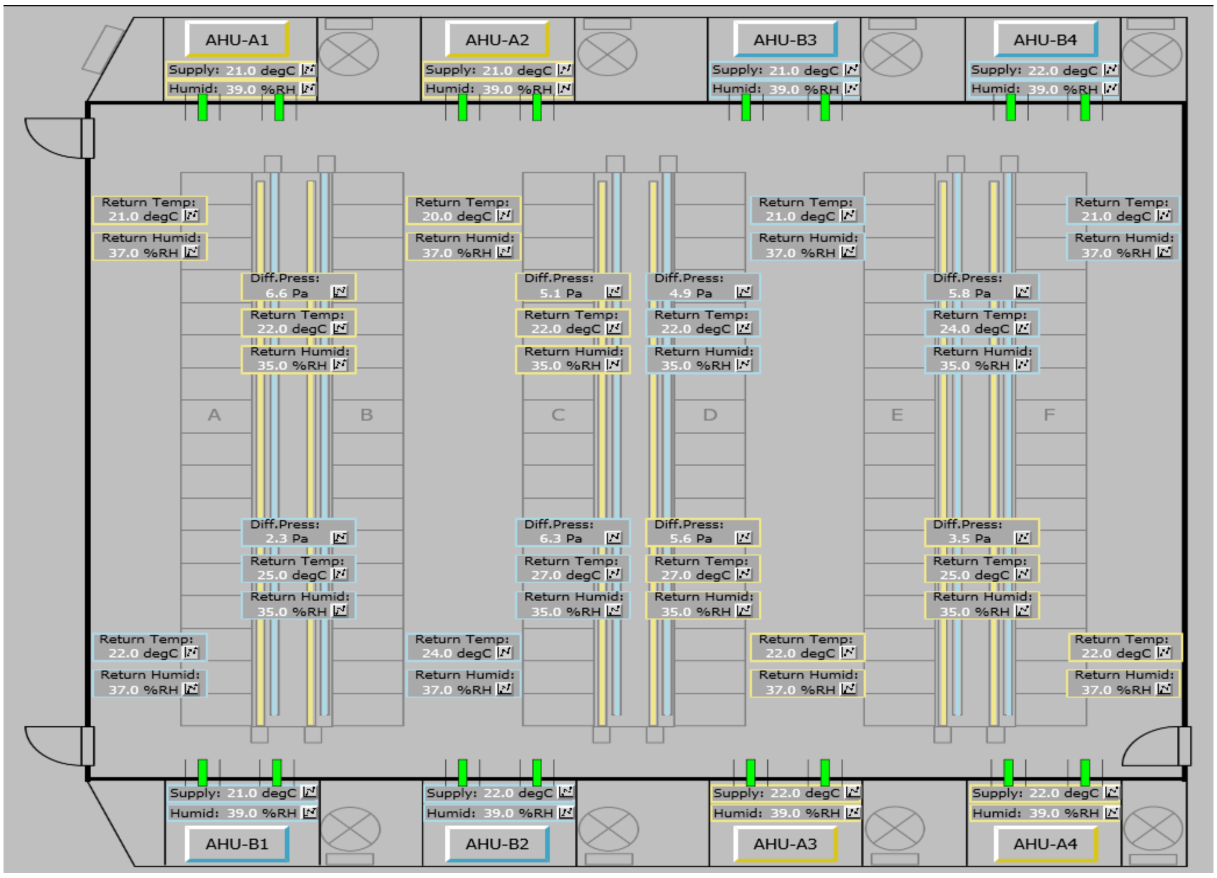
```
Functional Structure
Root, Domain
  DTB001 Workplace, Tabbed Workplace Root
    Building A, Display Tab
      DH1, Area
      Objects
        DTB001_DH1_AHU-A1, DTB001_DH1_AHU
        DTB001_DH1_AHU-A2, DTB001_DH1_AHU
        DTB001_DH1_AHU-A3, DTB001_DH1_AHU
        DTB001_DH1_AHU-A4, DTB001_DH1_AHU
        DTB001_DH1_AHU-B1, DTB001_DH1_AHU
        DTB001_DH1_AHU-B2, DTB001_DH1_AHU
        DTB001_DH1_AHU-B3, DTB001_DH1_AHU
        DTB001_DH1_AHU-B4, DTB001_DH1_AHU
        DTB001_DH1_ED-01, DTB001_DH1_ED
        DTB001_DH1_ED-02, DTB001_DH1_ED
        DTB001_DH1_ED-03, DTB001_DH1_ED
        DTB001_DH1_THS-CA-A1, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-A2, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-A3, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-A4, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-B1, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-B2, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-B3, DTB001_DH1_THS-CA
        DTB001_DH1_THS-CA-B4, DTB001_DH1_THS-CA
```

Figure 35 – Functional Structure

7.3 Data Hall Visualization

Graphical representation of the Data Hall layout and all controlled devices with corresponding signals is created using Graphics Builder. Each AHU unit placed inside its separate room is represented by a rectangle with its name inside and marked by yellow or blue color, where yellow color indicates section A and blue – section B. Doors are represented by the geometry at the corners of the Data Hall. THS-CA sensors, mounted to the walls, are not shown by the geometry but their current values are presented.

An operator can watch all the data and statuses remotely in a separate place. For this purpose, Operator Workplace is used. After everything is properly configured and communication is established, all the data will be displayed on the Data Hall Layout in Operator Workplace. Figure 36 shows the graphical representation of the Data Hall Layout with all current values of field devices received from the MCC.



Supply: 21.0 degC
Humid: 39.0 %RH
Supply: 21.0 degC
Humid: 39.0 %RH
Supply: 21.0 degC
Humid: 39.0 %RH
Supply: 22.0 degC
Humid: 39.0 %RH

Return Temp: 21.0 degC
Return Humid: 37.0 %RH
Return Temp: 20.0 degC
Return Humid: 37.0 %RH
Return Temp: 21.0 degC
Return Humid: 37.0 %RH
Return Temp: 21.0 degC
Return Humid: 37.0 %RH

Diff.Press: 6.6 Pa
Return Temp: 22.0 degC
Return Humid: 35.0 %RH
Diff.Press: 5.1 Pa
Return Temp: 22.0 degC
Return Humid: 35.0 %RH
Diff.Press: 4.9 Pa
Return Temp: 22.0 degC
Return Humid: 35.0 %RH
Diff.Press: 5.8 Pa
Return Temp: 24.0 degC
Return Humid: 35.0 %RH

Diff.Press: 2.3 Pa
Return Temp: 25.0 degC
Return Humid: 35.0 %RH
Diff.Press: 6.3 Pa
Return Temp: 27.0 degC
Return Humid: 35.0 %RH
Diff.Press: 5.6 Pa
Return Temp: 27.0 degC
Return Humid: 35.0 %RH
Diff.Press: 3.5 Pa
Return Temp: 25.0 degC
Return Humid: 35.0 %RH

Return Temp: 22.0 degC
Return Humid: 37.0 %RH
Return Temp: 24.0 degC
Return Humid: 37.0 %RH
Return Temp: 22.0 degC
Return Humid: 37.0 %RH
Return Temp: 22.0 degC
Return Humid: 37.0 %RH

Supply: 21.0 degC
Humid: 39.0 %RH
Supply: 22.0 degC
Humid: 39.0 %RH
Supply: 22.0 degC
Humid: 39.0 %RH
Supply: 22.0 degC
Humid: 39.0 %RH

Figure 36 – Current values of signals represented on the Data Hall Layout

For each AHU unit, current values of supply temperature and humidity are presented under the rectangle. Current values of differential pressure, return temperature, and humidity related to AHU units are shown at the racks where these values are collected for the corresponding unit. Supply and return air damper statuses (open/closed) are indicated by the change of geometry near each AHU room. For THS-CA sensors, current values of return temperature and humidity are shown in each cold aisle for the respective section. The doors' statuses are represented by the change of color from transparent (closed) to green (open). By clicking on a small button near each value unit, the history trend of the value changing in time can be seen.

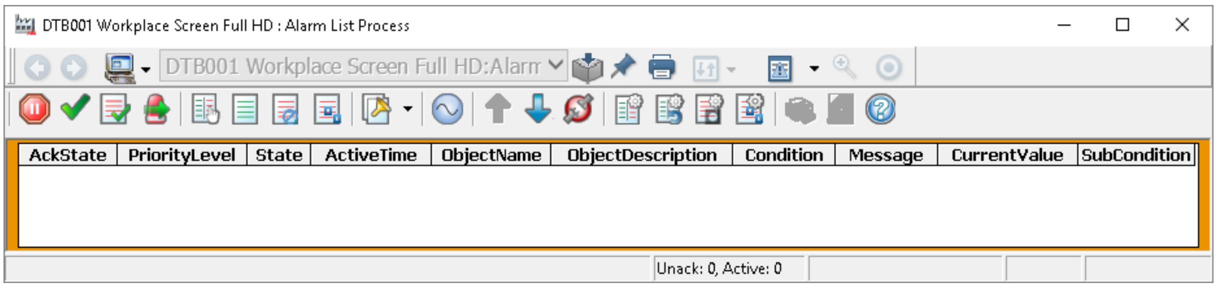
7.4 Alarm Lists

In the application for MCC, alarm states were configured for related digital and analog signals depending on the range of minimum and maximum values.

Each analog signal has four limits, exceeding which the corresponding alarm will be triggered:

- HIGH-HIGH stands at approximately 95% of the range between maximum and minimum values (Vmax - Vmin)
- HIGH – 90% of Vmax - Vmin
- LOW – 10% of Vmax - Vmin
- LOW-LOW – 5% of Vmax - Vmin

If any of the values exceed HIGH, HIGH-HIGH, LOW, or LOW-LOW limits, the corresponding alarm will be triggered. Alarm List in Operator Workplace displays currently active alarms. During the normal operation, no alarms should be activated, as shown in figure 37.

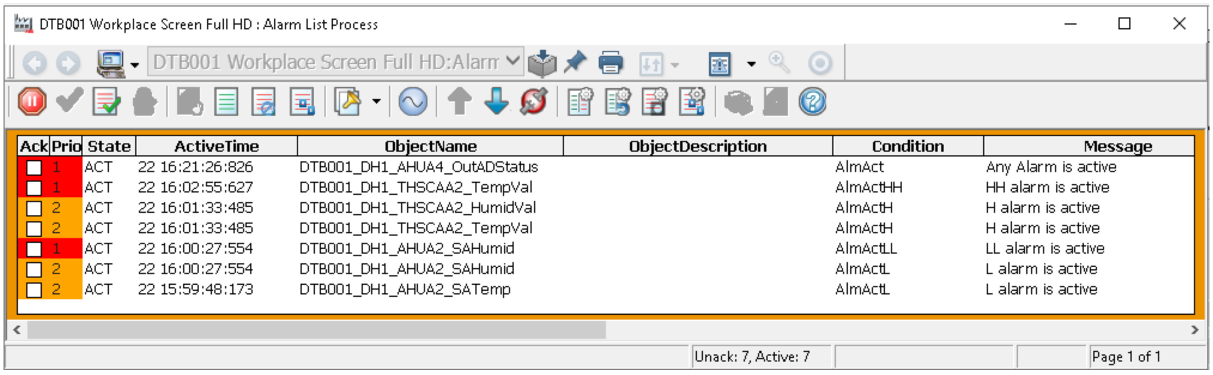

DTB001 Workplace Screen Full HD : Alarm List Process

AckState	PriorityLevel	State	ActiveTime	ObjectName	ObjectDescription	Condition	Message	CurrentValue	SubCondition

Unack: 0, Active: 0

Figure 37 – Alarm list process with no active alarms

Figure 38 demonstrates the situation in which several alarms for different binary and analog signals were triggered. Their state is specified as active or returned depending on the current value of the signal at that moment. The name of the signal for which the corresponding alarm was triggered is specified, also additional information is indicated. Alarms can be acknowledged and cleaned.


DTB001 Workplace Screen Full HD : Alarm List Process

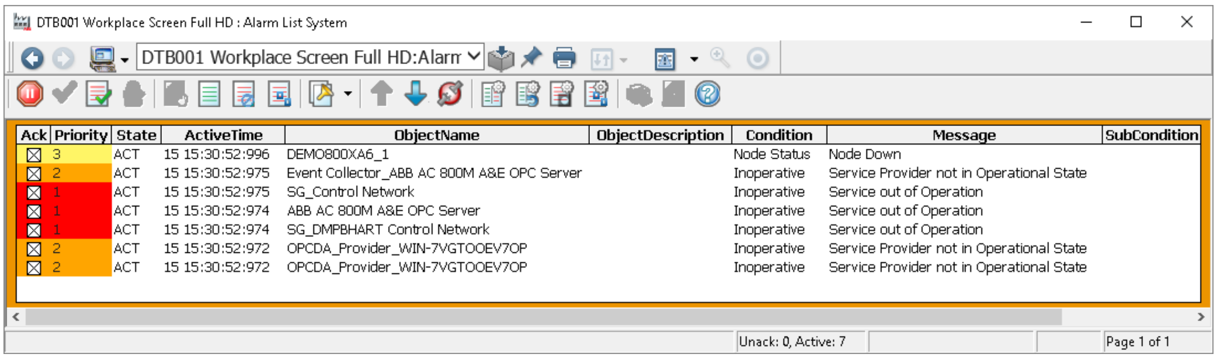
Ack	Prio	State	ActiveTime	ObjectName	ObjectDescription	Condition	Message
☐	1	ACT	22 16:21:26:826	DTB001_DH1_AHUA4_OutADStatus		AlmAct	Any Alarm is active
☐	1	ACT	22 16:02:55:627	DTB001_DH1_THSCAA2_TempVal		AlmActHH	HH alarm is active
☐	2	ACT	22 16:01:33:485	DTB001_DH1_THSCAA2_HumidVal		AlmActH	H alarm is active
☐	2	ACT	22 16:01:33:485	DTB001_DH1_THSCAA2_TempVal		AlmActH	H alarm is active
☐	1	ACT	22 16:00:27:554	DTB001_DH1_AHUA2_SAHumid		AlmActLL	LL alarm is active
☐	2	ACT	22 16:00:27:554	DTB001_DH1_AHUA2_SAHumid		AlmActL	L alarm is active
☐	2	ACT	22 15:59:48:173	DTB001_DH1_AHUA2_SATemp		AlmActL	L alarm is active

Unack: 7, Active: 7

Page 1 of 1

Figure 38 – Alarm list process with triggered alarms

Operator Workplace also includes system alarms, which are triggered when some service stopped working. In figure 39, some alarms are triggered because a physical server is missing to meet all requirements needed to run all general services of the 800xA System. In the same way, as for process alarms, their state is specified as active or returned depending on the current value of the service at that moment. The name of the service for which the corresponding alarm was triggered is specified, also additional information is indicated. Alarms can be acknowledged and cleaned.


DTB001 Workplace Screen Full HD : Alarm List System

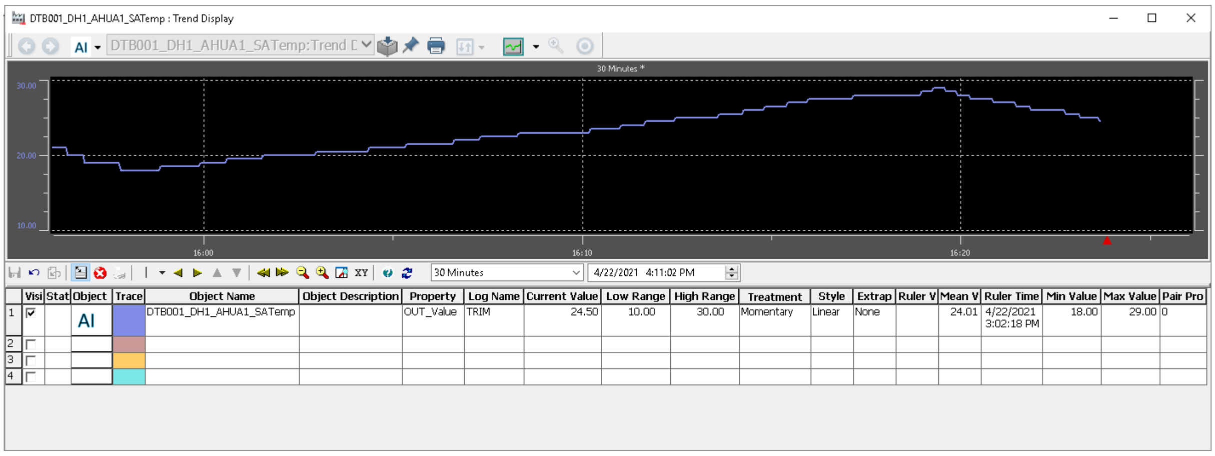
Ack	Priority	State	ActiveTime	ObjectName	ObjectDescription	Condition	Message	SubCondition
☒	3	ACT	15 15:30:52:996	DEMO800XA6_1		Node Status	Node Down	
☒	2	ACT	15 15:30:52:975	Event Collector_ABB AC 800M A&E OPC Server		Inoperative	Service Provider not in Operational State	
☒	1	ACT	15 15:30:52:975	SG_Control Network		Inoperative	Service out of Operation	
☒	1	ACT	15 15:30:52:974	ABB AC 800M A&E OPC Server		Inoperative	Service out of Operation	
☒	1	ACT	15 15:30:52:974	SG_DMPBHART Control Network		Inoperative	Service out of Operation	
☒	2	ACT	15 15:30:52:972	OPCDA_Provider_WIN-7VGTOOEV7OP		Inoperative	Service Provider not in Operational State	
☒	2	ACT	15 15:30:52:972	OPCDA_Provider_WIN-7VGTOOEV7OP		Inoperative	Service Provider not in Operational State	

Unack: 0, Active: 7 Page 1 of 1

Figure 39 – Alarm list system

7.5 History Trends

The Operator Workplace provides an opportunity to display history trends of changing analog signal value in a specified period of time. Figure 40 shows an example of changing the supply air temperature value from the AHU-A1 unit during the previous 30 minutes. Also, several trends can be shown in one graph, the history trend can be exported in ".csv" file.



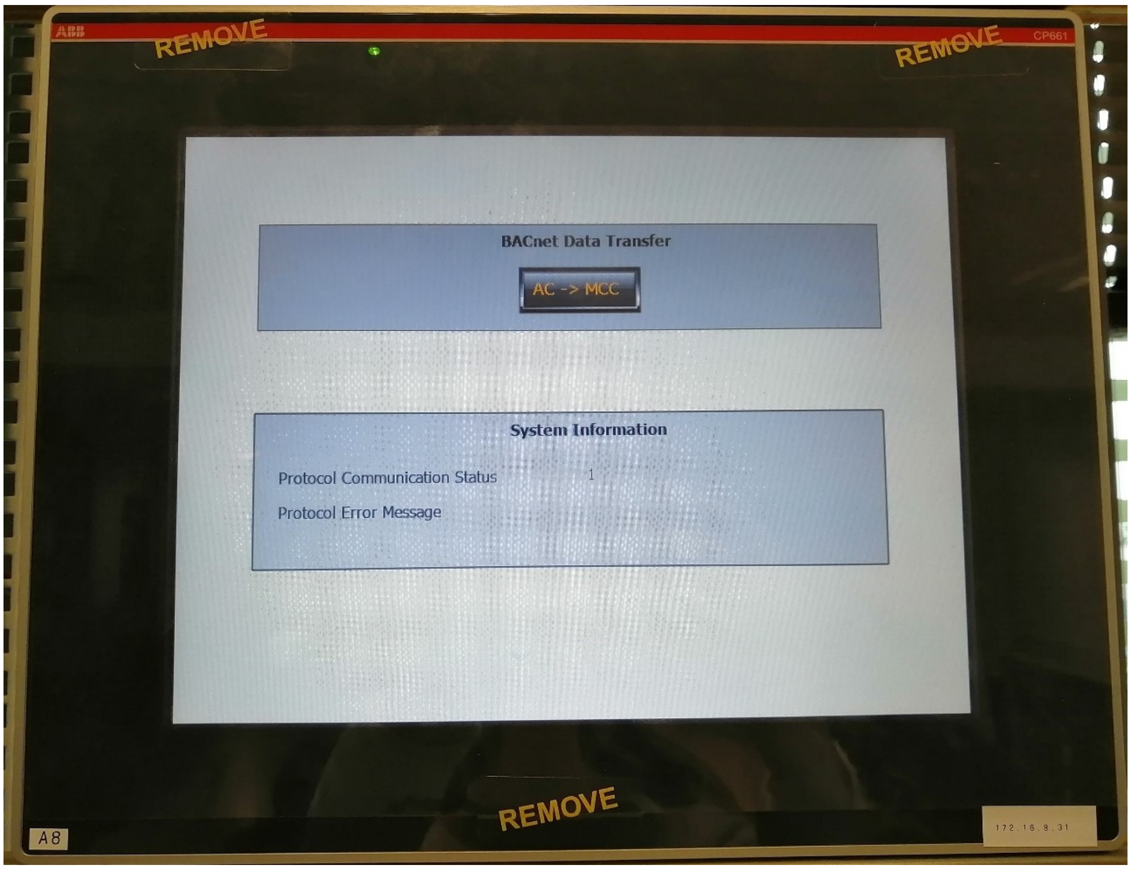
	Visi	Stat	Object	Trace	Object Name	Object Description	Property	Log Name	Current Value	Low Range	High Range	Treatment	Style	Extrap	Ruler V	Mean V	Ruler Time	Min Value	Max Value	Pair Pro
1	☑		AI	🟦	DTB001_DH1_AHUA1_SATemp		OUT_Value	TRIM	24.50	10.00	30.00	Momentary	Linear	None		24.01	4/22/2021 3:02:18 PM	18.00	29.00	0
2	☐			🟫																
3	☐			🟨																
4	☐			🟦																

Figure 40 – History trend of an analog signal for the specified period

8 Testing and Results

8.1 Quality of BACnet Communication

The Control panel provides options for displaying system information, for example, protocol communication status, protocol error message. For this project, a simple HMI screen was created for the control panel project to display basic information about the status of BACnet communication. The HMI screen is shown in figure 41. In the presented picture, protocol communication status is equal to 1 that designates correct work of data transfer and no protocol error message is shown.

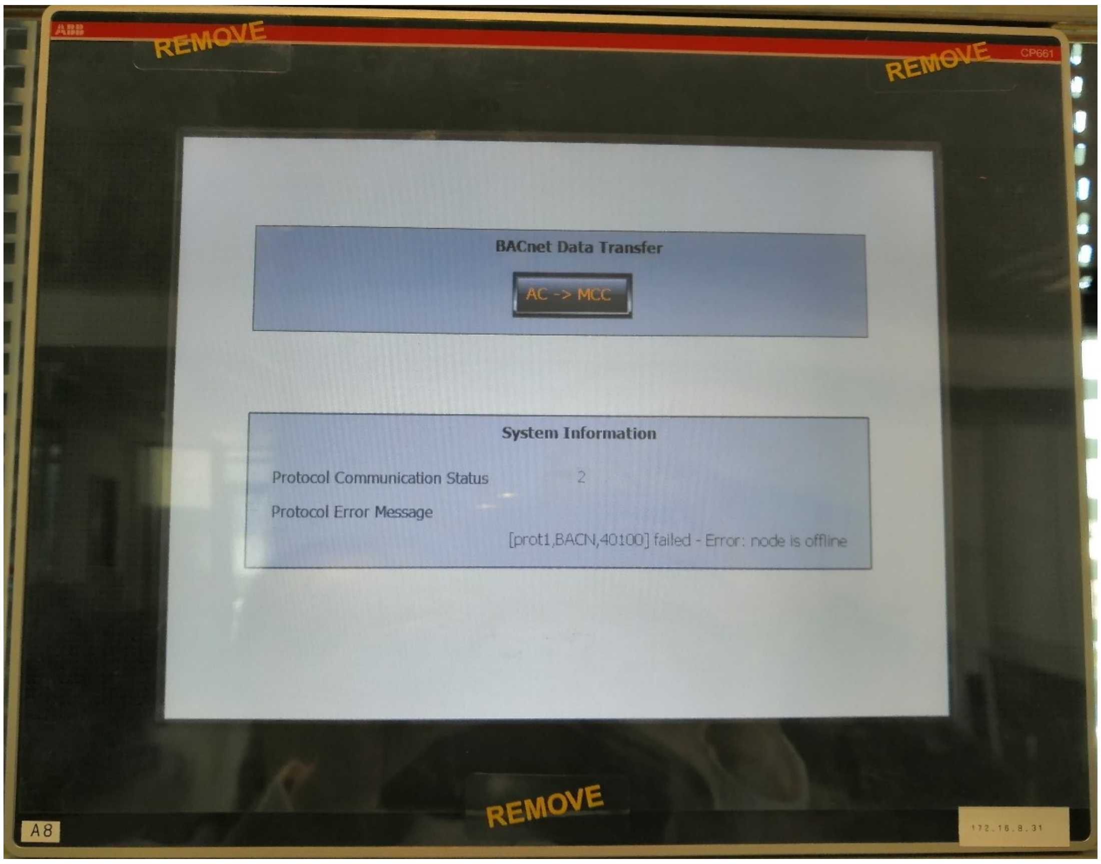


Protocol Communication Status 1

Protocol Error Message

Figure 41 – HMI screen with BACnet protocol communication status

When some error occurs, protocol communication status displays the error code and protocol error message is specified. This example is demonstrated in figure 42, where the error code is equal to 2 and the message “[prot1, BACN, 40100] failed – Error: node is offline” appears.


BACnet Data Transfer

AC -> MCC

System Information

Protocol Communication Status 2

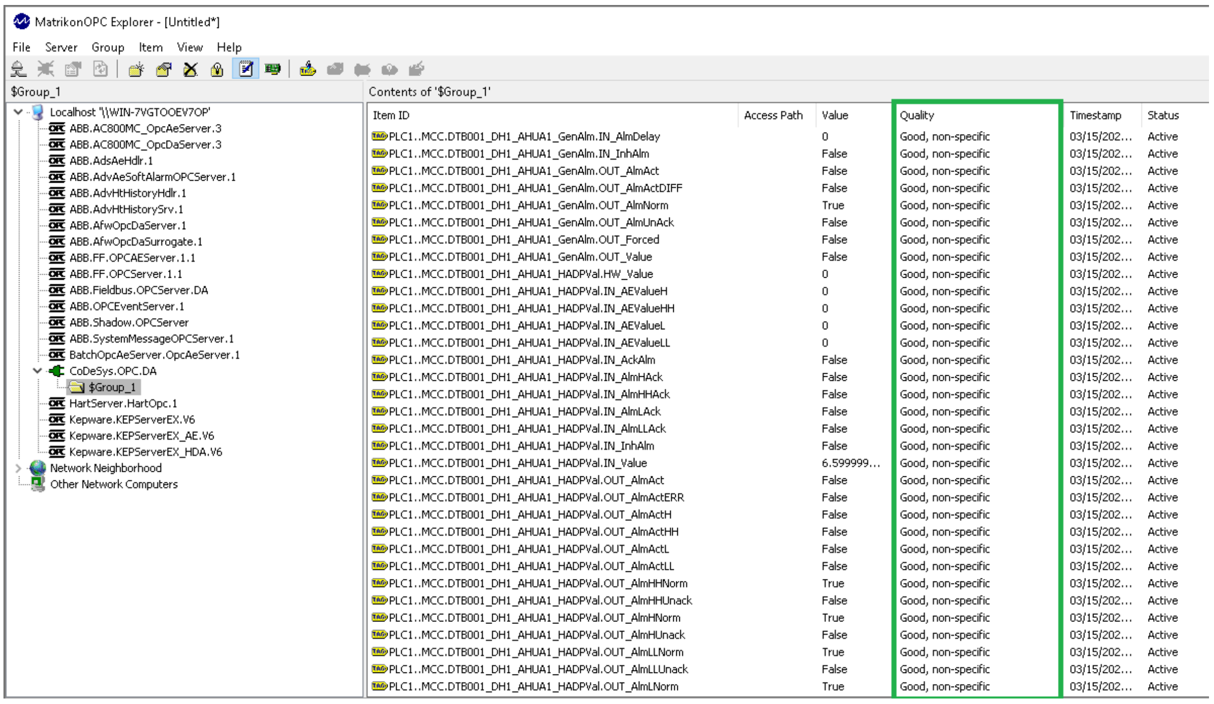
Protocol Error Message

[prot1,BACN,40100] failed - Error: node is offline

Figure 42 – BACnet protocol communication status with error

8.2 Quality of Data Transfer

Matrikon OPC Explorer is used as the OPC test client to check the connection to the PLC and the quality of data. “Codesys.OPC.DA” is the service that provides communication. After connecting to the server, the Application_MCC.sdb file is uploaded to the client side, which contains the symbol configuration file created in the PLC MCC application. All the tags existing in this configuration are available under the service. Figure 43 provides part of the list of available tags used in the project. The column “Quality” proves that the quality of all values is good and the transfer between the OPC client and the OPC server is established properly.


MatrikonOPC Explorer - [Untitled*]

File Server Group Item View Help

$Group_1

- Localhost '\\WIN-7VGTOOEV7OP'
  - ABB.AC800MC_OpcAeServer.3
  - ABB.AC800MC_OpcDaServer.3
  - ABB.AdsAeHdlr.1
  - ABB.AdvAeSoftAlarmOPCServer.1
  - ABB.AdvHtHistoryHdlr.1
  - ABB.AdvHtHistorySrv.1
  - ABB.AfwOpcDaServer.1
  - ABB.AfwOpcDaSurrogate.1
  - ABB.FF.OPCAEServer.1.1
  - ABB.FF.OPCServer.1.1
  - ABB.Fieldbus.OPCServer.DA
  - ABB.OPCEventServer.1
  - ABB.Shadow.OPCServer
  - ABB.SystemMessageOPCServer.1
  - BatchOpcAeServer.OpcAeServer.1
  - CoDeSys.OPC.DA
    - $Group_1
  - HartServer.HartOpc.1
  - Kepware.KEPServerEX.V6
  - Kepware.KEPServerEX_AE.V6
  - Kepware.KEPServerEX_HDA.V6
- Network Neighborhood
- Other Network Computers

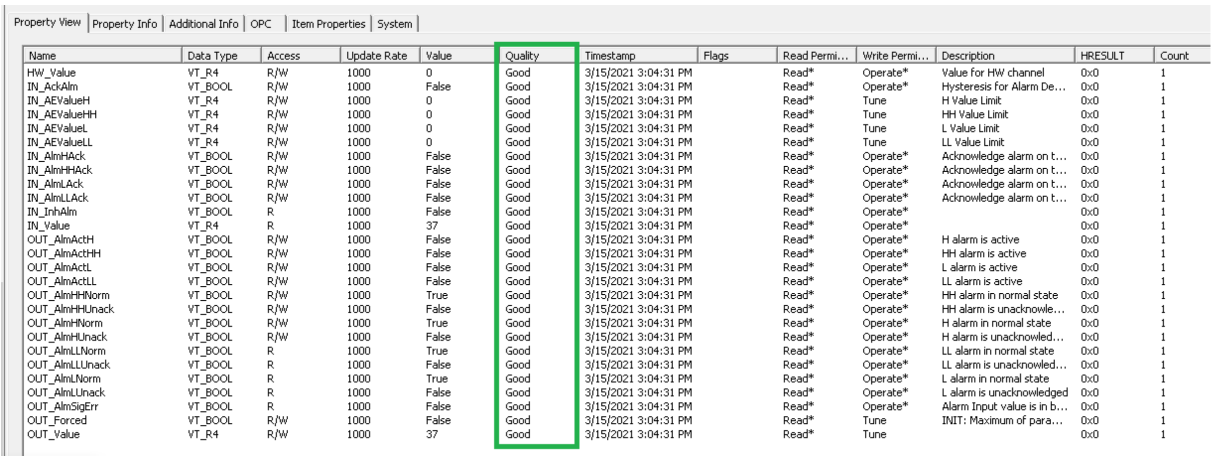
Contents of '$Group_1'

Item ID	Access Path	Value	Quality	Timestamp	Status
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.IN_AlmDelay		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.IN_InhAlm		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_AlmAct		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_AlmActDIFF		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_AlmNorm		True	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_AlmUnAck		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_Forced		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_GenAlm.OUT_Value		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.HW_Value		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AEValueH		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AEValueHH		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AEValueL		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AEValueLL		0	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AckAlm		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AlmHAck		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AlmHHAck		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AlmLAck		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_AlmLLAck		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_InhAlm		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.IN_Value		6.599999…	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmAct		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmActERR		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmActH		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmActHH		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmActL		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmActLL		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmHHNorm		True	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmHHUnack		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmHNorm		True	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmHUnack		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmLLNorm		True	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmLLUnack		False	Good, non-specific	03/15/202…	Active
PLC1..MCC.DTB001_DH1_AHUA1_HADPVal.OUT_AlmLNorm		True	Good, non-specific	03/15/202…	Active

Figure 43 – Quality of data transfer via OPC

After making sure that the communication works well, the quality of inputs and outputs of all objects in the Engineering Workplace must be checked.

Figure 44 shows that the data quality of all inputs and outputs in the selected analog signal is good.

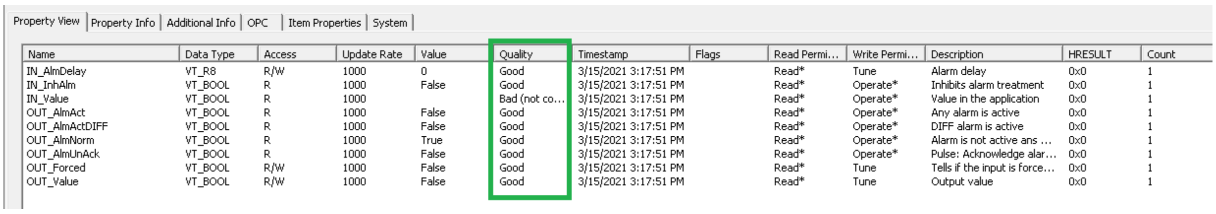

Property View | Property Info | Additional Info | OPC | Item Properties | System

Name	Data Type	Access	Update Rate	Value	Quality	Timestamp	Flags	Read Permi…	Write Permi…	Description	HRESULT	Count
HW_Value	VT_R4	R/W	1000	0	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Value for HW channel	0x0	1
IN_AckAlm	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Hysteresis for Alarm De…	0x0	1
IN_AEValueH	VT_R4	R/W	1000	0	Good	3/15/2021 3:04:31 PM		Read*	Tune	H Value Limit	0x0	1
IN_AEValueHH	VT_R4	R/W	1000	0	Good	3/15/2021 3:04:31 PM		Read*	Tune	HH Value Limit	0x0	1
IN_AEValueL	VT_R4	R/W	1000	0	Good	3/15/2021 3:04:31 PM		Read*	Tune	L Value Limit	0x0	1
IN_AEValueLL	VT_R4	R/W	1000	0	Good	3/15/2021 3:04:31 PM		Read*	Tune	LL Value Limit	0x0	1
IN_AlmHAck	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Acknowledge alarm on t…	0x0	1
IN_AlmHHAck	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Acknowledge alarm on t…	0x0	1
IN_AlmLAck	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Acknowledge alarm on t…	0x0	1
IN_AlmLLAck	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Acknowledge alarm on t…	0x0	1
IN_InhAlm	VT_BOOL	R	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*		0x0	1
IN_Value	VT_R4	R	1000	37	Good	3/15/2021 3:04:31 PM		Read*	Operate*		0x0	1
OUT_AlmActH	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	H alarm is active	0x0	1
OUT_AlmActHH	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	HH alarm is active	0x0	1
OUT_AlmActL	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	L alarm is active	0x0	1
OUT_AlmActLL	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	LL alarm is active	0x0	1
OUT_AlmHHNorm	VT_BOOL	R/W	1000	True	Good	3/15/2021 3:04:31 PM		Read*	Operate*	HH alarm in normal state	0x0	1
OUT_AlmHHUnack	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	HH alarm is unacknowle…	0x0	1
OUT_AlmHNorm	VT_BOOL	R/W	1000	True	Good	3/15/2021 3:04:31 PM		Read*	Operate*	H alarm in normal state	0x0	1
OUT_AlmHUnack	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	H alarm is unacknowled…	0x0	1
OUT_AlmLLNorm	VT_BOOL	R	1000	True	Good	3/15/2021 3:04:31 PM		Read*	Operate*	LL alarm in normal state	0x0	1
OUT_AlmLLUnack	VT_BOOL	R	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	LL alarm is unacknowled…	0x0	1
OUT_AlmLNorm	VT_BOOL	R	1000	True	Good	3/15/2021 3:04:31 PM		Read*	Operate*	L alarm in normal state	0x0	1
OUT_AlmLUnack	VT_BOOL	R	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	L alarm is unacknowledged	0x0	1
OUT_AlmSigErr	VT_BOOL	R	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Operate*	Alarm Input value is in b…	0x0	1
OUT_Forced	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:04:31 PM		Read*	Tune	INIT: Maximum of para…	0x0	1
OUT_Value	VT_R4	R/W	1000	37	Good	3/15/2021 3:04:31 PM		Read*	Tune		0x0	1

Figure 44 – Data quality of analog signal

Figure 45 shows that the data quality of all inputs and outputs in the selected digital signal is good.

Note: Signal IN_Value is not part of the analog signal configuration in CoDeSys, and therefore is not connected, since OUT_Value is used for data transfer. IN_Value property holds a value of the signal before applying user forcing methods. OUT_Value holds a value of the signal after applying user forcing methods.



Property View | Property Info | Additional Info | OPC | Item Properties | System

Name	Data Type	Access	Update Rate	Value	Quality	Timestamp	Flags	Read Permi…	Write Permi…	Description	HRESULT	Count
IN_AlmDelay	VT_R8	R/W	1000	0	Good	3/15/2021 3:17:51 PM		Read*	Tune	Alarm delay	0x0	1
IN_InhAlm	VT_BOOL	R	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Operate*	Inhibits alarm treatment	0x0	1
IN_Value	VT_BOOL	R	1000		Bad (not co…	3/15/2021 3:17:51 PM		Read*	Operate*	Value in the application	0x0	1
OUT_AlmAct	VT_BOOL	R	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Operate*	Any alarm is active	0x0	1
OUT_AlmActDIFF	VT_BOOL	R	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Operate*	DIFF alarm is active	0x0	1
OUT_AlmNorm	VT_BOOL	R	1000	True	Good	3/15/2021 3:17:51 PM		Read*	Operate*	Alarm is not active ans …	0x0	1
OUT_AlmUnAck	VT_BOOL	R	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Operate*	Pulse: Acknowledge alar…	0x0	1
OUT_Forced	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Tune	Tells if the input is force…	0x0	1
OUT_Value	VT_BOOL	R/W	1000	False	Good	3/15/2021 3:17:51 PM		Read*	Tune	Output value	0x0	1

Figure 45 – Data quality of digital signal

8.3 Comparison of BMS Protocols

Table 16 provides the comparison of BACnet protocol with other widely used solutions in BMS to bring further development opportunities.

Table 16 – Comparison between BACnet, Modbus, LonWorks, and KNX

Protocol / Parameter	BACnet	Modbus	LonWorks	KNX
Use	Communication across devices	Connection between devices	Networking devices through power lines, fiber optics, and other media	Device to device communication
Proprietary	No	No	Yes	No
Transmission Modes	Ethernet, IP, MS/TP, ARCNET, PTP	ASCII, RTU, TCP/IP	MS/TP, network, SNVT	TP, PL, IP, RF
Costs	Low; No charge for usage or licensing fees	Low; No charge for usage or licensing fees	High; Limited users	Low
Advantages	1. Scalability between cost, performance, and system size 2. Wide variety of media and topologies 3. Easily extensible to add new functionality 4. Supported by many manufacturers	1. Suitable for small/medium volumes of data (<=255 bytes) 2. Scalable for remote monitoring 3. Easy to deploy and maintain 4. Moves raw bits or words without placing restrictions on vendors	1. Less architecture at the device level 2. Web-based tool; saves time and cost 3. Products are developed by many companies 4. Devices are close to “plug & play” ability	1. Decentralized network structure 2. Independent on the platform 3. Low energy consumption 4. Supported by many manufacturers 5. Tree topology appropriate for large networks

Protocol / Parameter	BACnet	Modbus	LonWorks	KNX
Disadvantages	1. Lacks the network security features 2. Requires more memory for storage 3. Not a plug-and-play solution	1. No security against unauthorized commands 2. Limited number of remote communication devices 3. Complicated process of configuration and programming	1. Controlled devices and variables are connected to a separate control device 2. Limited possibility of extensions and standard development 3. Hardware specific	1. Network can be controlled only from an 8-bit microcontroller to a PC 2. The main line allows only up to 64 devices 3. Does not allow loop topologies 4. Relatively small community

An open system ensures support for lifetime projects, because of its four main aspects, such as open, interoperable, multi-vendor, and end-to-end solution. KNX system is a better approach for home automation, but the BACnet system is a more solid solution for building offices. BACnet seems to be the most flexible solution providing extensible opportunities for developers and manufacturers.

From the engineering point of view, BACnet protocol is easier in the configuration of all properties, including alarms configuration, the difference is significant in big projects. Also, BACnet is quicker for simulation and troubleshooting. Addressing modes are simpler in BACnet because it allows work with variables directly so that the data do not need to be saved in registers.

From the perspective of additional devices, the BACnet protocol does not require routers for communication within one network media. But since a PLC can act as a server only, a client is required to start the communication. This imposes additional costs on the system.

8.4 Evaluation of the Results

During the elaboration of this work, the communication between two PLCs AC500 via BACnet protocol was successfully established. The implemented solution is simple and effective and can be used for communication between master and area controllers in BMS applications.

The connectivity to the higher monitoring system 800xA was realized via OPC DA protocol. The communication was well-established for all used signals and their properties (figures 43, 44, 45).

The implemented system successfully demonstrates the long-term stability of individual services of the 800xA System and MCC PLC. It was the basic prerequisite for future use on larger systems. Considering the efficiency of selected engineering methods, the chances of future development are increased even more.

Conclusion

The goal of this work was to establish a modular BMS monitoring system that provides data acquisition from BMS field devices inside a data center. Considering the requirements for used PLC as a data center area and master controllers, a PLC AC500 manufactured by ABB was selected. During the elaboration of this work, knowledge about BACnet communication and its use in building automation was obtained.

The result of this work is a fully functional solution consisting of two PLCs AC500 and control panel CP600, which exchange information with each other using the communication protocol BACnet. The control panel serves as a gateway (client) in the communication between two PLCs, where both can act as BACnet servers only. It requests the data from one PLC representing the area controller and sends the present values to another PLC representing the master coordinating controller. The resulting network is also equipped with a user interface for remote control, monitoring, and data archiving.

The selected individual elements created a compact and portable system, which is suitable for training purposes and also serves as practical support for the implementation of a more comprehensive solution. This work will be used by ABB for training purposes of BMS monitoring systems based on BACnet protocol and AC500 PLCs with its further connectivity to the 800xA System.

The logic of the designed communication system covers data exchange between all levels of the three-tier automation hierarchy. Physical installation does not include field devices, their behavior is simulated in the area controller program. The control application conforms to the B-ASC device profile, the library provided by ABB is used for the implementation.

Graphical representation of the Data Hall layout and all controlled devices with corresponding signals is displayed in the Operator Workplace. It provides monitoring of current values and statuses, control of triggered alarms, and history trends of analog signals in a specified period of time. The quality of communication and transferred data was tested, which confirms the correct implementation of the communication system. The update screen rate on Operator Workplace which has been set to 3s was an observer for all displayed signals.

BACnet protocol was compared with other widely used BMS solutions in terms of engineering complexity and efficiency. BACnet protocol has proven to be quicker in simulation, troubleshooting, and system extension. On the other hand, since the PLC can act only as a server, a client is required for communication, which imposes additional costs to the system.

Further development opportunities of this work are to create a more sophisticated communication system. Since BACnet is frequently used at all levels of the three-tier model, the functionality of two different protocols can be merged into BACnet protocol. Many manufacturers of field devices integrate support of BACnet protocol, therefore this solution would simplify the engineering phase by combining the communication across field devices and PLCs up to the higher monitoring system.

References

[1] WILAMOWSKI, Bogdan M. and J. David IRWIN, *ed. IndustrIal CommunicatIon Systems: The Industrial Electronics Handbook.* 2nd edition. Boca Raton, FL: CRC Press, 2011. ISBN 978-1-4398-0281-6.

[2] MERZ, Hermann, Thomas HANSEMANN and Christof HÜBNER. *Building Automation. Communication Systems with EIB/KNX, LON and BACnet*. Germany: Springer-Verlag Berlin Heidelberg, 2009. ISBN 978-3-540-88828-4.

[3] IEC 61558. *Industrial communication networks – Fieldbus specifications.* Edition 2.0. Geneva, Switzerland: IEC, 2019.

[4] NEWMAN, H. Michael. *BACnet: The Global Standard for Building Automation and Control Networks*. 1st edition. New York: Momentum Press, 2013. ISBN 978-1606502884.

[5] ZURAWSKI, Richard, ed. *Industrial Communication Technology Handbook*. Second Edition. Boca Raton, FL: CRC Press, 2015. ISBN 978-1-4822-0733-0.

[6] ANSI/ASHRAE STANDARD 135-2012. *A Data Communication Protocol for Building Automation and Control Networks: BACnet*. 1st edition. Atlanta, GA: ASHRAE, 2012.

[7] IEC 60929. *AC and/or DC-supplied electronic control gear for tubular fluorescent lamps.* Edition 4.0. Geneva, Switzerland: IEC, 2011.

[8] OPC Classic: OPC Technologies. *OPC Foundation* [online]. OPC Foundation [cit. 2021-03-29]. Available from: https://opcfoundation.org/about/opc-technologies/opc-classic/

[9] *PLC Automation: PLCs, Control Panels, Engineering Suite AC500, CP600, Automation Builder*. Germany: ABB, 2019.

[10] *AC500 PLC: System Assembly and Device Specifications for AC500 V2 Products*. Germany: ABB, 2019.

[11] CPUs - AC500 (Programmable Logic Controllers PLCs). *ABB* [online]. Global site: ABB, 2020 [cit. 2021-03-16]. Available from: https://new.abb.com/plc/programmable-logic-controllers-plcs/ac500/cpus

[12] EDS-408A Series: Layer 2 Managed Switches. *MOXA* [online]. Moxa Inc., 2021 [cit. 2021-03-16]. Available from: https://www.moxa.com/en/products/industrial-network-infrastructure/ethernet-switches/layer-2-managed-switches/eds-408a-series

[13] CP600 control panels - CP600 control panels platform. *ABB* [online]. Global site: ABB, 2020 [cit. 2021-03-16]. Available from: https://new.abb.com/plc/control-panels/cp600

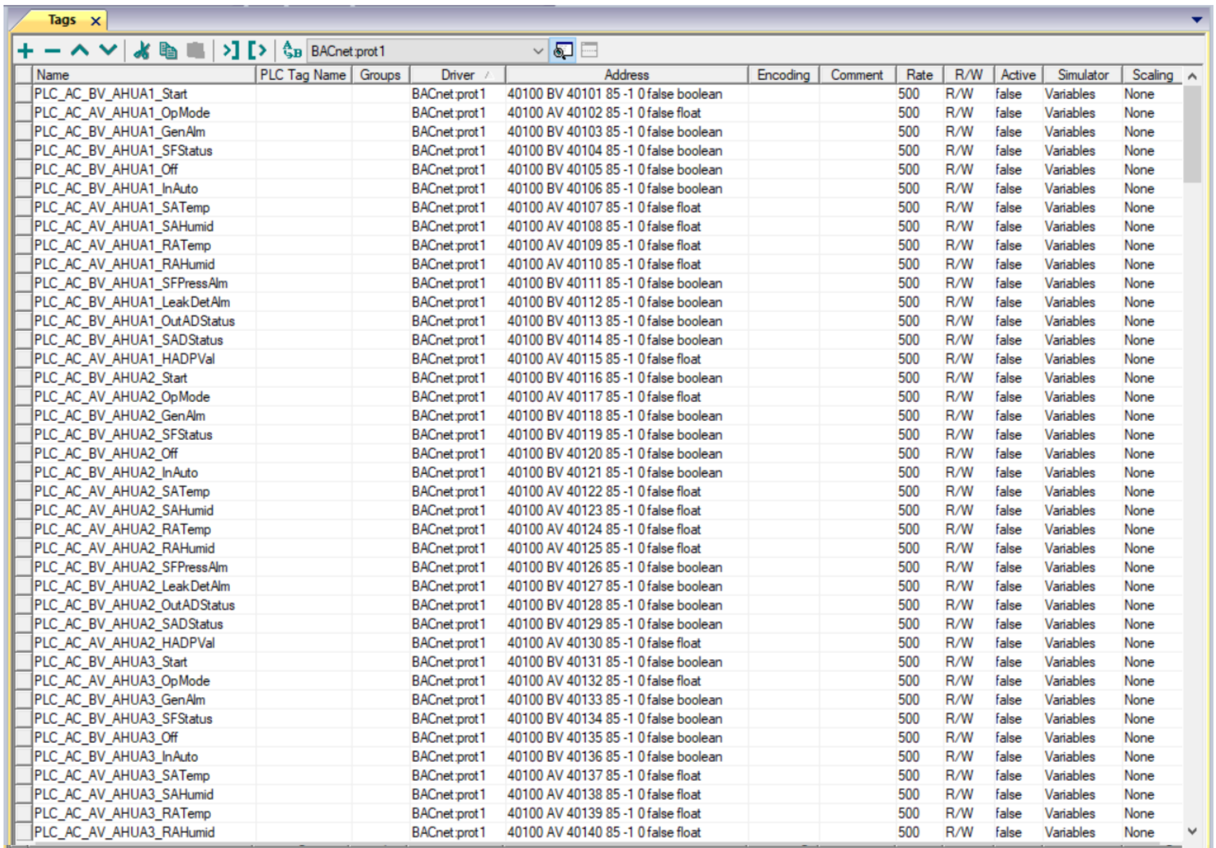
[14] ABB System 800xA - process, electrical, safety, telecoms in one system. *ABB* [online]. Global site: ABB, 2021 [cit. 2021-03-29]. Available from: https://new.abb.com/control-systems/system-800xa

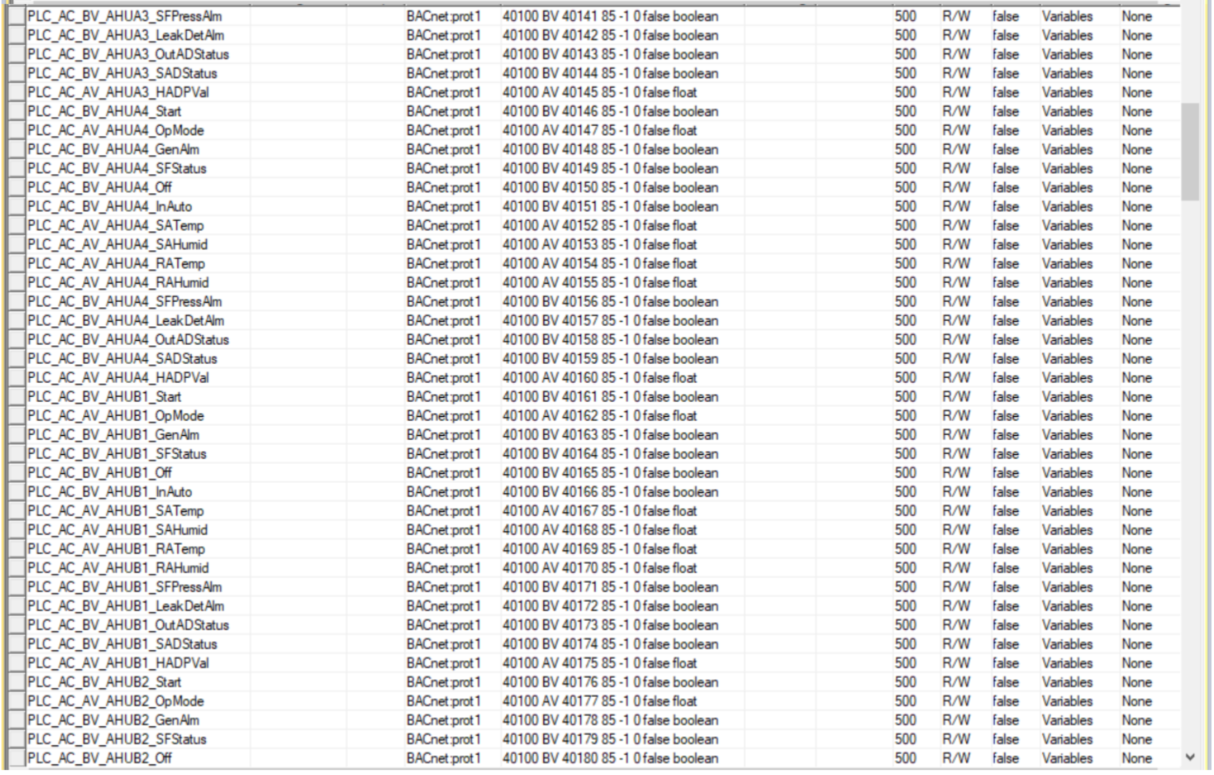
[15] *System 800xA: System Guide – Functional Description* [online]. 588. ABB, 2014, [cit. 2021-03-29]. Available from: https://search.abb.com/library/Download.aspx?DocumentID=3BSE038018-600&LanguageCode=en&DocumentPartId=&Action=Launch

[16] ABB. *AC500 BACNET IP: Data exchange between 2 CPUs via CP600 gateway*. Heidelberg, Germany, 2018.

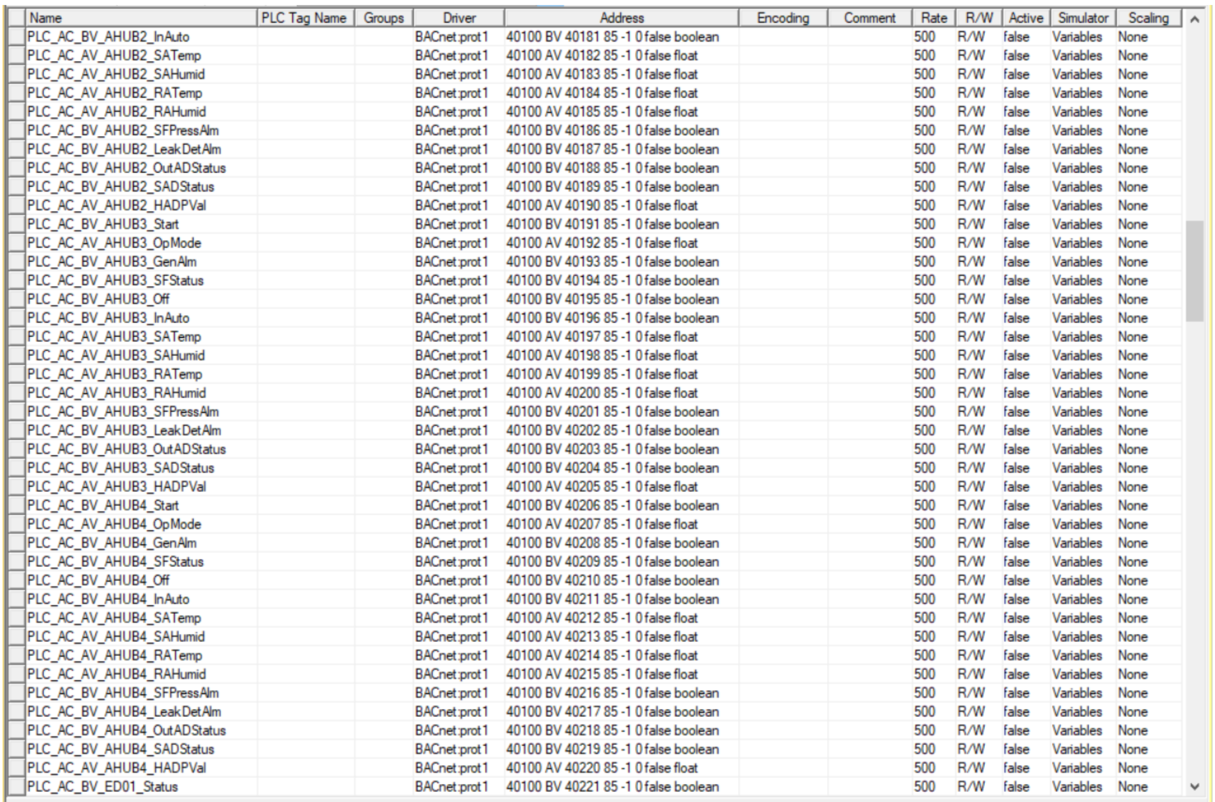
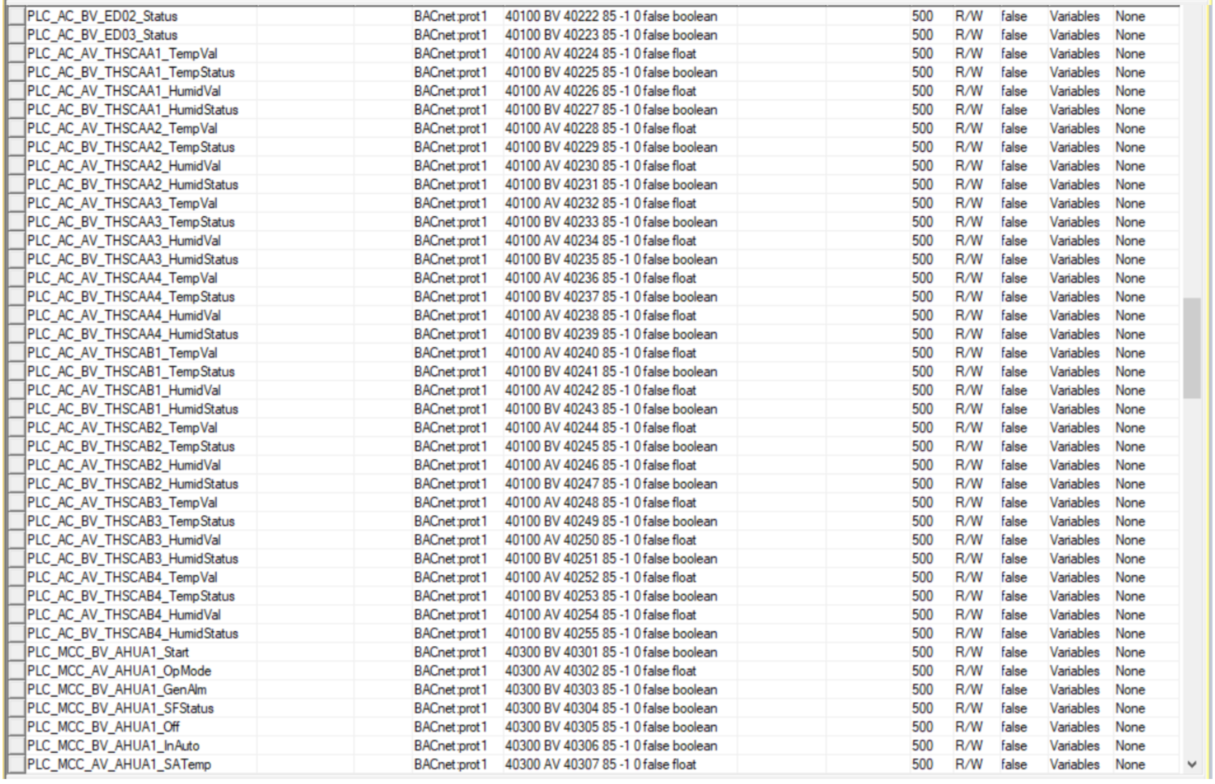
Appendices

Appendix A: List of tags used in the application for control panel

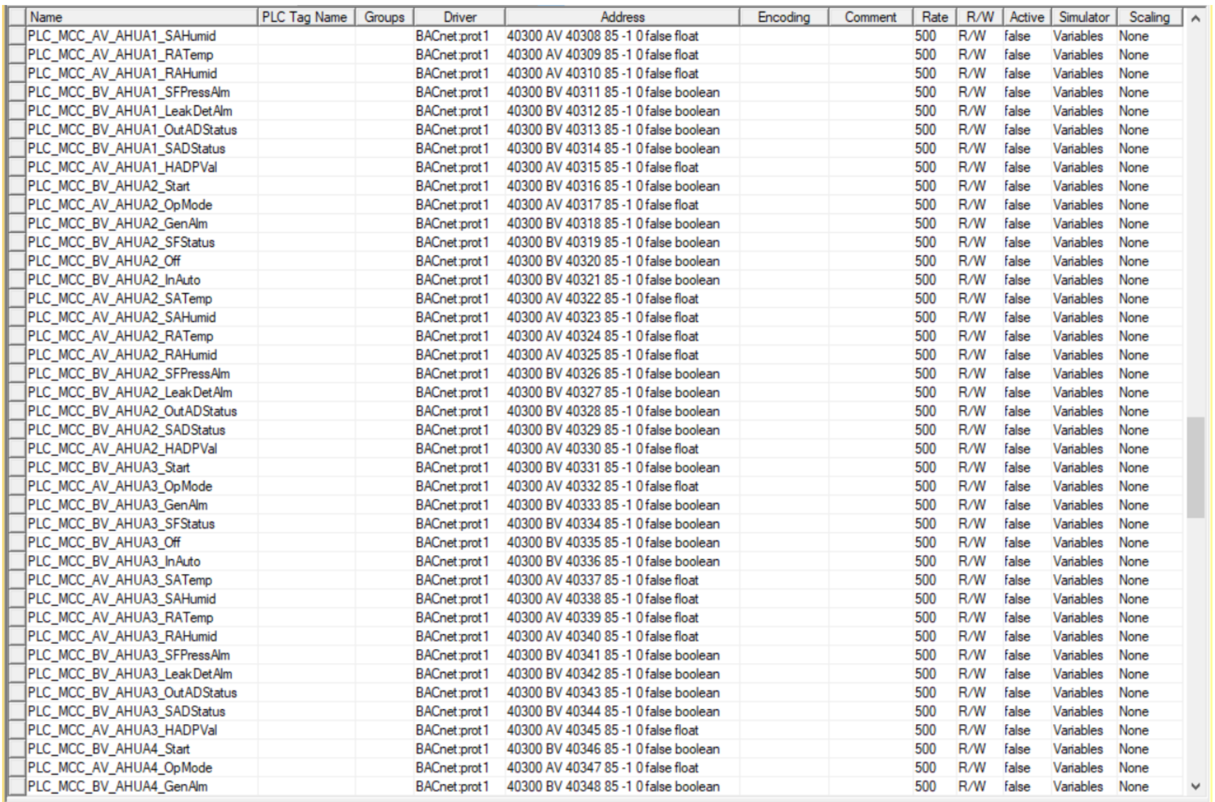


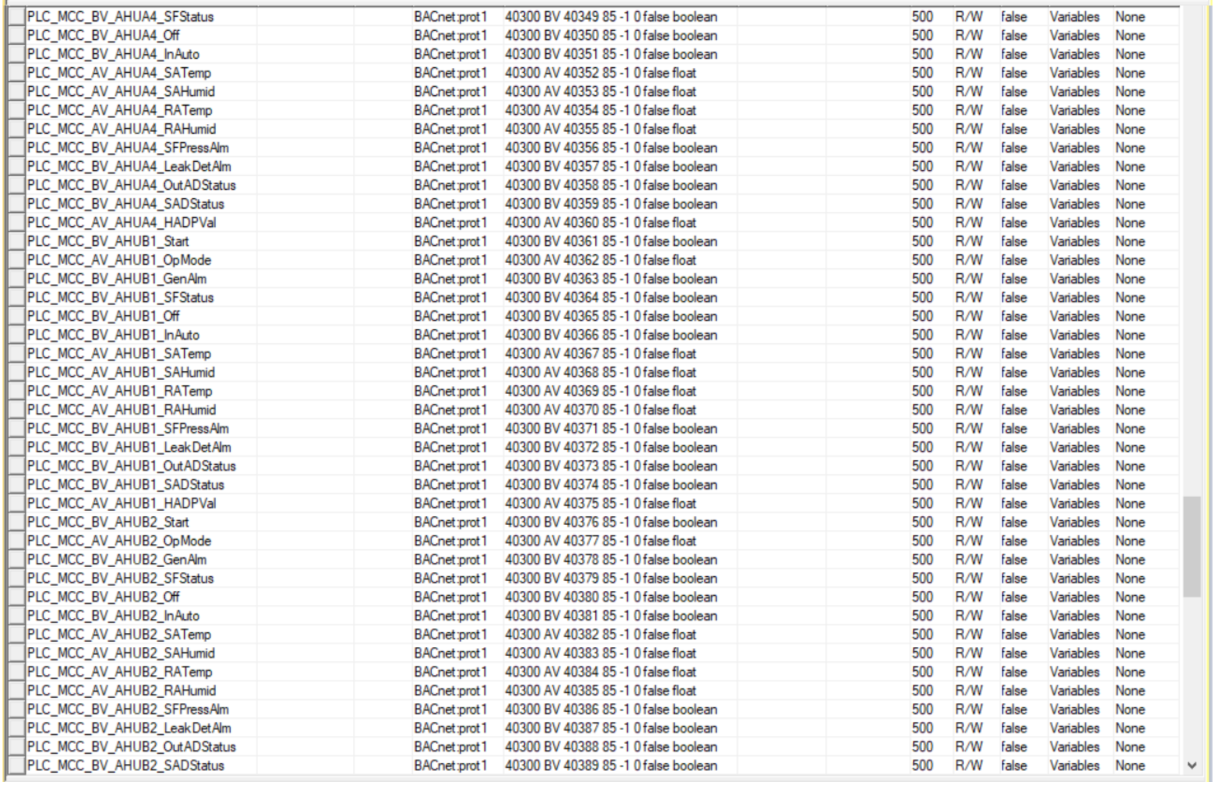


Name	PLC Tag Name	Groups	Driver	Address	Encoding	Comment	Rate	R/W	Active	Simulator	Scaling
PLC_AC_BV_AHUA1_Start			BACnet:prot1	40100 BV 40101 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_OpMode			BACnet:prot1	40100 AV 40102 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_GenAlm			BACnet:prot1	40100 BV 40103 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SFStatus			BACnet:prot1	40100 BV 40104 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_Off			BACnet:prot1	40100 BV 40105 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_InAuto			BACnet:prot1	40100 BV 40106 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_SATemp			BACnet:prot1	40100 AV 40107 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_SAHumid			BACnet:prot1	40100 AV 40108 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_RATemp			BACnet:prot1	40100 AV 40109 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_RAHumid			BACnet:prot1	40100 AV 40110 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SFPressAlm			BACnet:prot1	40100 BV 40111 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_LeakDetAlm			BACnet:prot1	40100 BV 40112 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_OutADStatus			BACnet:prot1	40100 BV 40113 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA1_SADStatus			BACnet:prot1	40100 BV 40114 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA1_HADPVal			BACnet:prot1	40100 AV 40115 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_Start			BACnet:prot1	40100 BV 40116 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_OpMode			BACnet:prot1	40100 AV 40117 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_GenAlm			BACnet:prot1	40100 BV 40118 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_SFStatus			BACnet:prot1	40100 BV 40119 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_Off			BACnet:prot1	40100 BV 40120 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_InAuto			BACnet:prot1	40100 BV 40121 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_SATemp			BACnet:prot1	40100 AV 40122 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_SAHumid			BACnet:prot1	40100 AV 40123 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_RATemp			BACnet:prot1	40100 AV 40124 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_RAHumid			BACnet:prot1	40100 AV 40125 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_SFPressAlm			BACnet:prot1	40100 BV 40126 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_LeakDetAlm			BACnet:prot1	40100 BV 40127 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_OutADStatus			BACnet:prot1	40100 BV 40128 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA2_SADStatus			BACnet:prot1	40100 BV 40129 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA2_HADPVal			BACnet:prot1	40100 AV 40130 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_Start			BACnet:prot1	40100 BV 40131 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_OpMode			BACnet:prot1	40100 AV 40132 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_GenAlm			BACnet:prot1	40100 BV 40133 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_SFStatus			BACnet:prot1	40100 BV 40134 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_Off			BACnet:prot1	40100 BV 40135 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_InAuto			BACnet:prot1	40100 BV 40136 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_SATemp			BACnet:prot1	40100 AV 40137 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_SAHumid			BACnet:prot1	40100 AV 40138 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_RATemp			BACnet:prot1	40100 AV 40139 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_RAHumid			BACnet:prot1	40100 AV 40140 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_SFPressAlm			BACnet:prot1	40100 BV 40141 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_LeakDetAlm			BACnet:prot1	40100 BV 40142 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_OutADStatus			BACnet:prot1	40100 BV 40143 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA3_SADStatus			BACnet:prot1	40100 BV 40144 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA3_HADPVal			BACnet:prot1	40100 AV 40145 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_Start			BACnet:prot1	40100 BV 40146 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_OpMode			BACnet:prot1	40100 AV 40147 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_GenAlm			BACnet:prot1	40100 BV 40148 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_SFStatus			BACnet:prot1	40100 BV 40149 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_Off			BACnet:prot1	40100 BV 40150 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_InAuto			BACnet:prot1	40100 BV 40151 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_SATemp			BACnet:prot1	40100 AV 40152 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_SAHumid			BACnet:prot1	40100 AV 40153 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_RATemp			BACnet:prot1	40100 AV 40154 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_RAHumid			BACnet:prot1	40100 AV 40155 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_SFPressAlm			BACnet:prot1	40100 BV 40156 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_LeakDetAlm			BACnet:prot1	40100 BV 40157 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_OutADStatus			BACnet:prot1	40100 BV 40158 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUA4_SADStatus			BACnet:prot1	40100 BV 40159 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUA4_HADPVal			BACnet:prot1	40100 AV 40160 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_Start			BACnet:prot1	40100 BV 40161 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_OpMode			BACnet:prot1	40100 AV 40162 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_GenAlm			BACnet:prot1	40100 BV 40163 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_SFStatus			BACnet:prot1	40100 BV 40164 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_Off			BACnet:prot1	40100 BV 40165 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_InAuto			BACnet:prot1	40100 BV 40166 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_SATemp			BACnet:prot1	40100 AV 40167 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_SAHumid			BACnet:prot1	40100 AV 40168 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_RATemp			BACnet:prot1	40100 AV 40169 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_RAHumid			BACnet:prot1	40100 AV 40170 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_SFPressAlm			BACnet:prot1	40100 BV 40171 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_LeakDetAlm			BACnet:prot1	40100 BV 40172 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_OutADStatus			BACnet:prot1	40100 BV 40173 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB1_SADStatus			BACnet:prot1	40100 BV 40174 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB1_HADPVal			BACnet:prot1	40100 AV 40175 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_Start			BACnet:prot1	40100 BV 40176 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_OpMode			BACnet:prot1	40100 AV 40177 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_GenAlm			BACnet:prot1	40100 BV 40178 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_SFStatus			BACnet:prot1	40100 BV 40179 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_Off			BACnet:prot1	40100 BV 40180 85 -1 0 false boolean			500	R/W	false	Variables	None

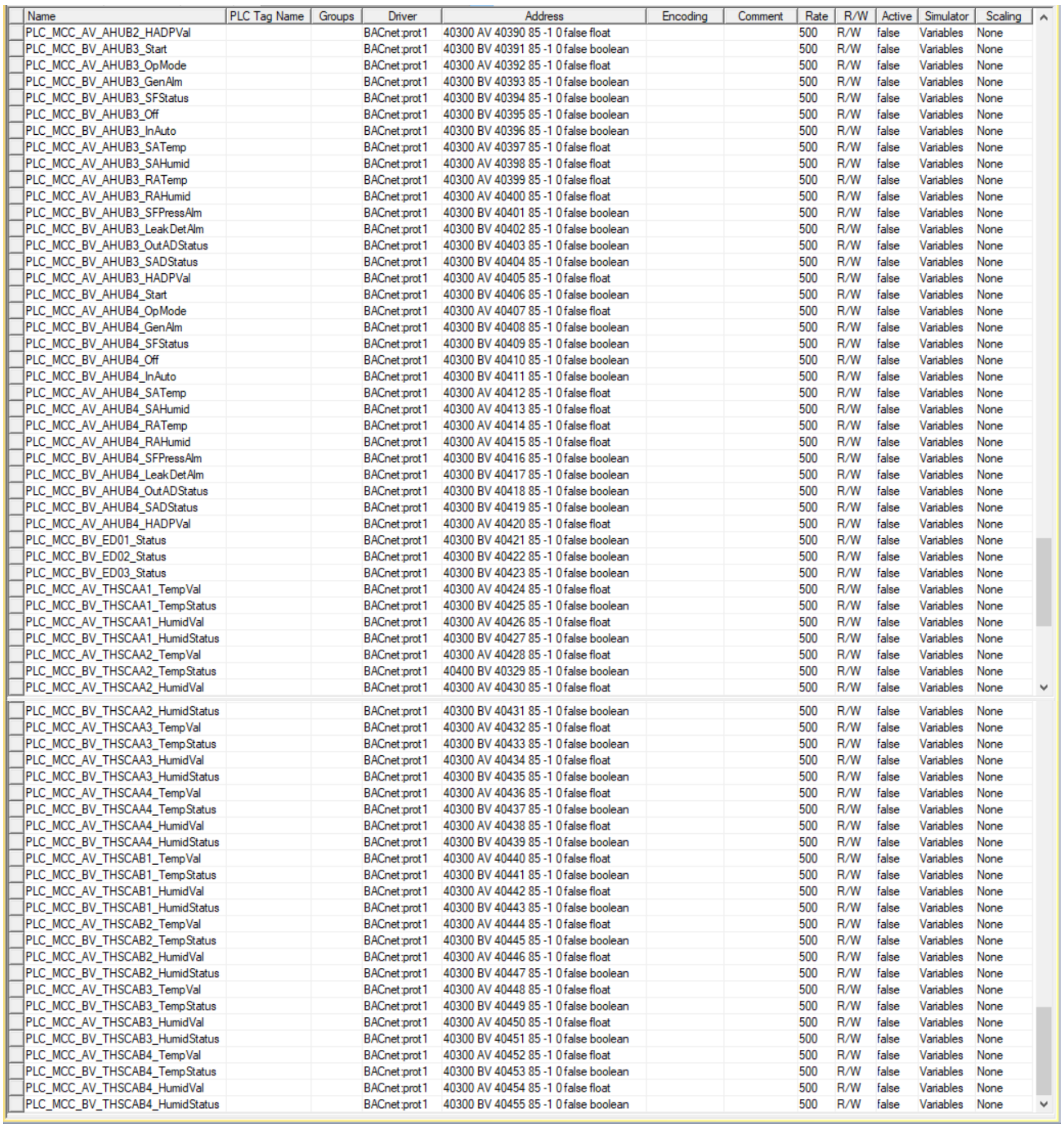



Name	PLC Tag Name	Groups	Driver	Address	Encoding	Comment	Rate	R/W	Active	Simulator	Scaling
PLC_AC_BV_AHUB2_InAuto			BACnet:prot1	40100 BV 40181 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_SATemp			BACnet:prot1	40100 AV 40182 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_SAHumid			BACnet:prot1	40100 AV 40183 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_RATemp			BACnet:prot1	40100 AV 40184 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_RAHumid			BACnet:prot1	40100 AV 40185 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_SFPressAlm			BACnet:prot1	40100 BV 40186 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_LeakDetAlm			BACnet:prot1	40100 BV 40187 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_OutADStatus			BACnet:prot1	40100 BV 40188 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB2_SADStatus			BACnet:prot1	40100 BV 40189 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB2_HADPVal			BACnet:prot1	40100 AV 40190 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_Start			BACnet:prot1	40100 BV 40191 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_OpMode			BACnet:prot1	40100 AV 40192 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_GenAlm			BACnet:prot1	40100 BV 40193 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_SFStatus			BACnet:prot1	40100 BV 40194 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_Off			BACnet:prot1	40100 BV 40195 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_InAuto			BACnet:prot1	40100 BV 40196 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_SATemp			BACnet:prot1	40100 AV 40197 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_SAHumid			BACnet:prot1	40100 AV 40198 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_RATemp			BACnet:prot1	40100 AV 40199 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_RAHumid			BACnet:prot1	40100 AV 40200 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_SFPressAlm			BACnet:prot1	40100 BV 40201 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_LeakDetAlm			BACnet:prot1	40100 BV 40202 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_OutADStatus			BACnet:prot1	40100 BV 40203 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB3_SADStatus			BACnet:prot1	40100 BV 40204 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB3_HADPVal			BACnet:prot1	40100 AV 40205 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_Start			BACnet:prot1	40100 BV 40206 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_OpMode			BACnet:prot1	40100 AV 40207 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_GenAlm			BACnet:prot1	40100 BV 40208 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_SFStatus			BACnet:prot1	40100 BV 40209 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_Off			BACnet:prot1	40100 BV 40210 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_InAuto			BACnet:prot1	40100 BV 40211 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_SATemp			BACnet:prot1	40100 AV 40212 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_SAHumid			BACnet:prot1	40100 AV 40213 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_RATemp			BACnet:prot1	40100 AV 40214 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_RAHumid			BACnet:prot1	40100 AV 40215 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_SFPressAlm			BACnet:prot1	40100 BV 40216 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_LeakDetAlm			BACnet:prot1	40100 BV 40217 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_OutADStatus			BACnet:prot1	40100 BV 40218 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_AHUB4_SADStatus			BACnet:prot1	40100 BV 40219 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_AHUB4_HADPVal			BACnet:prot1	40100 AV 40220 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_ED01_Status			BACnet:prot1	40100 BV 40221 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_ED02_Status			BACnet:prot1	40100 BV 40222 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_BV_ED03_Status			BACnet:prot1	40100 BV 40223 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA1_TempVal			BACnet:prot1	40100 AV 40224 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA1_TempStatus			BACnet:prot1	40100 BV 40225 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA1_HumidVal			BACnet:prot1	40100 AV 40226 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA1_HumidStatus			BACnet:prot1	40100 BV 40227 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA2_TempVal			BACnet:prot1	40100 AV 40228 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA2_TempStatus			BACnet:prot1	40100 BV 40229 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA2_HumidVal			BACnet:prot1	40100 AV 40230 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA2_HumidStatus			BACnet:prot1	40100 BV 40231 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA3_TempVal			BACnet:prot1	40100 AV 40232 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA3_TempStatus			BACnet:prot1	40100 BV 40233 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA3_HumidVal			BACnet:prot1	40100 AV 40234 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA3_HumidStatus			BACnet:prot1	40100 BV 40235 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA4_TempVal			BACnet:prot1	40100 AV 40236 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA4_TempStatus			BACnet:prot1	40100 BV 40237 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAA4_HumidVal			BACnet:prot1	40100 AV 40238 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAA4_HumidStatus			BACnet:prot1	40100 BV 40239 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB1_TempVal			BACnet:prot1	40100 AV 40240 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB1_TempStatus			BACnet:prot1	40100 BV 40241 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB1_HumidVal			BACnet:prot1	40100 AV 40242 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB1_HumidStatus			BACnet:prot1	40100 BV 40243 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB2_TempVal			BACnet:prot1	40100 AV 40244 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB2_TempStatus			BACnet:prot1	40100 BV 40245 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB2_HumidVal			BACnet:prot1	40100 AV 40246 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB2_HumidStatus			BACnet:prot1	40100 BV 40247 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB3_TempVal			BACnet:prot1	40100 AV 40248 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB3_TempStatus			BACnet:prot1	40100 BV 40249 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB3_HumidVal			BACnet:prot1	40100 AV 40250 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB3_HumidStatus			BACnet:prot1	40100 BV 40251 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB4_TempVal			BACnet:prot1	40100 AV 40252 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB4_TempStatus			BACnet:prot1	40100 BV 40253 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_AC_AV_THSCAB4_HumidVal			BACnet:prot1	40100 AV 40254 85 -1 0 false float			500	R/W	false	Variables	None
PLC_AC_BV_THSCAB4_HumidStatus			BACnet:prot1	40100 BV 40255 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_Start			BACnet:prot1	40300 BV 40301 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA1_OpMode			BACnet:prot1	40300 AV 40302 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_GenAlm			BACnet:prot1	40300 BV 40303 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_SFStatus			BACnet:prot1	40300 BV 40304 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_Off			BACnet:prot1	40300 BV 40305 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_InAuto			BACnet:prot1	40300 BV 40306 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA1_SATemp			BACnet:prot1	40300 AV 40307 85 -1 0 false float			500	R/W	false	Variables	None





Name	PLC Tag Name	Groups	Driver	Address	Encoding	Comment	Rate	R/W	Active	Simulator	Scaling
PLC_MCC_AV_AHUA1_SAHumid			BACnet:prot1	40300 AV 40308 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA1_RATemp			BACnet:prot1	40300 AV 40309 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA1_RAHumid			BACnet:prot1	40300 AV 40310 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_SFPressAlm			BACnet:prot1	40300 BV 40311 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_LeakDetAlm			BACnet:prot1	40300 BV 40312 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_OutADStatus			BACnet:prot1	40300 BV 40313 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA1_SADStatus			BACnet:prot1	40300 BV 40314 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA1_HADPVal			BACnet:prot1	40300 AV 40315 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_Start			BACnet:prot1	40300 BV 40316 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_OpMode			BACnet:prot1	40300 AV 40317 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_GenAlm			BACnet:prot1	40300 BV 40318 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_SFStatus			BACnet:prot1	40300 BV 40319 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_Off			BACnet:prot1	40300 BV 40320 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_InAuto			BACnet:prot1	40300 BV 40321 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_SATemp			BACnet:prot1	40300 AV 40322 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_SAHumid			BACnet:prot1	40300 AV 40323 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_RATemp			BACnet:prot1	40300 AV 40324 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_RAHumid			BACnet:prot1	40300 AV 40325 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_SFPressAlm			BACnet:prot1	40300 BV 40326 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_LeakDetAlm			BACnet:prot1	40300 BV 40327 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_OutADStatus			BACnet:prot1	40300 BV 40328 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA2_SADStatus			BACnet:prot1	40300 BV 40329 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA2_HADPVal			BACnet:prot1	40300 AV 40330 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_Start			BACnet:prot1	40300 BV 40331 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_OpMode			BACnet:prot1	40300 AV 40332 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_GenAlm			BACnet:prot1	40300 BV 40333 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_SFStatus			BACnet:prot1	40300 BV 40334 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_Off			BACnet:prot1	40300 BV 40335 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_InAuto			BACnet:prot1	40300 BV 40336 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_SATemp			BACnet:prot1	40300 AV 40337 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_SAHumid			BACnet:prot1	40300 AV 40338 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_RATemp			BACnet:prot1	40300 AV 40339 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_RAHumid			BACnet:prot1	40300 AV 40340 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_SFPressAlm			BACnet:prot1	40300 BV 40341 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_LeakDetAlm			BACnet:prot1	40300 BV 40342 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_OutADStatus			BACnet:prot1	40300 BV 40343 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA3_SADStatus			BACnet:prot1	40300 BV 40344 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA3_HADPVal			BACnet:prot1	40300 AV 40345 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_Start			BACnet:prot1	40300 BV 40346 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_OpMode			BACnet:prot1	40300 AV 40347 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_GenAlm			BACnet:prot1	40300 BV 40348 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_SFStatus			BACnet:prot1	40300 BV 40349 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_Off			BACnet:prot1	40300 BV 40350 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_InAuto			BACnet:prot1	40300 BV 40351 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_SATemp			BACnet:prot1	40300 AV 40352 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_SAHumid			BACnet:prot1	40300 AV 40353 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_RATemp			BACnet:prot1	40300 AV 40354 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_RAHumid			BACnet:prot1	40300 AV 40355 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_SFPressAlm			BACnet:prot1	40300 BV 40356 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_LeakDetAlm			BACnet:prot1	40300 BV 40357 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_OutADStatus			BACnet:prot1	40300 BV 40358 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUA4_SADStatus			BACnet:prot1	40300 BV 40359 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUA4_HADPVal			BACnet:prot1	40300 AV 40360 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_Start			BACnet:prot1	40300 BV 40361 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_OpMode			BACnet:prot1	40300 AV 40362 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_GenAlm			BACnet:prot1	40300 BV 40363 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_SFStatus			BACnet:prot1	40300 BV 40364 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_Off			BACnet:prot1	40300 BV 40365 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_InAuto			BACnet:prot1	40300 BV 40366 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_SATemp			BACnet:prot1	40300 AV 40367 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_SAHumid			BACnet:prot1	40300 AV 40368 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_RATemp			BACnet:prot1	40300 AV 40369 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_RAHumid			BACnet:prot1	40300 AV 40370 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_SFPressAlm			BACnet:prot1	40300 BV 40371 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_LeakDetAlm			BACnet:prot1	40300 BV 40372 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_OutADStatus			BACnet:prot1	40300 BV 40373 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB1_SADStatus			BACnet:prot1	40300 BV 40374 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB1_HADPVal			BACnet:prot1	40300 AV 40375 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_Start			BACnet:prot1	40300 BV 40376 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB2_OpMode			BACnet:prot1	40300 AV 40377 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_GenAlm			BACnet:prot1	40300 BV 40378 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_SFStatus			BACnet:prot1	40300 BV 40379 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_Off			BACnet:prot1	40300 BV 40380 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_InAuto			BACnet:prot1	40300 BV 40381 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB2_SATemp			BACnet:prot1	40300 AV 40382 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB2_SAHumid			BACnet:prot1	40300 AV 40383 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB2_RATemp			BACnet:prot1	40300 AV 40384 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB2_RAHumid			BACnet:prot1	40300 AV 40385 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_SFPressAlm			BACnet:prot1	40300 BV 40386 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_LeakDetAlm			BACnet:prot1	40300 BV 40387 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_OutADStatus			BACnet:prot1	40300 BV 40388 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB2_SADStatus			BACnet:prot1	40300 BV 40389 85 -1 0 false boolean			500	R/W	false	Variables	None



Name	PLC Tag Name	Groups	Driver	Address	Encoding	Comment	Rate	R/W	Active	Simulator	Scaling
PLC_MCC_AV_AHUB2_HADPVal			BACnet:prot1	40300 AV 40390 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_Start			BACnet:prot1	40300 BV 40391 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_OpMode			BACnet:prot1	40300 AV 40392 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_GenAlm			BACnet:prot1	40300 BV 40393 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_SFStatus			BACnet:prot1	40300 BV 40394 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_Off			BACnet:prot1	40300 BV 40395 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_InAuto			BACnet:prot1	40300 BV 40396 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_SATemp			BACnet:prot1	40300 AV 40397 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_SAHumid			BACnet:prot1	40300 AV 40398 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_RATemp			BACnet:prot1	40300 AV 40399 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_RAHumid			BACnet:prot1	40300 AV 40400 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_SFPressAlm			BACnet:prot1	40300 BV 40401 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_LeakDetAlm			BACnet:prot1	40300 BV 40402 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_OutADStatus			BACnet:prot1	40300 BV 40403 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB3_SADStatus			BACnet:prot1	40300 BV 40404 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB3_HADPVal			BACnet:prot1	40300 AV 40405 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_Start			BACnet:prot1	40300 BV 40406 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_OpMode			BACnet:prot1	40300 AV 40407 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_GenAlm			BACnet:prot1	40300 BV 40408 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_SFStatus			BACnet:prot1	40300 BV 40409 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_Off			BACnet:prot1	40300 BV 40410 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_InAuto			BACnet:prot1	40300 BV 40411 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_SATemp			BACnet:prot1	40300 AV 40412 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_SAHumid			BACnet:prot1	40300 AV 40413 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_RATemp			BACnet:prot1	40300 AV 40414 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_RAHumid			BACnet:prot1	40300 AV 40415 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_SFPressAlm			BACnet:prot1	40300 BV 40416 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_LeakDetAlm			BACnet:prot1	40300 BV 40417 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_OutADStatus			BACnet:prot1	40300 BV 40418 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_AHUB4_SADStatus			BACnet:prot1	40300 BV 40419 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_AHUB4_HADPVal			BACnet:prot1	40300 AV 40420 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_ED01_Status			BACnet:prot1	40300 BV 40421 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_ED02_Status			BACnet:prot1	40300 BV 40422 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_BV_ED03_Status			BACnet:prot1	40300 BV 40423 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA1_TempVal			BACnet:prot1	40300 AV 40424 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA1_TempStatus			BACnet:prot1	40300 BV 40425 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA1_HumidVal			BACnet:prot1	40300 AV 40426 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA1_HumidStatus			BACnet:prot1	40300 BV 40427 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA2_TempVal			BACnet:prot1	40300 AV 40428 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA2_TempStatus			BACnet:prot1	40400 BV 40329 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA2_HumidVal			BACnet:prot1	40300 AV 40430 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA2_HumidStatus			BACnet:prot1	40300 BV 40431 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA3_TempVal			BACnet:prot1	40300 AV 40432 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA3_TempStatus			BACnet:prot1	40300 BV 40433 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA3_HumidVal			BACnet:prot1	40300 AV 40434 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA3_HumidStatus			BACnet:prot1	40300 BV 40435 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA4_TempVal			BACnet:prot1	40300 AV 40436 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA4_TempStatus			BACnet:prot1	40300 BV 40437 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAA4_HumidVal			BACnet:prot1	40300 AV 40438 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAA4_HumidStatus			BACnet:prot1	40300 BV 40439 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB1_TempVal			BACnet:prot1	40300 AV 40440 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB1_TempStatus			BACnet:prot1	40300 BV 40441 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB1_HumidVal			BACnet:prot1	40300 AV 40442 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB1_HumidStatus			BACnet:prot1	40300 BV 40443 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB2_TempVal			BACnet:prot1	40300 AV 40444 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB2_TempStatus			BACnet:prot1	40300 BV 40445 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB2_HumidVal			BACnet:prot1	40300 AV 40446 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB2_HumidStatus			BACnet:prot1	40300 BV 40447 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB3_TempVal			BACnet:prot1	40300 AV 40448 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB3_TempStatus			BACnet:prot1	40300 BV 40449 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB3_HumidVal			BACnet:prot1	40300 AV 40450 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB3_HumidStatus			BACnet:prot1	40300 BV 40451 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB4_TempVal			BACnet:prot1	40300 AV 40452 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB4_TempStatus			BACnet:prot1	40300 BV 40453 85 -1 0 false boolean			500	R/W	false	Variables	None
PLC_MCC_AV_THSCAB4_HumidVal			BACnet:prot1	40300 AV 40454 85 -1 0 false float			500	R/W	false	Variables	None
PLC_MCC_BV_THSCAB4_HumidStatus			BACnet:prot1	40300 BV 40455 85 -1 0 false boolean			500	R/W	false	Variables	None

Appendix B: Simulation of collected data from field devices

```
(*DTB001_DH1_AHU-A1*)
AHUA1_Start := TRUE; (*START/STOP*)
AHUA1_OpMode := 0; (*Mode Of Operation*)
AHUA1_GenAlm := FALSE; (*General Alarm*)
AHUA1_SFStatus := TRUE; (*Supply Fan Status*)
AHUA1_Off := TRUE; (*OFF*)
AHUA1_InAuto := TRUE; (*In Auto mode*)
AHUA1_SATemp := 21.0; (*Supply Air Temperature*)
AHUA1_SAHumid := 39.0; (*Supply Air Humidity*)
AHUA1_RATemp := 22.0; (*Return Air Temperature*)
AHUA1_RAHumid := 35.0; (*Return Air Humidity*)
AHUA1_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUA1_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUA1_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUA1_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUA1_HADPVal := 6.6; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-A2*)
AHUA2_Start := TRUE; (*START/STOP*)
AHUA2_OpMode := 0; (*Mode Of Operation*)
AHUA2_GenAlm := FALSE; (*General Alarm*)
AHUA2_SFStatus := TRUE; (*Supply Fan Status*)
AHUA2_Off := TRUE; (*OFF*)
AHUA2_InAuto := TRUE; (*In Auto mode*)
AHUA2_SATemp := 21.0; (*Supply Air Temperature*)
AHUA2_SAHumid := 39.0; (*Supply Air Humidity*)
AHUA2_RATemp := 22.0; (*Return Air Temperature*)
AHUA2_RAHumid := 35.0; (*Return Air Humidity*)
AHUA2_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUA2_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUA2_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUA2_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUA2_HADPVal := 5.1; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-A3*)
AHUA3_Start := TRUE; (*START/STOP*)
AHUA3_OpMode := 0; (*Mode Of Operation*)
AHUA3_GenAlm := FALSE; (*General Alarm*)
AHUA3_SFStatus := TRUE; (*Supply Fan Status*)
AHUA3_Off := TRUE; (*OFF*)
AHUA3_InAuto := TRUE; (*In Auto mode*)
AHUA3_SATemp := 22.0; (*Supply Air Temperature*)
```

```
AHUA3_SAHumid := 39.0; (*Supply Air Humidity*)
AHUA3_RATemp := 27.0; (*Return Air Temperature*)
AHUA3_RAHumid := 35.0; (*Return Air Humidity*)
AHUA3_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUA3_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUA3_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUA3_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUA3_HADPVal := 5.6; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-A4*)
AHUA4_Start := TRUE; (*START/STOP*)
AHUA4_OpMode := 0; (*Mode Of Operation*)
AHUA4_GenAlm := FALSE; (*General Alarm*)
AHUA4_SFStatus := TRUE; (*Supply Fan Status*)
AHUA4_Off := TRUE; (*OFF*)
AHUA4_InAuto := TRUE; (*In Auto mode*)
AHUA4_SATemp := 22.0; (*Supply Air Temperature*)
AHUA4_SAHumid := 39.0; (*Supply Air Humidity*)
AHUA4_RATemp := 25.0; (*Return Air Temperature*)
AHUA4_RAHumid := 35.0; (*Return Air Humidity*)
AHUA4_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUA4_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUA4_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUA4_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUA4_HADPVal := 3.5; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-B1*)
AHUB1_Start := TRUE; (*START/STOP*)
AHUB1_OpMode := 0; (*Mode Of Operation*)
AHUB1_GenAlm := FALSE; (*General Alarm*)
AHUB1_SFStatus := TRUE; (*Supply Fan Status*)
AHUB1_Off := TRUE; (*OFF*)
AHUB1_InAuto := TRUE; (*In Auto mode*)
AHUB1_SATemp := 21.0; (*Supply Air Temperature*)
AHUB1_SAHumid := 39.0; (*Supply Air Humidity*)
AHUB1_RATemp := 25.0; (*Return Air Temperature*)
AHUB1_RAHumid := 35.0; (*Return Air Humidity*)
AHUB1_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUB1_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUB1_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUB1_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUB1_HADPVal := 2.3; (*Hot Aisle Differential pressure*)
```

```
(*DTB001_DH1_AHU-B2*)
AHUB2_Start := TRUE; (*START/STOP*)
AHUB2_OpMode := 0; (*Mode Of Operation*)
AHUB2_GenAlm := FALSE; (*General Alarm*)
AHUB2_SFStatus := TRUE; (*Supply Fan Status*)
AHUB2_Off := TRUE; (*OFF*)
AHUB2_InAuto := TRUE; (*In Auto mode*)
AHUB2_SATemp := 22.0; (*Supply Air Temperature*)
AHUB2_SAHumid := 39.0; (*Supply Air Humidity*)
AHUB2_RATemp := 27.0; (*Return Air Temperature*)
AHUB2_RAHumid := 35.0; (*Return Air Humidity*)
AHUB2_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUB2_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUB2_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUB2_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUB2_HADPVal := 6.3; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-B3*)
AHUB3_Start := TRUE; (*START/STOP*)
AHUB3_OpMode := 0; (*Mode Of Operation*)
AHUB3_GenAlm := FALSE; (*General Alarm*)
AHUB3_SFStatus := TRUE; (*Supply Fan Status*)
AHUB3_Off := TRUE; (*OFF*)
AHUB3_InAuto := TRUE; (*In Auto mode*)
AHUB3_SATemp := 21.0; (*Supply Air Temperature*)
AHUB3_SAHumid := 39.0; (*Supply Air Humidity*)
AHUB3_RATemp := 22.0; (*Return Air Temperature*)
AHUB3_RAHumid := 35.0; (*Return Air Humidity*)
AHUB3_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUB3_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUB3_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUB3_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUB3_HADPVal := 4.9; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_AHU-B4*)
AHUB4_Start := TRUE; (*START/STOP*)
AHUB4_OpMode := 0; (*Mode Of Operation*)
AHUB4_GenAlm := FALSE; (*General Alarm*)
AHUB4_SFStatus := TRUE; (*Supply Fan Status*)
AHUB4_Off := TRUE; (*OFF*)
AHUB4_InAuto := TRUE; (*In Auto mode*)
AHUB4_SATemp := 22.0; (*Supply Air Temperature*)
AHUB4_SAHumid := 39.0; (*Supply Air Humidity*)
```

```
AHUB4_RATemp := 24.0; (*Return Air Temperature*)
AHUB4_RAHumid := 35.0; (*Return Air Humidity*)
AHUB4_SFPressAlm := FALSE; (*Supply Fan Pressure Alarm*)
AHUB4_LeakDetAlm := FALSE; (*Leak Detection Alarm*)
AHUB4_OutADStatus := TRUE; (*Return Air Damper Status*)
AHUB4_SADStatus := TRUE; (*Supply Air Damper Status*)
AHUB4_HADPVal := 5.8; (*Hot Aisle Differential pressure*)

(*DTB001_DH1_ED-01*)
ED01_Status := FALSE; (*Status*)

(*DTB001_DH1_ED-02*)
ED02_Status := FALSE; (*Status*)

(*DTB001_DH1_ED-03*)
ED03_Status := FALSE; (*Status*)

(*DTB001_DH1_THSCAA1*)
THSCAA1_TempVal := 21.0; (*Temperature*)
THSCAA1_TempStatus := FALSE; (*Temperature Status*)
THSCAA1_HumidVal := 37.0; (*Humidity*)
THSCAA1_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAA2*)
THSCAA2_TempVal := 20.0; (*Temperature*)
THSCAA2_TempStatus := FALSE; (*Temperature Status*)
THSCAA2_HumidVal := 37.0; (*Humidity*)
THSCAA2_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAA3*)
THSCAA3_TempVal := 22.0; (*Temperature*)
THSCAA3_TempStatus := FALSE; (*Temperature Status*)
THSCAA3_HumidVal := 37.0; (*Humidity*)
THSCAA3_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAA4*)
THSCAA4_TempVal := 22.0; (*Temperature*)
THSCAA4_TempStatus := FALSE; (*Temperature Status*)
THSCAA4_HumidVal := 37.0; (*Humidity*)
THSCAA4_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAB1*)
THSCAB1_TempVal := 22.0; (*Temperature*)
```

```
THSCAB1_TempStatus := FALSE; (*Temperature Status*)
THSCAB1_HumidVal := 37.0; (*Humidity*)
THSCAB1_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAB2*)
THSCAB2_TempVal := 24.0; (*Temperature*)
THSCAB2_TempStatus := FALSE; (*Temperature Status*)
THSCAB2_HumidVal := 37.0; (*Humidity*)
THSCAB2_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAB3*)
THSCAB3_TempVal := 21.0; (*Temperature*)
THSCAB3_TempStatus := FALSE; (*Temperature Status*)
THSCAB3_HumidVal := 37.0; (*Humidity*)
THSCAB3_HumidStatus := FALSE; (*Humidity Status*)

(*DTB001_DH1_THSCAB4*)
THSCAB4_TempVal := 21.0; (*Temperature*)
THSCAB4_TempStatus := FALSE; (*Temperature Status*)
THSCAB4_HumidVal := 37.0; (*Humidity*)
THSCAB4_HumidStatus := FALSE; (*Humidity Status*)
```