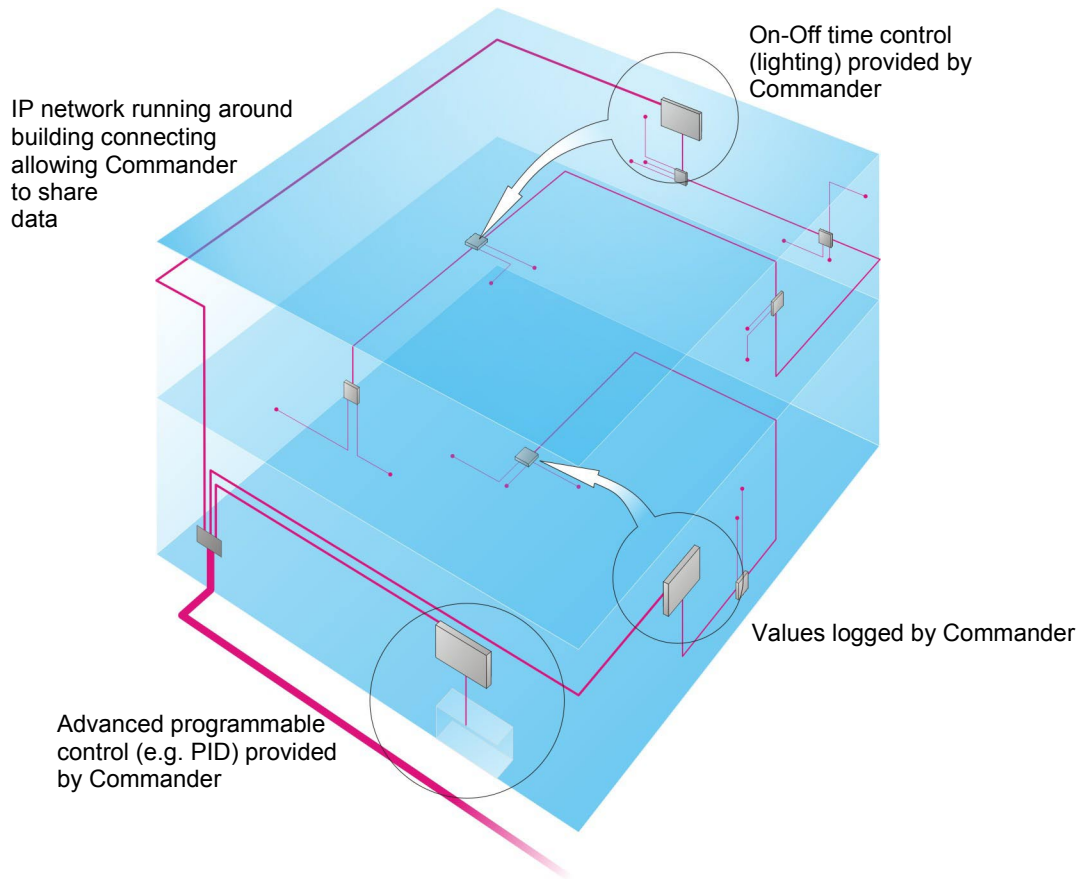


Introduction to Commander Control

Introduction

Commander is a powerful IP-based controller providing a configurable link between an Ethernet network and the connected system. This document describes the operation and configuration of the data acquisition and control based modules within Commander.



Contents

[Programmable Control](#) – configure processes to control objects within the attached systems

[Time Controllers](#) – provides on-off time control to other devices

[Data Logging](#) – collects and logs data for viewing

Further Information

For general engineering information about Commander and details of the interface, please refer to the Engineering Guide for the Commander interface installed.

For information relating to the Network connectivity of Commander including network configuration, IPBus, the user database, Web server, BACnet device, and authentication server, refer to the [Introduction to Commander Networking](#).

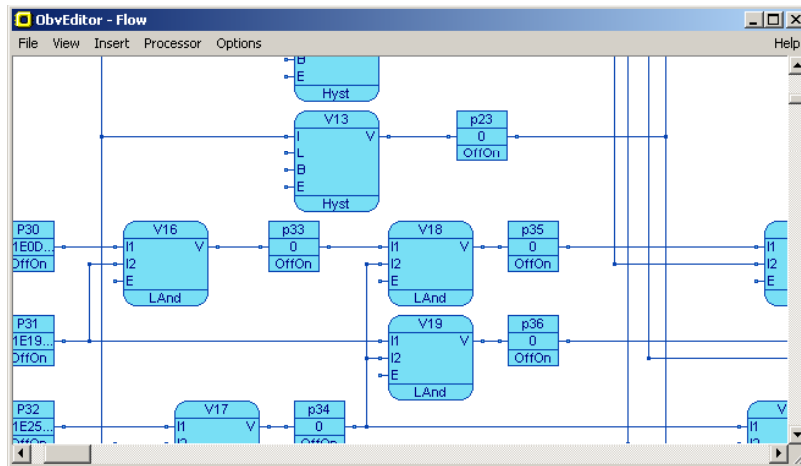
For information relating to the alarm event handling capabilities of Commander including alarm distribution and routing, event history, SNMP trap notification, e-mail notification, and the monitoring of values, refer to the [Introduction to Commander Alarms](#).

Programmable Control

The OBVEngine module allows the engineer to develop processes to control objects within Commander, the attached system, and on the Ethernet network.

OBVEngine is a configurable process controller that contains three ObVerse processors. Each processor runs ObVerse, which is the language used to engineer processes in the North product range. Refer to the [Introduction to ObVerse](#) document for information on the ObVerse language.

A process is programmed graphically using the OBVEditor application. Refer to the [OBVEditor v10 Window User Guide](#) document for further information.



Property Types Supported

OBVEngine supports the basic ObVerse property set:

| Property Type | Type of value | Other comments |
|---------------|--------------------------------|---|
| Obj\NoYes | 0 or 1 | Used to hold a No/Yes state: 0=No; 1=Yes |
| Obj\OffOn | 0 or 1 | Used to hold a Off/On state: 0=Off; 1=On |
| Obj\Num | -134217728 to +134217727 | Used to hold whole numbers, such as counts |
| Obj\Float | -9999999.9999 to +9999999.9999 | Used to hold floating-point numbers, such as temperatures, voltages |
| Obj\Text | ASCII character-based | Used to hold text values, such as labels, names |
| Obj\Obj | Remote object reference | Used to hold references of remote objects |
| Obj\ENum | 0...255 | Used to hold numeric codes that represent states |

Module Types Supported

OBVEngine supports the basic ObVerse module set:

| Module Type | Function | More Info |
|--------------|--|--------------------------------------|
| Obj\Add | Adds two numbers | Object Specification |
| Obj\Alarm | Creates a standards North Alarm event | Object Specification |
| Obj\Ave | Averages up to 8 values | Object Specification |
| Obj\Counter | Counts leading edges, and counts seconds | Object Specification |
| Obj\Delay | Delays a logical value (Bucket delay) | Object Specification |
| Obj\Div | Divides one number by another | Object Specification |
| Obj\Equal | Tests that one number is equal to another | Object Specification |
| Obj\Feedback | X amount of time for a value to reach a band | Object Specification |
| Obj\Gate | Gates one of two values | Object Specification |
| Obj\Gt | Test that one number is greater than another | Object Specification |
| Obj\Hyst | Performs hysteresis on a number | Object Specification |
| Obj\LAnd | Logical AND | Object Specification |
| Obj\Less | Test that one number is less than another | Object Specification |
| Obj\LInv | Logical inverter | Object Specification |
| Obj\LOR | Logical OR | Object Specification |
| Obj\LXor | Logical XOR | Object Specification |
| Obj\Madd | Adds up to 8 values | Object Specification |
| Obj\Mand | ANDs up to 8 values | Object Specification |
| Obj\Max | Finds the maximum values from up to 8 values | Object Specification |
| Obj\Min | Finds the minimum value from up to 8 values | Object Specification |
| Obj\Mod | Finds the modulus of a number | Object Specification |
| Obj\MOr | Logically ORs up to 8 values | Object Specification |
| Obj\Mult | Multiplies two numbers | Object Specification |

| Module Type | Function | More Info |
|--------------------|---|---|
| Obv\ObjRead | Reads an object that is remote from the process | <u>Object Specification</u> |
| Obv\ObjWrite | Writes an object that is remote from the process | <u>Object Specification</u> |
| Obv\OODelay | On/Off delay of a logical value | <u>Object Specification</u> |
| Obv\PID | Performs control using the PID algorithm | <u>Object Specification</u> |
| Obv\Profile | Creates two on-off time periods | <u>Object Specification</u> |
| Obv\Pulser | Creates a pulsing logical value | <u>Object Specification</u> |
| Obv\Random | Generates a random number | <u>Object Specification</u> |
| Obv\Rescale | Rescales and limits a number | <u>Object Specification</u> |
| Obv>Select | Gates one of eight values (not text) | <u>Object Specification</u> |
| Obv\Sqrt | Finds the square-root of a number | <u>Object Specification</u> |
| Obv\Sub | Subtracts one number from another | <u>Object Specification</u> |
| Obv\Sysinfo | Extracts information about the platform running the OBVEngine | <u>Object Specification</u> |

Maximum Objects

Each ObVerse processor can accept a total of 320 modules and properties.

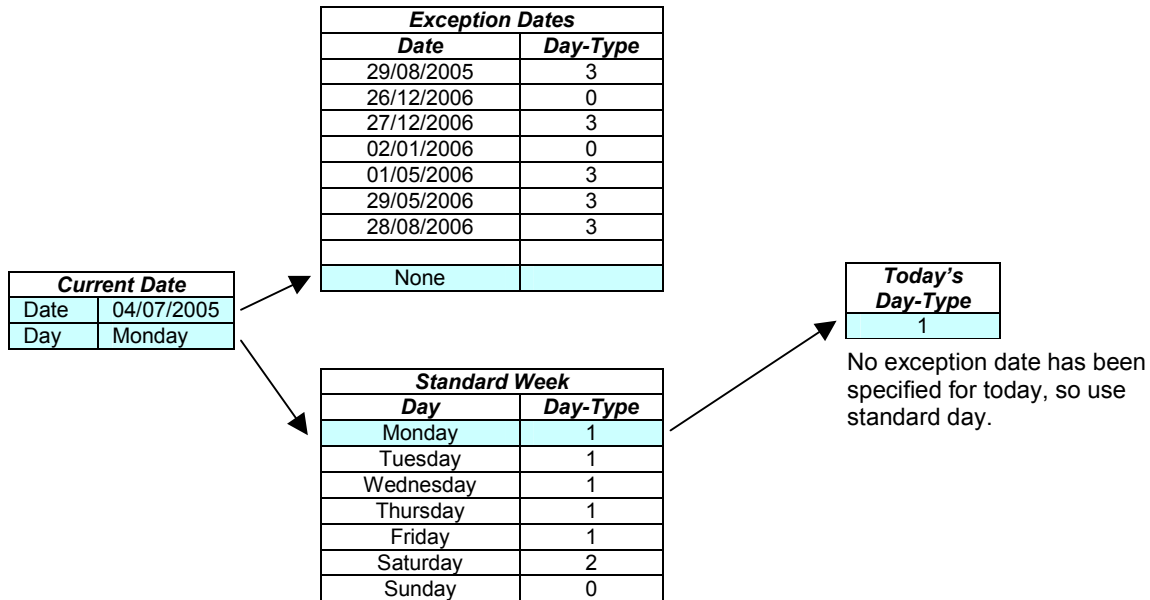
Time Controllers

The Calendar Timer module uses Commander's date and time to provide on-off time control to objects within Commander, the attached system, and on the Ethernet network.

Calendar

The calendar is used to specify a day-type for each day of a standard week. A list of exception dates is also available to define a different day-type for specific dates.

The calendar uses today's date to determine the current day-type. If today is an exception date, then the exception day-type is selected, otherwise the standard day-type for today is used.



Alternatively, the Calendar Timer module can obtain the current day-type from a remote source. This allows several Commanders to share a common calendar.

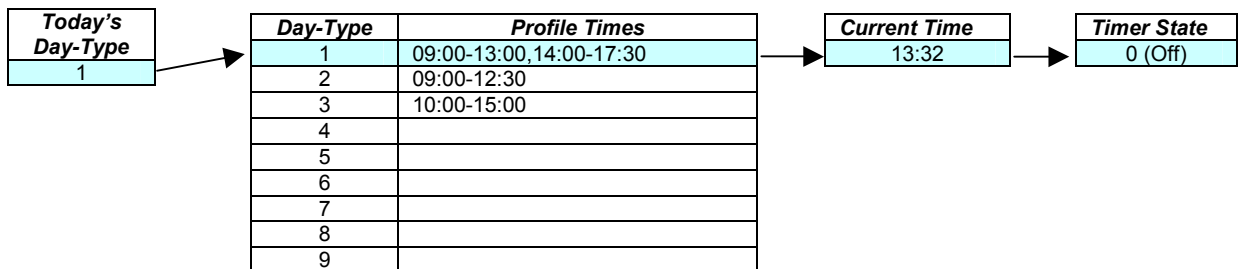
Day-types are a number in the range 0, and 1 to 9, where 0 refers to off. The particular use of day-types is left to the engineer.

Timer

The Calendar Timer module contains 10 timers. Each timer has the following parameters:

- Label – a brief label describing the timer
- Profile Times – up to 5 on-off times for each day-type
- State – the calculated on-off state of the timer
- Destination Object – the object reference to write the timer state

The timer uses today's day-type, from the calendar, to select a profile to use. Each timer then uses the current time to calculate its on-off state from the profile.



If required, the timer state can be written to a remote object elsewhere on the system.

Logging

Commander can collect and record historical data values from objects in the connected systems with the LogMax module.

The LogMax module logs up to 32 data channels and stores the data for use by other object-based software, such as the ObSys LogView or Data Manager applications.

The LogMax module can store approximately 15000 values, which is shared by the total number of data channels enabled in the LogMax Setup module. If two data channels are enabled then each log will be able to store approximately 7800 values, if 32 data channels are enabled then each log will be able to store approximately 380 values.

The logger optimises its storage by only recording a new value if it differs, by a specified amount, from the previously logged value. In this way, values that remain constant do not consume storage space unnecessarily.

Each of the data channels should be configured to collect data using the following parameters:

- Label – a brief label describing the log
- Enable – the data channel must be enabled for logging to occur
- Source Object – an object reference to read. The object may be from within the Commander, or accessible via the connected system or IPBus network. The object could also be a log
- Source Read Rate – the frequency in which the source object should be read
- Source Fail Count – the number of consecutive failed attempts at reading the source object (used for troubleshooting)
- Value – the current value of the log
- Precision – the amount the value should change before a new value is logged
- Destination Object – optional object reference to write the log data when the log is full

The data channels are rolling logs that when full will delete the first 25% of the data stored in them to make room for new values. Use the destination object to write data to larger logging store, such as ObSys Data Manager, before it is deleted.