CONTROL CONCEPTS

controlconcepts.net

Control Functionality Specification Module/Driver Development

The AV Technology Partner You Need.

Module Design Overview

AV systems have become very complicated and highly configurable. The control system programmer tends to be asked to do almost anything when it comes to system control. The problem comes in when the control system modules become bloated with functionality and extra control points that the system must deal with, even when those control points aren't needed in a particular project. It has become a more frequently used practice to break module design up into components, with each part handling very specific functionality. Because the components have very specific reduced functionality and only those items that need to be controlled will be included, this keeps the overhead and footprint at the minimum required from system to system and gives the control system programmer complete control of the features included in their project.

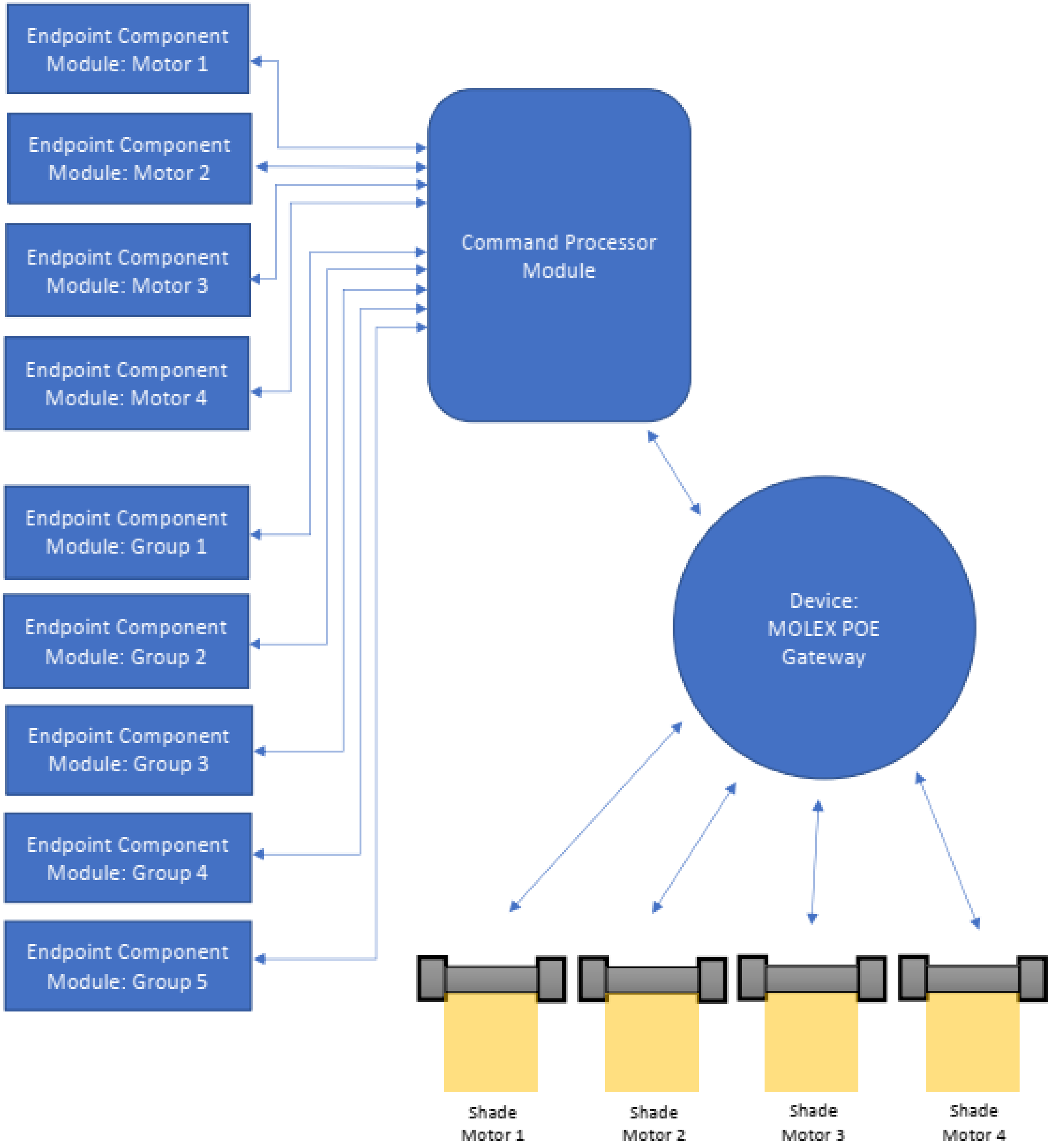
In addition, separating functionality by component also significantly enhances the ability to maintain and change the modules over time to support new features. New functionality (or entirely new components) can be added without affecting or breaking other components. This separation also makes it easier to track down and fix bugs and issues that may arise. Overall, this leads to a more stable and extensible product in the long run.

The design of the Somfy POE SIMPL# design enables all communication with a single POE gateway over an IP connection. The communication is managed through a command processor module. The command processor module specifies IP address and port information to connect to the Somfy gateway. An endpoint can consist of a single motor or group of motors. Each endpoint is managed by a component module. Each component module registers with the command processor module. The component module can specify a single motor ID, group ID, or wildcard (*) that identifies the motor or group of motors actuated by the command being sent. A gateway may control up to 4 motors, and motors may be gathered in up to 5 groups. Grouped motors act as a single unit, but the motors are given a priority within a group to determine which motor moves first. Grouping motors and setting motor priority for a specific gateway is managed through Somfy configuration software that is outside the scope of this module. Any desired components are mapped to the primary communication module by the Crestron programmer.

With this idea, these are the following components that will be developed as part of the scope of work.

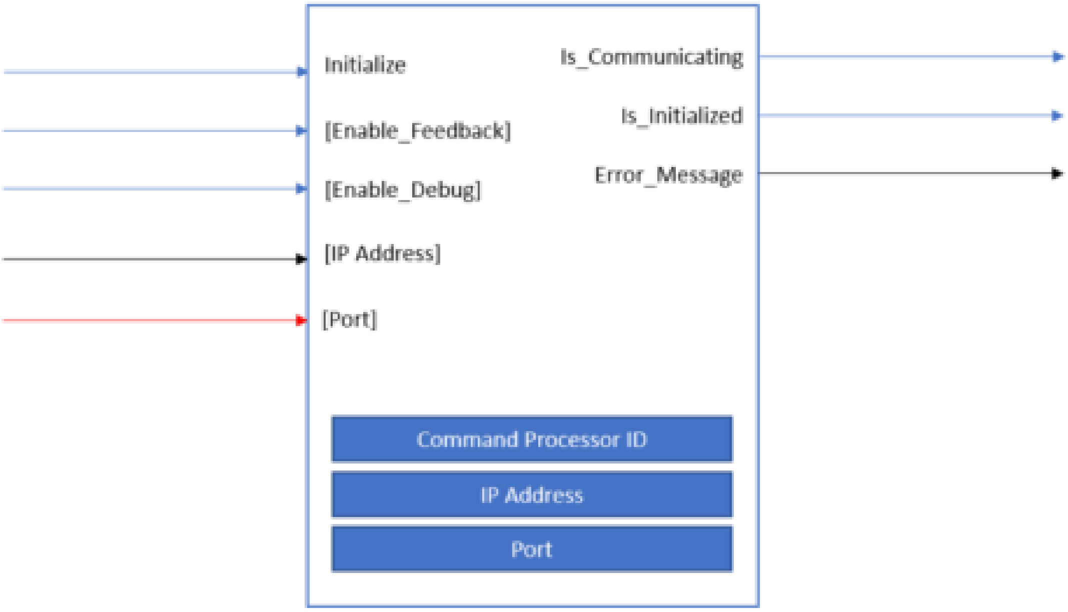
1. Primary command processor module
2. Endpoint component module

Example of how the components work within the Crestron module to talk to the Somfy POE device:


Endpoint Component Module: Motor 1
Endpoint Component Module: Motor 2
Endpoint Component Module: Motor 3
Endpoint Component Module: Motor 4
Endpoint Component Module: Group 1
Endpoint Component Module: Group 2
Endpoint Component Module: Group 3
Endpoint Component Module: Group 4
Endpoint Component Module: Group 5
Command Processor Module
Device: MOLEX POE Gateway
Shade Motor 1
Shade Motor 2
Shade Motor 3
Shade Motor 4

Primary Command Processor Module

The Primary Comm module (i.e. Command Processor) is responsible for establishing and maintaining communication with a specific POE gateway device and for passing messages between the device and any registered component modules. The protocol for communicating to the gateway will be TCP-IP.



Control

- **Initialize**: Manually re-establishes communication with the MOLEX POE device, if needed. The initialization process will be automated internally in the module on startup.
- **Enable_Feedback (optional)**: When set high, enables feedback events from the endpoint that provides shade position during a move command.
- **Enable_Debug (optional)**: When set high, the programmer sees relevant processes happening inside the SIMPL# module from within the SIMPL Debugger tool. Though normally off, this is a useful feature when debugging.
- **IP_Address (optional)**: This signal overrides the default IP address assignment specified in the symbol parameter list. The signal can be used to change the IP address during runtime of the program without requiring recompilation.
- **Port (optional)**: This signal overrides the default port assignment specified in the symbol parameter list. The signal can be used to change the communication port during runtime of the program without requiring recompilation.

Feedback

- **Is_Communicating**: When high, indicates the device is communicating with the program.
- **Is_Initialized**: When high, indicates that every component mapped to the comm module is ready

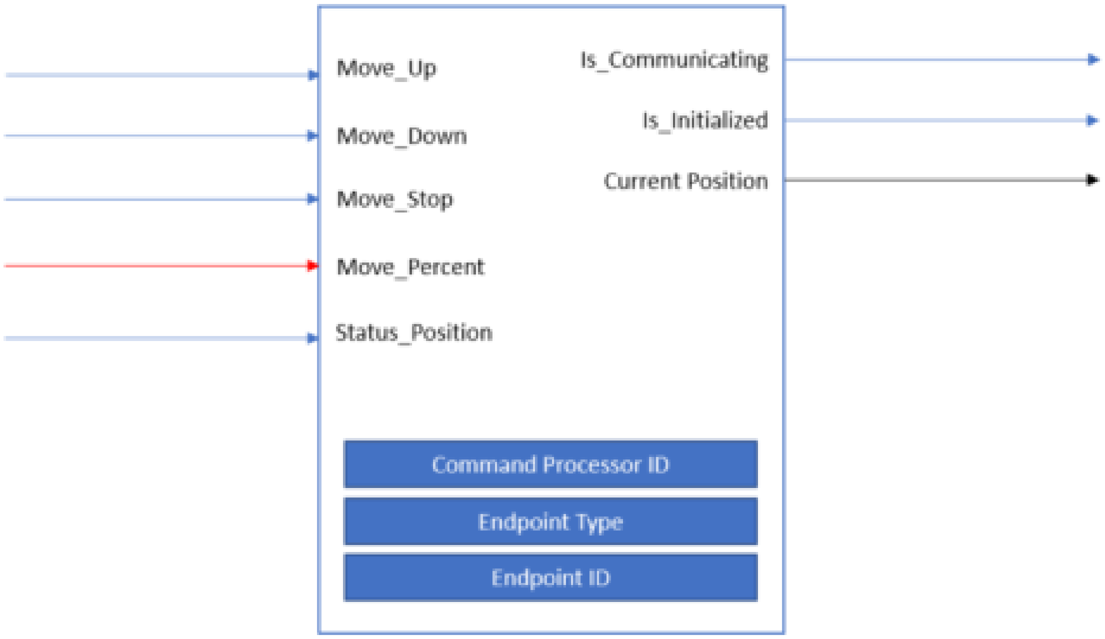
for control and that any relevant status information for the component is synced with the device.
- **Error_Message**: text indicating the code and description for any error responses from the device.

Parameters

- **Command_Processor_ID**: A dropdown list of ID's to select from. As each comm module is responsible for talking to a particular device, each will need its own unique ID.
- **IP_Address**: Text field for entry of the POE IP address. This is the default IP address. This value can be overridden by passing a new IP address through the IP_Address signal.
- **Port**: Text field for entry of the POE port. This is the default IP communication port This value can be overridden by passing a new IP address through the Port signal.

Endpoint Component

The Endpoint Component is responsible for directly controlling a single endpoint device (i.e. a motor connected to a particular channel of the MOLEX POE device) or a single group of devices. Multiple Endpoint components can be registered to each Comm module, one for each of the endpoints or groups the programmer would like to control.



Control

- **Move_Up**: Moves the shade motor up until it reaches the end limit or is interrupted with the stop command.
- **Move_Down**: Moves the shade motor down until it reaches the end limit or is interrupted with the stop command.
- **Move_Stop**: Halts movement of the shade motor.
- **Move_Percent**: Moves the shade to be closed (down) by the percentage specified.
- **Status_Position**: Queries the gateway for the current position of the endpoint. The value is returned in the **Current Position** output serial signal.

Feedback

- **Is_Communicating**: Every component in the command processor list will periodically ping each endpoint associated with the component. When high, this signal indicates that the endpoint is communicating/available.
- **Is_Initialized**: When high, this signal that the component is ready for control.
- **Current_Position:** Returns the current position of the motor as a percent (0-100).

Parameters

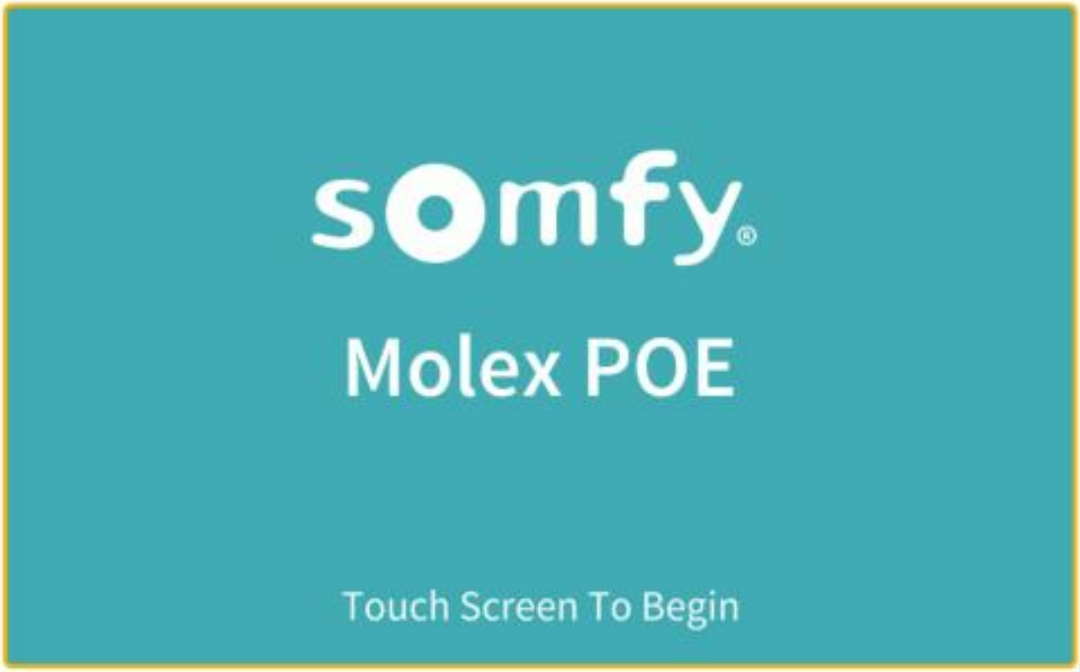
- **Command_Processor_ID**: The unique identifier for the module connected to a specific gateway. The value is a dropdown list of ID's to select from, and will need to correspond to the command processor the component should register itself to.
- **Endpoint_Type**: The specification of the endpoint type, either single motor, or group. The value is a dropdown list to select the type.
- **Endpoint_ID**: The unique identifier for the specific endpoint this module controls. The value is a text field for entry of the endpoint ID (or group ID if the endpoint type is Group).

Demo Program and Touchpanel

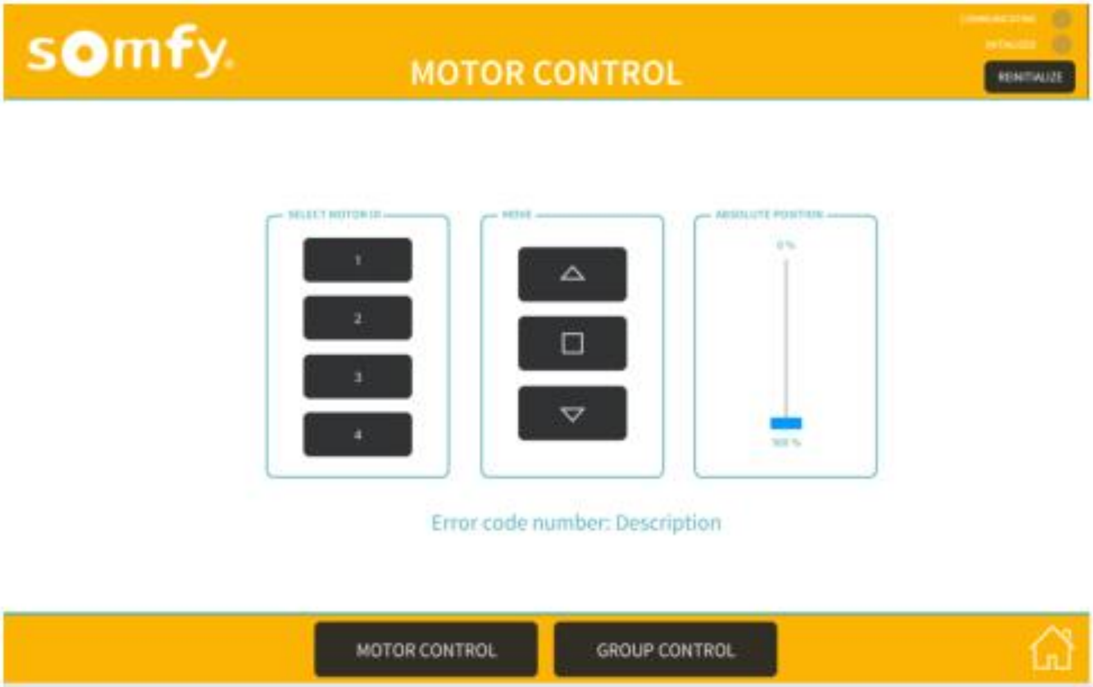
Per Crestron requirements and CCI standards, we will provide a demo program and UI whose purpose is to show the Crestron programmer how to use the modules. The demo program and UI is not intended to be a full AV system program and is instead intended to be as simple as possible and show the minimum programming required to use the modules. The Crestron programmers would take the concepts shown in the demo program and enhance/expand them to suit their own needs.

Below are samples of the intended demo UI.

Main Screen



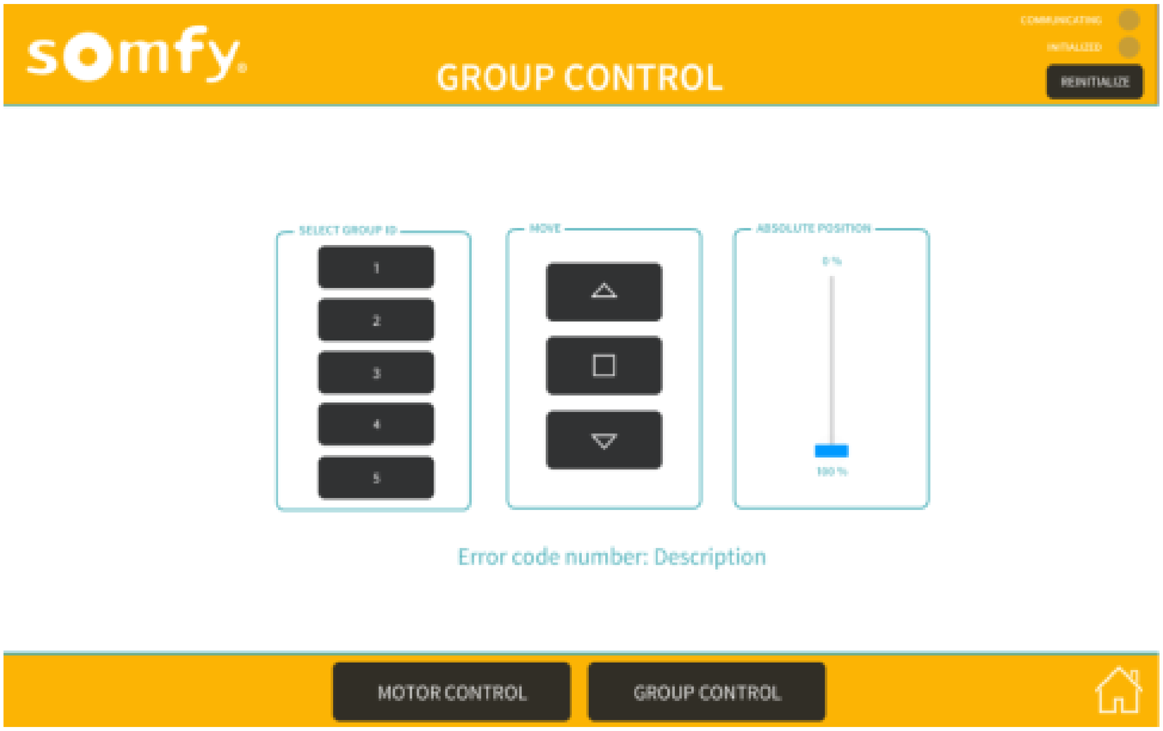
Motor Control (Single Motor)



Error code number: Description

**Note: Error text above is shown for reference and would normally be hidden, only showing when an error occurs.*

Group Control (Group of Motors)



Error code number: Description

**Note: Error text above is shown for reference and would normally be hidden, only showing when an error occurs.*

CONTROL CONCEPTS

controlconcepts.net

See our scalable, organized, proven approach to interface design for audio conferencing, video conferencing and presentation applications in action. Schedule a demo now.

Learn how Control Concepts can create a branded, flexible solution that takes your meeting applications to the next level.

Control Concepts, Inc.
controlconcepts.net
projects@controlconcepts.net
201.797.7900

The AV Technology Partner You Need.