ISA Draft 88.00.05

Batch Control
Part 5: Implementation Models & Terminology for Modular Equipment Control

Working Draft 06

July 2010

ISA-88 Draft on Batch production record

ISA

67 Alexander Drive

P. O. Box 12277

Research Triangle Park, NC 27709 USA

Preface

This document has been prepared as part of the service of ISA - The Instrumentation, Systems, and Automation Society, toward a goal of uniformity in the field of instrumentation. To be of real value, this document should not be static but should be subject to periodic review. Toward this end, the Society welcomes all comments and criticisms and asks that they be addressed to the Secretary, Standards and Practices Board; ISA; 67 Alexander Drive; P. O. Box 12277; Research Triangle Park, NC 27709; Telephone (919) 549-8411; Fax (919) 549-8288; E-mail: standards@isa.org.

The ISA Standards and Practices Department is aware of the growing need for attention to the metric system of units in general, and the International System of Units (SI) in particular, in the preparation of instrumentation standards. The Department is further aware of the benefits to USA users of ISA standards of incorporating suitable references to the SI (and the metric system) in their business and professional dealings with other countries. Toward this end, this Department will endeavor to introduce SI-acceptable metric units in all new and revised standards, recommended practices, and technical reports to the greatest extent possible. *Standard for Use of the International System of Units (SI): The Modern Metric System*, published by the American Society for Testing & Materials as IEEE/ASTM SI 10-97, and future revisions, will be the reference guide for definitions, symbols, abbreviations, and conversion factors.

It is the policy of ISA to encourage and welcome the participation of all concerned individuals and interests in the development of ISA standards, recommended practices, and technical reports. Participation in the ISA standards-making process by an individual in no way constitutes endorsement by the employer of that individual, of ISA, or of any of the standards, recommended practices, and technical reports that ISA develops.

CAUTION — ISA adheres to the policy of the American National Standards Institute with regard to patents. If ISA is informed of an existing patent that is required for use of the standard, it will require the owner of the patent to either grant a royalty-free license for use of the patent by users complying with the standard or a license on reasonable terms and conditions that are free from unfair discrimination.

Even if ISA is unaware of any patent covering this Standard, the user is cautioned that implementation of the standard may require use of techniques, processes, or materials covered by patent rights. ISA takes no position on the existence or validity of any patent rights that may be involved in implementing the standard. ISA is not responsible for identifying all patents that may require a license before implementation of the standard or for investigating the validity or scope of any patents brought to its attention. The user should carefully investigate relevant patents before using the standard for the user’s intended application.

However, ISA asks that anyone reviewing this standard who is aware of any patents that may impact implementation of the standard notify the ISA Standards and Practices Department of the patent and its owner.

Additionally, the use of this standard may involve hazardous materials, operations or equipment. The standard cannot anticipate all possible applications or address all possible safety issues associated with use in hazardous conditions. The user of this standard must exercise sound professional judgment concerning its use and applicability under the user’s particular circumstances. The user must also consider the applicability of any governmental regulatory limitations and established safety and health practices before implementing this standard.

The scope of this Part 4 standard is structured to follow the IEC guidelines. Therefore, the first three clauses discuss the *Scope* of the standard, *Normative References*, and *Definitions*, in that order.

Clause 4 is informative. The intent of this clause is to.

Clause 5 is normative. The intent of this clause is to.

Clause 6 is normative. The intention of this clause is to state the completeness, compliance and conformance requirements for this part of the standard.

Annex A is informative. It defines the

Annex B is informative. It presents a list of frequently asked questions and answers about this standard as it is currently envisioned.

This document is intended for people who are

- responsible for defining manufacturing equipment requirements
- Responsible for the automation of manufacturing equipment

CONTENTS

1 Scope 10

2 Normative references 11

3 Definitions 11

4 Modular equipment control concepts 13

4.1 Introduction 13
4.2 General concepts used in this part 14
4.3 Overlap of Part 1 and Part 5 15
4.4 Control Strategies in Manufacturing 18
4.4.1 Strategies in a manufacturing enterprise 18
4.4.2 Manufacturing control strategies 19
4.5 Equipment control 20
4.5.1 Types of control and levels 20
4.5.2 Equipment Coordination control 20
4.5.3 Equipment Procedural control 21
4.5.4 Equipment Basic control 21
4.6 Device 22
4.7 Control Module 22
4.7.1 Elemental Component 22
4.7.2 Compound Component 22
4.7.3 Coordination control 23
4.7.4 Contained and referenced control components 23
4.8 Modes 23
4.8.1 Control & operational modes 23

5 Automation Objects 26

Modular Equipment Control 26

5.1 Introduction 26
5.2 Automation Object 26
5.2.1 Functional Strategy 27
5.2.2 Function Manager 27
5.2.3 Resource Manager 27
5.2.4 Interfaces 28

1. Director Interfaces-- 28

2. Supervisory Managed Interfaces-- 28

3. Supervisory Non-Managed Interfaces--28

4. Internal Interfaces-- 28

5. External interfaces--- 28

6. Supervisory Interfaces--- 29

7. Subordinate Interfaces-- 29

5.3 Control Module--- 29

5.4 Equipment Module-- 29

5.5 Procedural Interface Module--- 29

5.5.1 State-- 30

5.5.2 Commands-- 31

5.5.3 Procedural Interface Model-- 31

5.6 Automation Module Implementation Examples------------------------------------33

5.6.1 System View-- 33

5.6.2 Multi Procedural Interface Ownership--------------------------------------33

5.6.3 Dynamic Binding---33

5.6.4 Resource Implementations--- 33

5.6.5 Module Encapsulation--33

6 Completeness--- 35

6.1 Compliance-- 35

6.2 Conformance-- 35

6.3 Extending the object model--35

Annex A Working Notes – Definitions--- 36

Annex B Questions and Answers-- 43

Annex C PID Example-- 44

C.1 PID Implementation--- 44

C.2 MEC PID Control Module--44

C.3 PID Control Element with Control Shell---45

C.4 MEC Analog Input Functional Strategy for PID Measured Variable Input---------47

C.5 MEC Virtual Input Functional Strategy for PID Manual Output and PID Setpoint Inputs-- 49

C.6 MEC Analog Output Functional Strategy for PID Output-----------------------------51

Annex D - Cascaded PID Implementation--- 54

6.4 MEC Cascaded PID Implementation---54

Annex E – Prototype Automation Object Example---55

1. Resource Manager (RM) Supervisory Interface------------------------------------- 55

2 Resource Manager (RM) Subordinate Interface-------------------------------------56

3 Resource Manager (RM) Administrative Interface----------------------------------- 58

6.4.2 Function Manager Incoming Commands---59

Control Request Inputs (CRI) Commands--- 59

4 Functional Manager (FM) Outgoing Command Outputs--------------------------- 61

5 MEC Status Overview-- 63

6 MEC Reset Command--- 64

7 MEC Control Shell Variable List--- 65

LIST OF FIGURES

Figure 1 – Part 1 Figure 4 Equipment entity model---16

Figure 2 – Part 1 Figure 24 - Control Activity Model--16

Figure 3 - Part 1 Process Control Activity Model-- 17

Figure 4 – Part 1 Figure 16 — Control recipe procedure referencing equipment procedural elements at the phase level -- 18

Figure 5 – Spectrum of control-- 23

Figure 6 - PID Function Block Example with Auto/Manual-------------------------------------24

Figure 7 - Automation Object-- 26

Figure 8 - Procedural Interface State and Command Model----------------------------------- 32

Figure 9 – Example organization of Automation Objects---------------------------------------33

Figure 10 - MEC PID Control Module-- 44

Figure 11 – PID basic controller in MEC format--- 45

Figure 12 - PID Control Element-- 46

Figure 13 – PID Functional Strategy-- 47

Figure 14 - MEC Analog Input-- 48

Figure 15 - Analog Input-- 49

Figure 16 – MEC Virtual Input --- 50

Figure 17 - Virtual Input-- 51

Figure 18 – MEC Analog Output-- 52

Figure 19 - Analog Output-- 53

List of Tables

Table 1 - Procedural Interface State and Command Matrix------------------------------------ 31

1) The formal decisions or agreements of the IEC on technical matters, prepared by technical committees on which all the National Committees having a special interest therein are represented, express, as nearly as possible, an international consensus of opinion on the subjects dealt with.

2) They have the form of recommendations for international use and they are accepted by the National Committees in that sense.

3) In order to promote international unification, the IEC expresses the wish that all National Committees should adopt the text of the IEC recommendation for their national rules in so far as national conditions will permit. Any divergence between the IEC recommendation and the corresponding national rules should, as far as possible, be clearly indicated in the latter.

4) The IEC has not laid down any procedure concerning marking as an indication of approval and has no responsibility when an item of equipment is declared to comply with one of its recommendations.

Introduction

ANSI-ISA-88.00.01-1995 entitled *Part 1: Models and Terminology* (referred to as Part 1 throughout this standard) provides abstract models and terminology applicable to batch control. Part 1 Clause 5 *Batch Control Concepts* defines types and hierarchies of control functionality and allocation, modes, and states of equipment and controls. Clauses 6.6 *Unit Supervision* and 6.7 *Process Control* describe the control activities that must be implemented at the equipment control level. Although the Part 1 standard provides some hierarchical models for equipment control, it does not provide sufficient guidance to enable consistent modular implementation of many key common performance requirements cited elsewhere in the document. It is assumed that the reader of this Part has a general knowledge of Part 1.

ANSI/ISA-88.00.02-2001 entitled *Part 2: Data Structures and Guidelines for Languages* (referred to as Part 2 throughout this standard) specifies visual and data representations of detailed recipe information, the content of which interacts with and determines the scope and reusability of equipment control element design.

ANSI/ISA-88.00.03-2003 entitled *Part 3: General and Site Recipe Models and Representation* (referred to as Part 3 throughout this standard) provides some insights on reusability impact, but in relation to recipe mappings.

ANSI/ISA-88.00.04-2006 entitled *Part 4*: *Batch Production Records* (referred to as Part 4 throughout this standard) defines a reference model for batch production records, the content of which may place certain design requirements on equipment control elements.

This Part 5 standard defines implementation models and terminology for modular equipment control in a consistent modular fashion based on the Part 1 equipment control concepts. The intent of this standard is to provide a clear hierarchical structure for defining and implementing the control found in Control Module and Equipment Module Entities.

This Part 5 will provide a consistent framework for a modular and distributed design of these functions with predictable interactions. This part describes an extensible core set of exposed interface functions and sample protocols. The models in this part cover encapsulation of basic control functionality, equipment procedural control at the phase level (as described in Part 1), and coordination control among different equipment entities and external systems.

This standard is consistent with the other parts of the ANSI/ISA88 standard, particularly building upon Part 1. Although this series of standards is intended primarily for batch processes, adoption of its models can provide value in other manufacturing domains.

1 Scope

This part defines a reference model for modular equipment control within equipment entities that address the control ascribed to equipment phases, equipment modules, both general and recipe aware, and control modules based on the equipment control concepts described in Part 1 of this standard. The reference model applies to equipment that may be used in batch processes and on other types of production and packaging processes. The reference model defines:

1. A template for definition of reusable modules that defines a standard terminology.
2. A template for command and control functionality within and among modules.
3. A method to describe and identify each module.
4. A method for exchanging module definitions among interested parties.

The reference model in this part may also apply to higher level equipment entities (process cells and units) but these are not explicitly covered.

Editor's Note: Need to evaluate the use of the automation object at the unit and cell level.

Chairman's Note: The "4. method for exchanging component definitions" may be removed from the scope if this turns out to take too much time for the initial release.

2 Normative references

The following normative documents contain provisions, which through reference in this text, constitute provisions of this part of this standard. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this part of this standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below.

— IEC 60902:1987, Industrial-process measurement and control: Terms and definitions

— IEC 61512:1997, Batch control- Part 1: Models and terminology

— IEC 62264-1:2000, Enterprise/control system integration – Part 1: Models and terminology

— ANSI/ISA-88.01-1995, Batch control- Part 1: Models and terminology

— ANSI/ISA-88.00.02-2001, Batch control - Part 2: Data Structures and Guidelines for Languages.

— ANSI/ISA-95.00.01-2000, Enterprise-Control System Integration – Part 1: Models and Terminology

— ANSI/ISA-95.00.02-2001, Enterprise-Control System Integration – Part 2: Object Model Attributes

— ANSI/ISA -104

— ISO/IEC 19501-1, Information technology – Unified Modeling Language (UML) – Part 1: Specification

— IEC 61131-3, Programmable Controllers – Part 3: Programming languages

— IEC 61499-1, Function Blocks – Part 1 Architecture

3 Definitions

For the purposes of this part of this draft, the following definitions apply. Definitions and concepts expressed in the Part 1 standard apply, except where differences are explicitly stated in this part.

3.1

control strategy

Device

- A Device is a piece of physical equipment ~~and~~ that has no control associated with it.

~~Part 1 refers to this as a Control Module. When control is associated with a device it becomes a Control Module Entity.~~

~~Note: …~~

4 Modular equipment control concepts

4.1 Introduction

This part defines an approach to develop control and coordination functions that can be used to combine function blocks as defined by IEC 61131-3, ISA104 and IEC 61499, in a way that supports dynamic coordination of multiple controlling entities, control strategies and operating schemes without requiring recoding or reengineering.

This part describes models and terminology used in automated equipment control for cells, units, equipment modules and control modules. This control is used to successfully carry out the intent of any procedural entity through the parameterization and management of the Equipment Phase(s) that are referenced by those Entities. The Equipment Control, procedural and coordination, which exists as part of the Equipment Phase is created and managed as an engineering activity and is part of the equipment and is only referenced by Recipe Entities[1] and are not contained by any recipe entity.

For clear understanding this part will differentiate between the recipe oriented procedural and coordination control that exists at the operation level and above and the equipment oriented control at the equipment phase layer and below.

The equipment hierarchy defined in Part 1 of this standard consists of groupings of process cells, units, equipment modules and control modules. Process cells can contain or reference units, equipment modules and control modules as shown in . Units can contain or reference equipment modules and control modules. Equipment modules can contain or reference other equipment modules and control modules, control modules can contain or reference other control modules. Control modules provide the only interface to the physical equipment. Part 1 is a conceptual and abstract model that can support manual operations as well as provide a guide to some possible automation concepts and approaches. This part also supports the concept of manual and automated solutions or combinations of both. As manufacturing disciplines other than batch address the need to create more efficient methods to automate, adapting the concepts put forward by this series of standards to the needs of these disciplines can provide similar benefits realized by the batch industries.

Part 1 of the S88 series provided guidance on how to separate the complex recipes required in batch control from the control of equipment. Part 1 provides details of “what to do” but leaves much of the “how to do” when delivering automated or hybrid equipment control systems up to the designers. This often leads to inconsistent solutions that are incompatible with other similar systems. The concepts, models and terminology in this part provide guidelines on the “how to do it” component of equipment control so that applications can be more consistent and allow easier integration. The concepts and approaches described in this part provide a consistent way to deliver the required functionality across a distributed control environment which may consist of automation from multiple suppliers using a variety of control equipment. While a full range of capability is presented it is up to the user to determine the right level of completeness for a specific application.

[1] This is defined in the 2007-2008 update to the ANSI/ISA88.01 standard, with an expected release in 2010.

4.2 General concepts used in this part

There are several general concepts that provide background and understanding for the approaches defined in this part. Many of these concepts are beyond the scope of the standard, but are identified here to place the models in context.

- Multiple strategies exist within an enterprise. The focus on this part is on manufacturing control strategies.
- The logic or automation program that implements *Equipment Control*[2] is contained in an automation object as farther defined in this part in Clause 5. The following functions are implemented by the automation object to perform the various types of control within an equipment entity:
    - functional strategy, this is the control function that exists to implement the required process action related to a specific equipment entity. All automation objects must have a functional strategy. A functional strategy may have internal and private methods to interact with other entities.
    - functional management, this is a function that can interact publicly with entities outside of an automation object. This function is optional for an automation object, if it exists the command, status and data requirements and capabilities of the associated functional strategy must be exposed such that any other entity can use them to interact with that functional strategy.
    - resource management, this function provides the automation object the ability to interact with multiple external entities in a secure and managed way. This function is optional for an automation object, if it exists the allowable interactions must be identified.
    -

    Note 1 There may be additional Control Components that implement other facets of control, such as recipe execution (recipe interpretation), equipment procedural elements above the equipment phase, and process cell coordination, but these are not defined in this part. However, these additional components could use the models defined in this part.

- An elemental module is the lowest level to which something can be reduced and still be functional. The scope of an elemental module is defined by the user and may vary between applications and between users.

  Example 1 a block valve with feedback, has the elements of the valve body, proximity sensors, and an actuator. Bringing these elements together create a compound of a block valve.

  Example 2 The block valve in Example 1 above is viewed as a single elemental module by operations personal.

- A compound module consists of two or more elemental modules that during normal operation will be coordinated and commanded by the supervisory modules that make up the hierarchy of the compound module.

[2] As defined in ANSI/ISA88.01 - *The equipment-specific functionality that provides the actual control capability for an equipment entity, including procedural, basic, and coordination control, and that is not part of the recipe.*

- A compound module my manage access to the elemental modules that make up that compound module in any of several methods.

  Example 1: All interaction with the elements of a compound module is managed at the highest level and no direct access to the elements is permitted.

  Example 2: At any level in the hierarchy of the compound module the normal supervisory modules commands and control can be suspended and command and control to the module(s) below that supervisory module can be provided by an entity that is external to the compound module.

Note 1 It is good practice to manage a module such that any external entity which can provide commands is validated before the command is allowed to pass to the functional strategy.

- Automation objects may be compound either through a containment structure (one component encapsulates another) or through a reference structure (one component references another component). Compound Control Components implement the hierarchy of control defined in Part 1.
- Equipment and control module entities may have the capability to react to multiple control modes. A control mode defines the operating scheme that is active and what functions can be performed on and by the equipment when it is in the corresponding mode.

Example 1 Sanitization ~~Control~~ Mode – The equipment entity is only able to execute the sanitization functional strategy.

Example 2 Production ~~Control~~ Mode – The equipment entity is only able to execute the production functional strategy.

Example 3 Maintenance ~~Control~~ Mode – All equipment procedural control (implementing recipe procedures) is suspended, all referenced components can be controlled by maintenance personnel, and interlocks may be manually disabled.

Note 2 The Part 1 term *Equipment Entity Mode* (auto, manual, and semi-auto) applies to an individual control component and is different from the Control Mode of the equipment entity. This refers to the functional strategy that is active in a module.

- State Models for all equipment entities: *Note: This may be removed after the major work is complete and we see if this is still needed.*

4.3 Overlap of Part 1 and Part 5

The following figures from Part 1 identify where the scope of this part fits.

illustrates the equipment entity hierarchy defined in Part 1. The scope of this part is shown as the highlighted area. The overlap with the Unit in Figure 1 is only in relation to an equipment phase when it is managed by the unit rather than an equipment module.

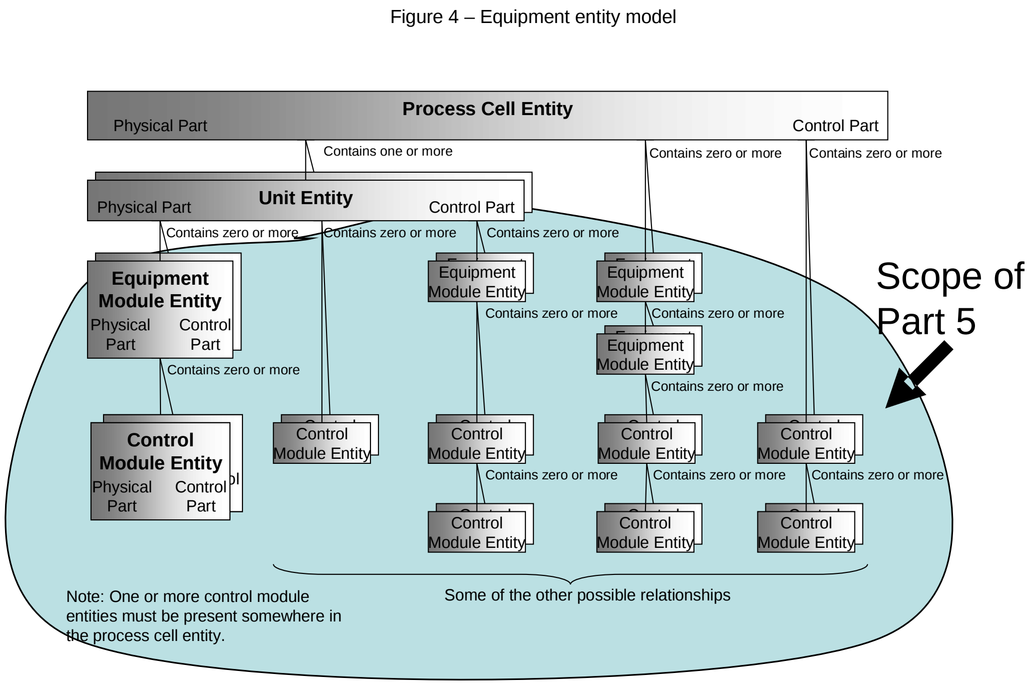

Figure 1 – Part 1 Figure 4 Equipment entity model

This part deals with process control as indicated in the Figure 2 – Part 1 Figure 24 - Control Activity Model.

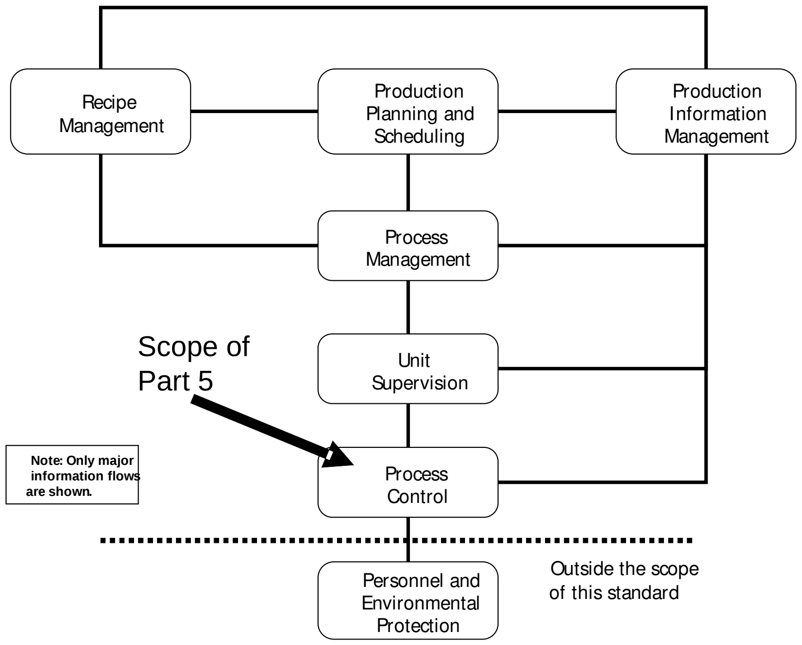
Recipe Management
Production Planning and Scheduling
Production Information Management
Process Management
Scope of Part 5
Unit Supervision
Note: Only major information flows are shown.
Process Control
Personnel and Environmental Protection
Outside the scope of this standard

Figure 2 – Part 1 Figure 24 - Control Activity Model

Figure 3 - Part 1 Process Control Activity Model illustrates the Part 1 process control activity model and where the scope of this part is shown as the highlighted area. This part only deals with procedural control from the equipment phase and down.

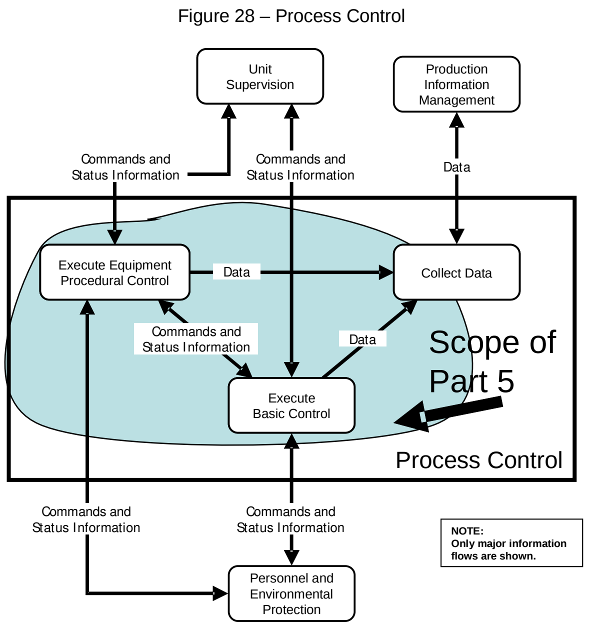
Figure 28 – Process Control
Unit Supervision
Production Information Management
Commands and Status Information
Commands and Status Information
Data
Execute Equipment Procedural Control
Data
Collect Data
Commands and Status Information
Data
Scope of Part 5
Execute Basic Control
Process Control
Commands and Status Information
Commands and Status Information
NOTE: Only major information flows are shown.
Personnel and Environmental Protection

Figure 3 - Part 1 Process Control Activity Model

The scope of this part in relation to the Part 1 procedure model is shown in Figure 4 – Part 1 Figure 16 — Control recipe procedure referencing equipment procedural elements at the phase level .

Figure 16 — Control recipe procedure referencing equipment procedural elements at the phase level

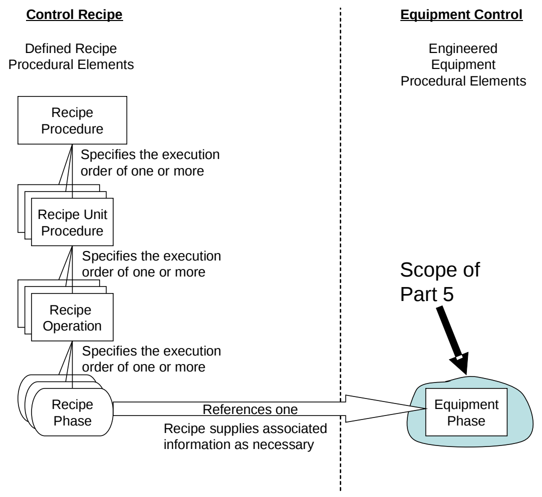
Control Recipe
Equipment Control
Defined Recipe Procedural Elements
Engineered Equipment Procedural Elements
Recipe Procedure
Specifies the execution order of one or more
Recipe Unit Procedure
Specifies the execution order of one or more
Recipe Operation
Specifies the execution order of one or more
Recipe Phase
References one
Recipe supplies associated information as necessary
Scope of Part 5
Equipment Phase

Figure 4 – Part 1 Figure 16 — Control recipe procedure referencing equipment procedural elements at the phase level

4.4 Control Strategies in Manufacturing

4.4.1 Strategies in a manufacturing enterprise

All enterprises have strategies to guide their efforts. Some common strategies are contrasted in the following examples.

Example Strategies may include:

Business Strategy – Develop and deliver products using optimized manufacturing to meet local market needs.

Product Strategy – Product shall be developed by the corporate central offices and shall be described using general recipe concepts with procedure and formula in the form of Procedure Function Charts (PFC) and all required manufacturing methods identified and the sequences needed to create the final product. Product recipes shall be provided to manufacturing sites which will adapt them to the local market and manufacturing needs and capabilities. Finished products will meet the supplied product specifications.

Manufacturing Strategy – Use batch and/or continuous systems based on production quantities required and locate within a maximum 500 kilometers of any customer requiring 100 Kiloton per year of product. Production rates greater than 1 Kiloton per day should consider continuous systems, less than 1 Kiloton should consider batch systems.

Manufacturing Operating Strategy – Minimize human labor in developed areas through automation of manufacturing. In undeveloped areas automate only those manufacturing operations that are required to maintain quality of the product. In all cases the operations will be conducted the same whether by automation or by human action.

Manufacturing Control Strategy – ANSI/ISA88 principles will be followed. Products specific procedures will be specified using a Master Recipe with PFC. The PFC will be used to create a Control Recipe that will be followed to direct the equipment to carry out the equipment operation necessary to deliver the expected result of the recipe.

This part focuses only on Manufacturing Control Strategies.

4.4.2 Manufacturing control strategies

There may be multiple manufacturing control strategies within an equipment entity. There is a Normal Control Strategy that is used when production is proceeding in the normal state. When the process enters an undesired state from which the normal control strategy can not recover, then other control strategies are required. These other strategies may enable humans or other automated systems to intercede and deal quickly and effectively with unplanned upsets or events.

Note Environmental and personal safety are outside the scope of the strategies motioned here and must be covered by the safety organizations. Product and Equipment safety is addressed.

A manufacturing control strategy uses the Control Modes of the Automation Objects to achieve its purpose.

4.4.2.1 Normal control strategy

There should be a normal control strategy that defines how the equipment is required to operate for normal production. This is the strategy developed by the equipment, process and automation designers to deal with all known control needs to make the required products. Mixing of human actions and automated actions to carry out the desired control strategy may be required.

The normal control strategy is the strategy that should be active the majority of the time.

4.4.2.2 Exception control strategy

There should be an exception control strategy that identifies predictable upsets and methods to recover from them. Mixing of human actions and automated actions to carry out the desired control strategy may be required.

4.4.2.3 Intervention control strategy

There should be an intervention control strategy that enables operations to suspend the normal strategy and defines how the equipment operates under the direction of production personnel.

Equipment and product protection shall be included with the Intervention Control Strategy.

This strategy is required when the equipment is not responding as expected by the normal or exception control strategy. The Intervention Control Strategy allows operations to direct the equipment in ways not anticipated by the normal control strategy. This enables operations to bring the equipment back into a state in which the normal strategy can resume control.

Note 1 This control strategy is often overlooked because of a focus on the normal control strategy.

Note 2 An often used alternate method of taking direct control of the actual equipment is not the best approach because it does not provide the equipment and product adequate protection and does not provide operations with appropriate overview and guidance that should be included in an Intervention Control Strategy.

Note 3 Company policies may determine which personnel have the authority and rights to start the Intervention Control Strategy.

4.4.2.4 Maintenance control strategy

There should be a maintenance control strategy that enables authorized personnel to suspend the other strategies for purposes of maintaining equipment. In the maintenance control strategy all equipment procedural control (implementing recipe procedures) is suspended, all referenced components can be controlled by maintenance personnel, and interlocks may be manually disabled.

The maintenance control strategy should allow direct personnel access to Control Components. In this strategy, equipment and product protection may be bypassed and it will be the responsibility of authorized personnel to ensure that equipment and product are protected as needed.

Note Company policies may determine which personnel have the authority and rights to start the Maintenance Control Strategy.

4.5 Equipment control

Equipment control described by this part is the logic or programming that implements the Process Control represented in Figure 3 - Part 1 Process Control Activity Model to cause the equipment to meet the intent of a Phase.

There are three types of automation objects that make up equipment control as follows:

1. The automation object that contains the Equipment Phase functionality that is recipe aware.
2. The automation object that contains the Equipment Phase functionality that is not recipe aware.
3. The automation object that contains the Equipment Basic Control functionality.

Equipment control can be very simple, consisting only of a single object performing basic control , to complex structures with many interactive automation objects using all types of control.

4.5.1 Types of control and levels

Part 1 of this standard describes procedural, basic and coordination as the three types of control required to manage a batch process. These types of control are implemented in various levels of the equipment. In addition different types of control are implemented at different hierarchy levels. This part does not address any of the control types defined by part 1 for recipes or the corresponding equipment control above the level of Equipment Phase.

4.5.2 Equipment Coordination control

Equipment Coordination control is the type of control in units, equipment modules and control modules that handles the:

1. coordination ~~of an~~ within automation- objects.
2. coordination between automation objects

   Example: (Tom to provide)

3. propagation of Mode of Action to contained and referenced automation- objects.
4. propagation of Modes of Control to contained and referenced automation objects.

   a. Selection of the appropriate operating scheme could be a result of this coordination control action.

5. arbitrating requests for usage.

4.5.3 Equipment Procedural control

The equipment procedural control addressed in this part is that which is described by Part 1 as an e~~E~~quipment p~~P~~hase or generic equipment module. It has quiescent state and requires an external command to become active. It implements a sequence of control steps to carry out an equipment procedure and when complete returns to its normally quiescent state.

Part 1 has a distinction between the recipe procedures and those that may reside in the equipment. The procedural model of Part 1 has as its lowest level the Phase which in this part is equivalent to the Equipment Phase. This equipment pahse is not part of the equipment operation procedure but may be referenced by either a recipe operation through the use of the recipe phase or referenced by an equipment operation. This part defines this type of Equipment Phase as one that is Recipe Aware.

There can be another type of equipment phase that the recipe is not aware of that also provides equipment procedural control. This type of control is generally resident in equipment modules and receives commands and instructions from other control objects.

This type of control is often implemented as a functional strategy as identified in clause #

~~Equipment Phase Control is a type of control component that handles the exposing of a phase interface and the interfacing of the phase commands and controls to other control components.~~

~~This control component is primarily coordination control and provides the interface to the Procedure Phase or Operation, depending on how these layers are implemented.~~

~~Note IEC 61131-3 documents methods of defining sequenced control, such as Sequential Function Charts (SFCs), this is commonly used to implement equipment phase control.~~

~~The state model for equipment phase control is the procedure state model defined in Part 1.~~

~~Equipment Sequenced Control is a type of control component that is responsible for the sequential set of equipment oriented actions required to implement equipment principal control.~~

~~Equipment Sequenced Control is sequential in nature and has quiescent state(s) which typically require an external command to become active or to move from state to state.~~

~~Note IEC 61131-3 documents methods of defining sequenced control, such as Sequential Function Charts (SFCs), this is commonly used to implement equipment sequenced control.~~

~~The state model for equipment sequenced control is function specific and is generally different from the procedure state model defined in Part 1.~~

4.5.4 Equipment Basic control

Equipment Basic control is the control that is dedicated to establishing and maintaining a specific state of equipment or process condition. It may include regulatory control, interlocking, monitoring, exception handling, and discrete or cyclic sequential control. Basic control is implemented in the Functional Strategy within a Control Module as identified in clause #.

Equipment Basic Control is persistent and always active and performs its function based on inputs from the equipment/process, and higher levels of control or humans.

4.6 ~~Device~~

~~The assumption that the final interface with equipment that is bounded by a Control Module has led to confusion because of the use of the term Device in other standards and industries to convey a similar concept to the Control Module Entity. To help reduce this area of confusion this part is including the concept of the physical Device.~~

~~Devices can be simple or complex. A simple device generally has a single function, such as a proximity sensor, or an actuator. A complex device is generally made up of several simple devices, such as a block valve that has and actuator, a valve body, and possibly a proximity sensor(s) to determine position of the valve.~~

~~A Device is a physical thing and contains no control. When basic control is applied to a device it is done through the actions of a Control Module Entity. A device is often constructed of several deices that together enable a processing activity.~~

~~Looking at a block valve with feedback, there are the elements of the valve body, proximity sensors, and an actuator. Bringing these elements together we create a compound of a block valve. The valve itself is a compound device made up of the three elemental devices. There is no control yet, so the block valve can't do anything.~~

4.7 Control Module

Control Modules must ultimately combine with a device. It may do so through other Control Modules, or be the first point of contract between the logical control realm and the physical world. There can be Control Modules that process information from a sensing device, such as a proximity sensor in a block valve, that will apply the control necessary to de-bounce the sensor.

4.7.1 ~~Elemental Component~~

~~An Elemental Component is the lowest level to which something can be reduced and still be functional. The scope of an Elemental Component is defined by the user and may vary between applications.~~

~~Example 1 a block valve with feedback, has the elements of the valve body, proximity sensors, and an actuator. Bringing these elements together create a compound of a block valve.~~

~~Example 2 The block valve in Example 1 is treated as a single Elemental component.~~

4.7.2 ~~Compound Component~~

~~A Compound Component consists of two or more Components that can be commanded by another entity from outside of the Compound Component.~~

~~Note 1 It is assumed that a Control Component can have one entity at a time that can provide commands, this is part of coordination control provided by the Mode of Control concepts.~~

~~Editor's note: Because this uses the word "strategy", there may be the need for explanation of how this relates to the control strategies defined above.~~

~~Figure 5 – Spectrum of control~~

4.7.3 ~~Coordination control~~

~~Need a lot here~~

4.7.4 ~~Contained and referenced control components~~

~~Control components may use the services of other control components to accomplish its functional strategy. The use of the services may be through a containment structure or a reference structure.~~

~~Example An equipment module may use the services of multiple control modules.~~

~~A containment structure is when a control component encapsulates one or more control components, and normal access to the encapsulated component is through the parent component.~~

~~A reference structure is when a control component does not encapsulate another control component but uses the services of other control components, and access to all of the control components is the same.~~

~~An implementation may use containment, reference, or a combination. The template defined in this Part makes no assumptions on the structure used and is equally valid for all choices.~~

~~Note An implementation of containment or referenced control components may be dependent on the functionality of the controller and programming environment.~~

4.8 Modes

4.8.1 Control & operational modes

Part 5 defines two classes of "mode" that can be applied to the Equipment Phase Control, Equipment Sequence Control and Equipment Basic Control.

4.8.1.1 Class 1: Mode of Action

The first class is how the internal activity of a single Control Component will operate and the algorithm's relation to the outputs it directly manages. This is the Mode of Action. Each Control Component will have at least one Mode of Action.

The internal activity of any control component will generally have at least the two classic modes described by S88 of Automatic and Manual. In the example of a PID function block, shown in Figure 6, when in a mode of Automatic the PID algorithm will be directly managing the Output. When the PID function block is in Manual the algorithm is no longer managing the Output which is now managed by something outside of the function block, sometimes an operator through some form of Human Machine Interface (HMI). Two examples of an HMI are digital values on a graphic display and a physical rheostat on a panel. Other Control Components might also be able to manipulate the output through this method.

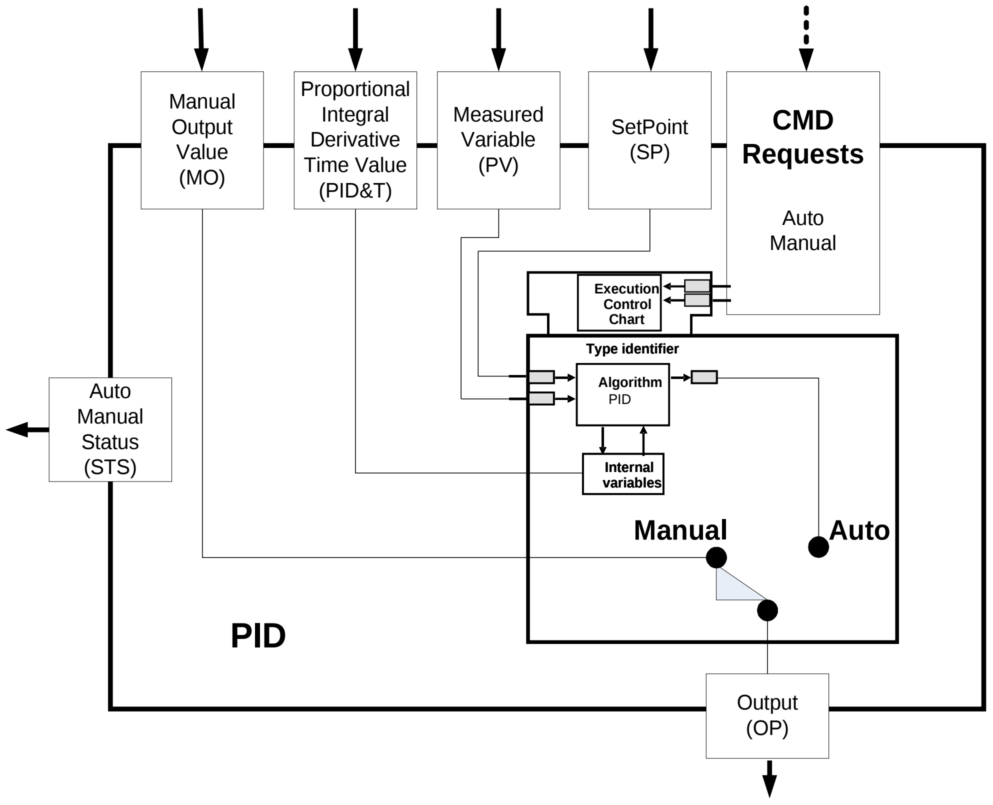

Figure 6 - PID Function Block Example with Auto/Manual

It is possible for a Control Component to have more than one Mode that describes its behavior, these modes can be complementary to one another and can be active at the same time. Besides Auto/Manual some other possible modes for Class 1 modes are: Simulation/Not-Simulation, Active/Inactive and any others that might be required. The Control Components themselves will generally manage this internal mode and might do so based upon the overall Mode of Control that this Control Component is part of.

If the Control Component is sufficiently complex and capable of executing complex sequences or procedures and will be manipulation the inputs of other Control Components being able to manage the internal execution of the sequence or procedure becomes important. In this situation the Mode of Action would follow the S88 Procedural Modes of Automatic, Semi-Automatic and Manual.

Note: need to develop a diagram of a “sequence” and “phase” control component

4.8.1.2 Class 2: Mode of Control

The second class is how multiple Control Components operate & interact together to carry out a more complex activity than is reasonable in a single control Component. This is the Mode of Control. Each grouping of Control Components will have a Mode of Control that is descriptive of what that grouping is designed to accomplish.

When two or more Control Components are grouped together to carry out a function it is often beneficial to allow these grouping to be represented by the concept of a Mode that describes what it is that grouping is designed to accomplish at any given time. Modes such as: Cascade

Control of PID loops, Operator Control for recovery from process upsets, and Maintenance Control for any unforeseen requirements. When considering larger groupings Modes such as Clean-Out, Sanitization, Production and others are often used.

Having a method to manage and coordinate the different possible modes for a Control Component grouping is a requirement and can be viewed as a separate Basic or Sequence Control Component that manages the group or groupings of other Control Components.

5 Automation Objects

Modular Equipment Control

5.1 Introduction

The component that, when combined with a physical module, creates an equipment entity per part 1 is defined as an automation object in this part. Automation objects provide a container for the functional strategy that implements the intent the control activity, basic, procedural or coordination. In practice an equipment module or control module will have at least one automation object associated with it to become an entity, either by human activity or automation. When automation exists in a control system an automation object as described in this part should be used.

5.2 Automation Object

The Automation object as shown in Figure 7 - Automation Object is made up of three main components. The Resource Manager, Functional Manager, Functional Strategy. The methods of interaction with these components are through the individual Status, Request, Command and Control Interfaces.

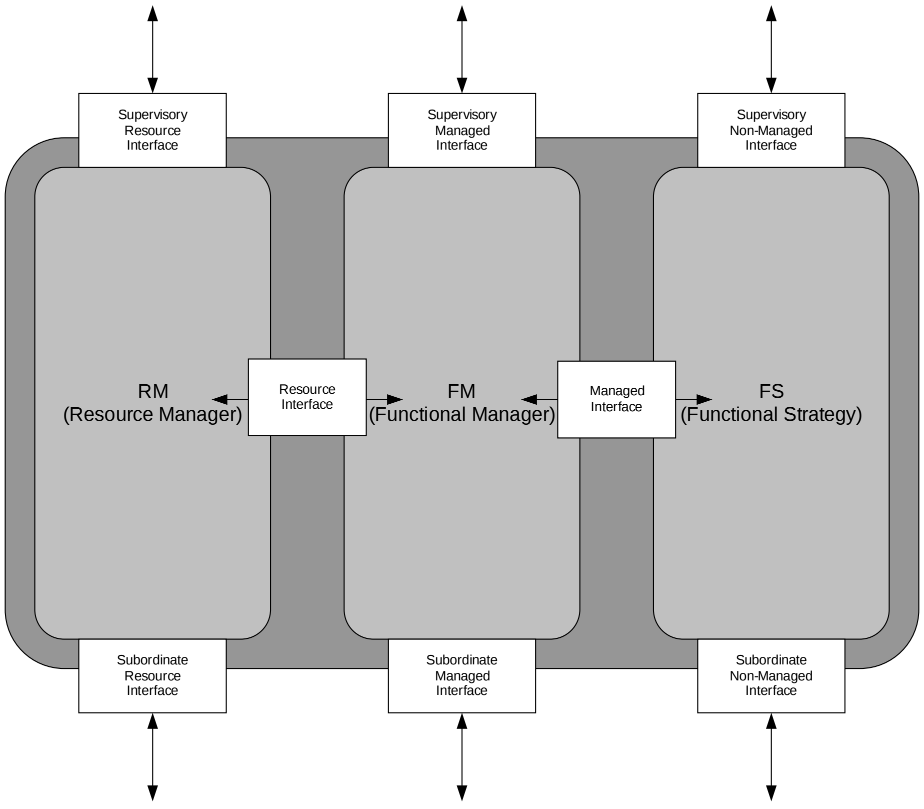
Supervisory Resource Interface
Supervisory Managed Interface
Supervisory Non-Managed Interface
RM (Resource Manager)
Resource Interface
FM (Functional Manager)
Managed Interface
FS (Functional Strategy)
Subordinate Resource Interface
Subordinate Managed Interface
Subordinate Non-Managed Interface

Figure 7 - Automation Object

5.2.1 Functional Strategy

Functional Strategy (FS) as shown in Figure 7 - Automation Object implements the core control of the Automation Module. In a Control Module this would be basic control. In an Equipment Module this would be procedural control that may or may not be recipe aware.

It is presumed that the Functional Strategy of an Automation Module can be represented as one or more state diagrams, no matter where it fits in the S88 continuum of equipment procedural, equipment coordination or equipment basic control. Some examples of devices that would be implemented in a Functional Strategy are Servos, Material Transfers, Bipolar Devices, Two-way Valves, Motors, Pumps, etc. The Functional Strategy implements the control that is necessary to make these devices work. The Functional Strategy is not limited to physical devices. The Functional Strategy can define a process or complete machine.

The Functional Strategy Status reflects all necessary aspects of the internal workings of the Functional Strategy and are available for view by any component. Some status attributes for the Functional Strategy can be found in the example appendix.

Then Functional Strategy also has command and control attributes that are used to command and control subordinate Automation Modules. The command attributes are used to execute actions in the subordinate Automation Modules. The control attributes are used by the subordinate Automation Modules to perform Procedural, Coordination and Basic Control. Some status attributes for the Functional Strategy can be found in the example appendix..

There are many different organizations (i.e. OpenPLC, OMAC, IEC 61131) that have already defined Functional Strategies for devices, processes and machines that are in use in manufacturing. This part will not define functional strategies.

5.2.2 Function Manager

The Function Manager is designed to accept command and control attributes from a supervisory source (another AM, an operator through an HMI, etc.). The supervisory source is determined by the Resource Manager. Only the supervisory source specified by the Resource Manager will be allowed access to the functional strategy by the Function Manager.

The Function Manager will accept command attributes which cause an action to be performed in the Functional Strategy. The control attributes are typically used by the Functional Strategy to maintain Control. Examples of control attributes are interlocks and permissives that may be passed to the Functional Strategy by the Function Manager. Some command and control attributes for the Function Manager can be found in the example appendix.

Example: When a maintenance person is the supervisory source the function manager will allow the interlocks to be bypassed to the functional strategy. In all other cases interlocks are passed unaltered to the FS.

The Function Manager also includes status attributes that provide status of commands and control signals. The Function Manager is also in charge of determining the status of the Functional Strategy and determining whether the Automation Module is in a safe state to allow the directing module to change in the Resource Manager.

5.2.3 Resource Manager

The Resource Manager is designed to regulate who is able to command the FS. Every Automation Module will have a Director that is the owner of the Automation Object. The director may delegate authority to a different supervisory source to interact with the FM & FS. Also it could have one or more subordinate Automation Modules that it directs. If there is more than one subordinate Automation Module, there will be logic which will combine the multiple attributes of the subordinate Automation Modules into one attribute for the directing Automation Module. The Resource Manager’s job is to take requests from other supervisory sources and change the directing resource of its Automation Module. It will validate whether it is OK to change the directing resource and determine whether to direct or release subordinate Automation Modules. The interface for the Resource Manager will be divided into two different sections, the Director Interface and the Subordinate Interface. The Director Interface deals with commands and status values that get sent to and from the supervisory sources. The Subordinate Interface deals with commands and status values that get sent to and from this module’s Subordinate Automation Modules.

5.2.4 Interfaces

The interfaces represent the communication of information that the Automation Module requires in order to operate properly. These interfaces include the internal interfaces for communication between the Functional Strategy and Functional Manager and the communication between the Functional Manager and the Resource Manager as well as the external interfaces between other supervisory sources. The information typically falls into one of three groups; Director Interface, Supervisory Managed Interface, and Supervisory Non-Managed Interface.

1. Director Interfaces

Resource information is the type that deals with Automation Module ownership, command, and status. This includes dealing with interfacing to supervisory and subordinate controls for resource management purposes.

2. Supervisory Managed Interfaces

Managed information is the type of information that is supplied to the Functional Strategy. This information is filtered and manipulated based on the module’s Resource Information. This includes dealing with interfacing to supervisory and subordinate controls for information sharing purposes.

3. Supervisory Non-Managed Interfaces

Non-Managed information is the type of information that the Functional Strategy always deal with directly is not subject to resource management concerns.

4. Internal Interfaces

The internal interfaces are used to represent the information that flows between components of the Automation Module.

5. External interfaces

External interfaces are the methods used to communicate with supervisory and subordinate equipment control.

6. Supervisory Interfaces

The Supervisory Interface represents the communication that occurs between the Automation Module and equipment control that exists at a higher level in the equipment entity hierarchy. This interface consists of three different groups of information that needs comminucating; Resource Commands and Control, Managed , Managed

7. Subordinate Interfaces

Discuss how the Supervisory Interfaces are used to connect a AO to subordinate controls.

5.3 Control Module

Equipment control normally found at this level directly manipulates actuators and other control modules. A control module can direct commands to actuators if they have been configured as part of the control module. A control module can also direct commands to other control modules if they are contained or in some way referenced by that control module. Control of the process is affected through the equipment specific manipulation of control modules and actuators.

Examples of equipment control in control modules include

— opening or closing a valve, with confirmation failure alarms;

— regulating the position of a control valve based on a sensor reading and PID control algorithm;

setting and maintaining the state of several valves in a material header. This could be a single control module configured to contain all of the valves directly or a compound control module configured to contain several control modules, each of which is configured to directly contain a single valve.

5.4 Equipment Module

The primary purpose of equipment control in an equipment module is to coordinate the functions of other equipment modules and lower level control modules.

Two types of equipment modules are identified: recipe-aware and non-recipe-aware.   A recipe-aware equipment module is distinguished from a non-recipe-aware equipment module by having an equipment phase with a recipe interface. Non-recipe-aware equipment modules do not have such an interface. This part of the standard is only concerned with the recipe-aware equipment module. In this part of the standard, when the term “equipment module” is used, it refers to the recipe-aware equipment module.

5.5 Procedural Interface Module

The primary purpose of the procedural interface in an equipment module is to provide a recipe interface. When an equipment entity, be it an Equipment Module or Unit, requires a recipe phase interface, the Procedural Interface Module is used to provide that functionality. The PIM is primarily made up of coordination control but may also contain procedural control. In addition to providing the interface for the recipe phase, it also coordinates the functions of other equipment modules and lower level control modules.

5.5.1 State

IDLE: The procedural element is waiting for a START command that will cause a transition to the RUNNING state.

STARTING:

RUNNING: Normal operation

COMPLETING:

COMPLETE: Normal operation has run to completion. The procedural element is now waiting for a RESET command that will cause a transition to ENABLING.

HOLDING: The procedural element has received a HOLD command and is executing its HOLDING logic to put the procedural element or equipment entity into a known state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the HELD state. HOLD commands are usually generated as a result of external events that require a temporary stop.

HELD: The procedural element has completed its HOLDING logic and has been brought to a known or planned state. This state is usually used for a short-term stop. The procedural element or equipment entity is waiting for a further command to proceed.

UN-HODING: The procedural element has received a UN-HOLD command while in the HELD state. It is executing its restart logic in order to return to the RUNNING state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the RUNNING state. UN-HOLD commands are usually generated as a result of external events that require a returning to a running state after a temporary stop.

SUSPENDING: The procedural element has received a SUSPEND command and is executing its SUSPENDING logic to put the procedural element or equipment entity into a known state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the SUSPENDED state. SUSPEND commands are usually generated as a result of internal events that require a temporary stop.

SUSPENDED: The procedural element has completed its SUSPENDING logic and has been brought to a known or planned state. This state is usually used for a short term stop. The procedural element or equipment entity is waiting for a further command to proceed.

UN-SUSPENDING: The procedural element has received a UN-SUSPEND command while in the SUSPENDED state. It is executing its restart logic in order to return to the RUNNING state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the RUNNING state. UN-SUSPEND commands are usually generated as a result of internal events that require a returning to a running state after a temporary stop.

STOPPING: The procedural element has received a STOP command and is executing its STOPPING logic, which facilitates a controlled normal stop. This might be used by an operator to end a phase when a normal operation such as a titration has run to satisfactory completion but the final target condition specified in the running equipment phase has not yet been totally reached. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the STOPPED state.

STOPPED: The procedural element has completed its STOPPING logic. The procedural element or equipment entity is waiting for a RESET command to transition to IDLE.

RESETTING:

ABORTING: The procedural element has received an ABORT command and is executing its ABORT logic, which is the logic that facilitates a quicker, but not necessarily controlled, abnormal stop. If no sequencing is required, then the procedural element transitions immediately to the ABORTED state.

ABORTED: The procedural element has completed its ABORTING logic. The procedural element is waiting for a RESET command to transition to IDLE.

CLEARING:

5.5.2 Commands

RESET:

START: This command orders the procedural element to begin executing the normal RUNNING logic. This command is only valid when the procedural element is in the IDLE state.

STOP: This command orders the procedural element to execute the STOPPING logic . This command is valid when the procedural element is in the RUNNING, PAUSING, PAUSED, HOLDING, HELD, OR RESTARTING state.

HOLD: This command orders the procedural element to execute the HOLDING logic. This command is valid when the procedural element is in the RUNNING, PAUSING, PAUSED or RESTARTING state.

UN-HOLD: This command orders the procedural element to execute the RESTARTING logic to safely return to the RUNNING state. This command is only valid when the procedural element is in the HELD state.

SUSPEND

UN-SUSPEND

ABORT: This command orders the procedural element to execute the ABORTING logic. The command is valid in every state except for IDLE, COMPLETED, ABORTING and ABORTED.

CLEAR:

5.5.3 Procedural Interface Model

Table 1 - Procedural Interface State and Command Matrix

States					**Commands**					
	Auto Trans.	Enable	Start	Pause	Un-Pause	Hold	Un-Hold	Complete	Stop	Abort
Stopped		x							x	x
Enabling	X								x	x
Idle			x						x	x
Starting	X								x	x
Running			x	x		x		x	x	x
Pausing	X								x	x
Paused					x				x	x

Un-Pausing	X								x	x
Holding	X								x	x
Held							X		x	x
Un-Holding	X								x	x
Completing	X								x	x
Completed	X	x							x	x
Stopping	X									x
Stopped										x
Aborting	X									
Aborted										

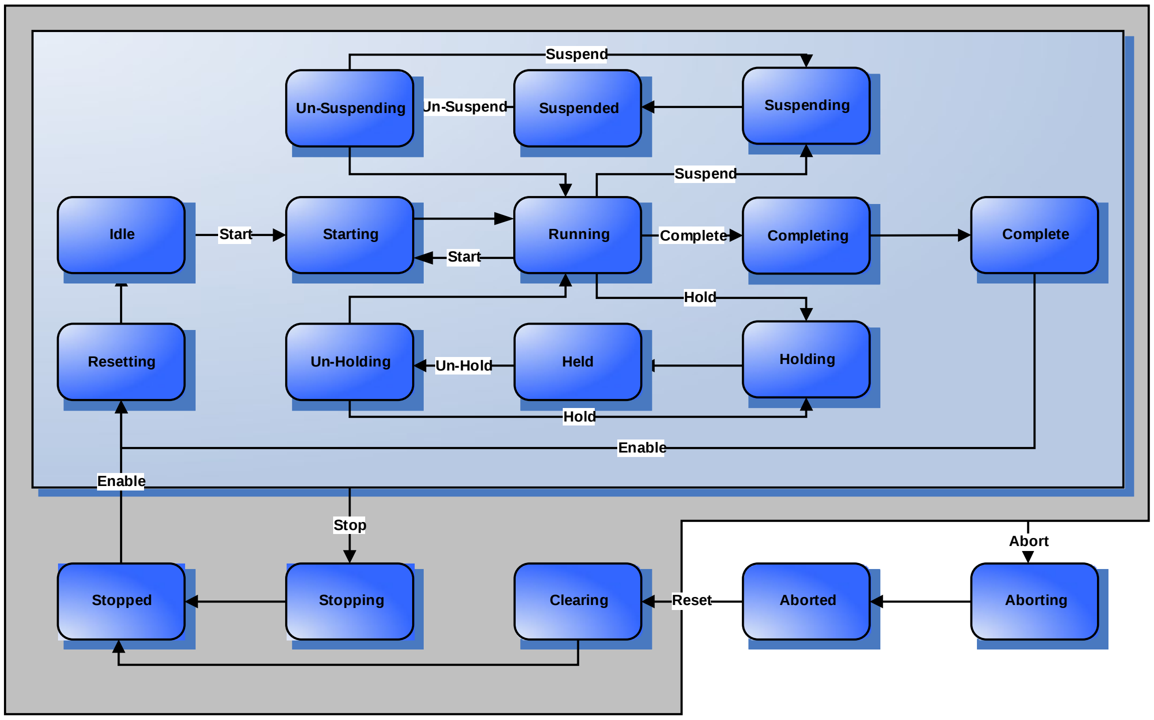

Figure 8 - Procedural Interface State and Command Model

5.6 Automation Module Implementation Examples

5.6.1 System View

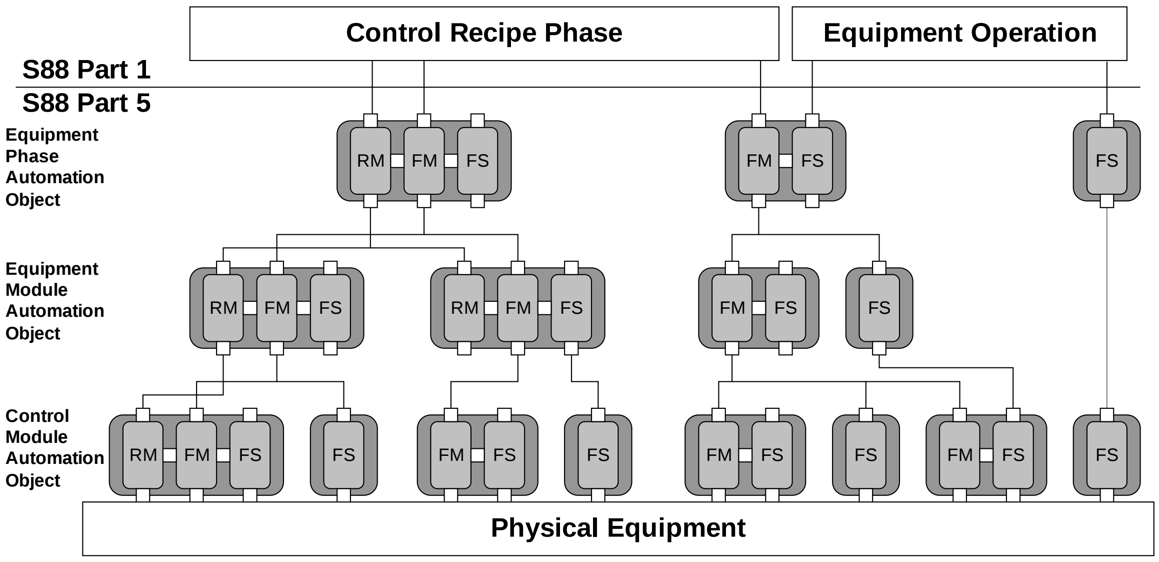
Control Recipe Phase
Equipment Operation
S88 Part 1
S88 Part 5
Equipment Phase Automation Object
Equipment Module Automation Object
Control Module Automation Object
RM
FM
FS
Physical Equipment

Figure 9 – Example organization of Automation Objects

5.6.2 Multi Procedural Interface Ownership

5.6.3 Dynamic Binding

5.6.4 Resource Implementations

5.6.5 Module Encapsulation

6 Completeness

The number of object models, objects and attributes supported, as defined in Clause 6 shall determine the degree of completeness of a specification or application.

6.1 Compliance

Any assessment of the degree of compliance of a specification shall be qualified by the following:

1. The use of object names
2. The use of the attributes for each supported object
3. A statement of the degree to which they then conform partially or totally to definitions and attribute names;

In the event of partial compliance, areas of non-compliance shall be explicitly identified.

6.2 Conformance

Any assessment of the degree of conformance of an application shall be qualified by the following:

1. Documentation of the object models and objects, as listed in Clause 5.6 through Clause 5.19, conformed to;
2. Documentation of the attributes conformed to;
3. A statement of the mapping of the application's attributes and object names to the objects and attributes listed in part of the standard.

In the event of partial conformance, areas of non-conformance shall be explicitly identified.

Any additional batch production record objects and attributes supported by an application should be explicitly identified as extensions to the standard format.

6.3 Extending the object model

The objects in a batch production record represent a wide range of data types and formats intended to cover common requirements in the industry. In order to accommodate industry, business and application requirements in the future it may be necessary to add new objects and/or attributes to an implementation of the batch production record.

When objects or attributes are added to an implementation of the object model the following rules shall be followed to be in compliance with this part.

1. New objects and attributes may be created provided their names clearly identify them as different from objects and attributes defined in clause Erreur : source de la référence non trouvée of this part;
2. Existing objects and attributes shall not be redefined.

Annex A Working Notes – Definitions

Shared Concepts (common terms for Making and Packing, but not explicitly defined in S88)

Term	Make2Pack Description
Command	A signal to a component to change mode or state. Commands are generally momentary in nature and latched by the receiving component and reset when the command has been executed.
Control Component	Control software part of a Unit, Equipment Module or Control Module.
Cycling	The repetitive motion of a machine for discrete manufacturing. The machine typically performs 1 cycle for each processed item.
Device	An apparatus for performing a prescribed function [Definition from ANSI/ISA-51.1 – 1979 (R1993)] Used here to describe elements of the physical environment that control systems connect to/control.
External Control	Control that acts externally to a control component.
Internal Control	Control that acts internally within a control component. This will only apply to procedural elements in automatic mode.
State	Defined in S88.01; hold pending changes there. The operating conditions of a component, whether it is a lower level control module or a higher level procedural component. States are only relevant if the component is in automatic or semiautomatic mode. A state can be stationary or transient. A transient state is a temporary state that the component is in while moving from one quiescent state to another. Examples of quiescent states are: • Stopped • Producing • Cleaning • Cycling • Halted Examples of transient states are: • Starting • Stopping S88: *state: The condition of an equipment entity or of a procedural element at a given time.* *NOTE — The number of possible states and their names vary for equipment and for procedural elements.*

S88 Physical Model (parallel terms for Making and Packing) – Non-Normative

Batch term	Packaging term	Make2Pack Description
Equipment Entity	Mechatronics	Object oriented machine design. In object oriented machine design an object is the combination of hardware and logic.
Process Cell	Production Line (per S95 maybe)	A collection of one or more machines, linked together, to perform one or multiple tasks of the process for one or more products in a defined sequence. • Continuous Process (e.g. forming line in the food industry) • Converting Line (e.g. paper, fibers) • Discrete Manufacturing (e.g. assembly) • Packaging Line (from filling to secondary and tertiary packaging) S88: *A process cell contains all of the units, equipment modules, and control modules required to make one or more batches/lots*.
Unit	Machine	In packaging, the unit corresponds to the logical grouping of mechanical and electrical assemblies that historically have been called machines. The term unit may apply to single function machine (filler, capper) or a multifunctional machine (monoblock filler/capper or any other configuration that combines functions within a single machine frame and control system). A multifunctional machine/unit can perform some or all functions of a packaging line, corresponding to process cell, that perform some or all of the functions of primary, secondary and tertiary packaging. A multifunctional machine may be logically broken down into several units corresponding to the individual functions. S88: *A unit is made up of equipment modules and control modules. The modules that make up the unit may be configured as part of the unit or may be acquired temporarily to carry out specific tasks.*
Equipment Module	Equipment Module	A group of equipment and its associated control located within the context of a machine or unit, designed and or arranged to perform a certain function. This can be equivalent to what is often called a Station in a discrete machine. The equipment module may be made up of control modules and subordinate equipment modules . E.g. the capper station in bottle filling machine. S88: *Physically, the equipment module may be made up of control modules and subordinate equipment modules. An equipment module may be part of a unit or a stand-alone equipment grouping within a process cell. If engineered as a stand-alone equipment grouping, it can be an exclusive-use resource or a shared-use resource.*

Batch term	Packaging term	Make2Pack Description
Control Module	Control Module	A Control Module is the lowest module in a physical model breakdown of a unit. The term Control Module relates to the combination of (a) physical device(s) and the lowest level control component that controls this(these) device(s) to carry out a physical process action. There may be control modules without directly associated physical devices. These control modules coordinate/supervise/sequence other control modules. NOTE: The use of the term control module to describe the supervisory/sequencing/coordinating functions is proving confusing and difficult to convey the concepts. Needs further consideration. Make2Pack examples of Control Modules: Servo Conveyor Pneumatic Cylinder with feed-back S88: *A control module is typically a collection of sensors, actuators, other control modules, and associated processing equipment that, from the point of view of control, is operated as a single entity. A control module can also be made up of other control modules. For example, a header control module could be defined as a combination of several on/off automatic block valve control modules.*
Compound Control Module	n/a	Control Module that coordinates/sequences lower level Control Modules. The Compound Control Module in general does not connect directly to physical devices like lower level Control Modules.
Compound Equipment Module	n/a	Equipment Module that coordinates/sequences lower level Equipment Modules. The Compound Equipment Module, like lower level Equipment Modules, does not connect directly to physical devices.
Unit/Machine Control Element (placeholder for Coordination Control element)	n/a	Supervisory Control Element on the unit level that handles general unit functionality, (e.g. reset, mode propagation, alarm management)

S88 Procedural Model (Making terms from S88 part 1 adopted by Packing)

Procedural Control: *Control that directs equipment-oriented actions to take place in an ordered sequence in order to carry out some process-oriented task.*

Recipe: *The necessary set of information that uniquely defines the production requirements for a specific product.*

NOTE — There are four types of recipes defined in this standard: general, site, master, and control.

Control Recipe: *A type of recipe which, through its execution, defines the manufacture of a single batch of a specific product.*

NOTE 1 — The control recipe may not be omitted from the recipe types model.

Equipment Control: *The equipment-specific functionality that provides the actual control capability for an equipment entity, including procedural, basic, and coordination control, and that is not part of the recipe.* Code in equipment control that issues commands to underlying control components in order to affect the process as intended. A procedural element is only active while carrying out its task, then becomes idle.

Procedural Control Element	Recipe Procedural Element	Equipment Procedural Element	Making Examples	Packing Examples
Procedure: *The* sequencing *strategy for carrying out a process*, defined in terms of an ordered set of unit procedures. *NOTE — In general, it refers to the strategy for making a batch within a process cell. It may also refer to a process that does not result in the production of product, such as a clean-in-place procedure.*	Recipe Procedure: *The part of a recipe that defines the [sequencing] strategy for producing a batch.*	Equipment Procedure: *A procedure that is part of equipment control.* NOTE: An equipment procedure has the same function as a recipe procedure.	*"Make PVC"*	The Procedure for a discrete Production Line relates to the higher level process of producing goods, e. g. how to get the line ready, how to get into the producing state or how to clean out the line.
Unit Procedure: *A unit procedure consists of an ordered set of operations that cause a contiguous production sequence to take place within a unit. Only one unit procedure may be active in a unit at any time.*	Recipe Unit Procedure: *A unit procedure that is part of a recipe procedure in a master or control recipe.*	Equipment Unit Procedure: *A unit procedure that is part of equipment control.*	*—Polymerize VCM.* *—Recover residual VCM* *—Dry PVC*	*Producing* *Cleanout* Trimset Setup Maintenanc e Jogging

Procedural Control Element	Recipe Procedural Element	Equipment Procedural Element	Making Examples	Packing Examples
Operation: *An operation is an ordered set of phases that defines a major processing sequence that takes the material being processed from one state to another, usually involving a chemical or physical change.* *Operation boundaries should be located at points in the procedure where normal processing can safely be suspended. Only one operation should be active in a unit at any time.*	Recipe Operation: *An operation that is part of a recipe procedure in a master or control recipe.*	Equipment Operation: *An operation that is part of equipment control.*	• *Preparation : Pull a vacuum on the reactor and coat the walls with antifoulant.* • *Charge: Add demineralized water and surfactants.* — • *React: Add VCM and catalyst, heat, and wait for the reactor pressure to drop.*	*Starting* *Aborting* *Holding* *Suspending* *Held*
Phase: *The smallest element of procedural control that can accomplish a process-oriented task is a phase. Phases are implemented by principle control within equipment.* NOTE: A procedural element can only have one master at a time.	Recipe Phase: *A phase that is part of a recipe procedure in a master or control recipe.*	Equipment Phase: *A phase that is part of equipment control.* NOTE: Equipment phases commanded by recipe phases require procedural states similar to those of the S88 example, but it is possible to have equipment phases that are directed by other control components based on different state models.		

Procedural Control Element	Recipe Procedural Element	Equipment Procedural Element	Making Examples	Packing Examples
Principle Control: Principle control is responsible for achieving the control actions of equipment procedural elements (e.g. the phase). Principle control is sequential in nature and has quiescent state(s) which typically require an external command to become active or to move from state to state.	N/A	Principle Control: Code in equipment control that issues commands to other control components in order to affect the process as intended. An equipment phase will only be active while it is carrying out its task. When complete the equipment phase becomes idle.	Equipment Logic	

Modes of Procedural Control (terminology differences for Making and Packing)

Making term	Packing term	Make2Pack Description
Mode (Procedural Mode)	Mode (Unit Control Mode)	Modes apply to 3 different types of entities Unit Modes: define the operation of a machine in terms of what it does. Examples might be: • Producing product 1 • Producing product 2 • Clean-out • Jogging • Single Step Unit modes are machine dependent and defined by the machine designers. • Procedure Modes: define how procedural elements respond to command inputs. Only 3 different procedural modes should be used: Automatic • Semiautomatic • Manual Mode for Equipment Entities: Equipment Entities are the lowest level control component and the only type of component that connect to the physical devices.

Making term	Packing term	Make2Pack Description
		Equipment entities do not have procedural control. There are only 2 modes for equipment entities: Manual: The logic within the control component is not active. The physical device is controlled directly by the operator. Automatic: The logic within the control component is active. The logic controls the physical device based on inputs from other control components. The mode definitions for Procedure Modes and Equipment Entity Modes correspond to the definitions in S88 Part 1. S88: *mode: The manner in which the transition of sequential functions are carried out within a procedural element or the accessibility for manipulating the states of equipment entities manually or by other types of control.*
Unit Procedure	Mode	Procedural element for the highest level procedural control within a Unit. S88: *unit procedure: A strategy for carrying out a contiguous process within a unit. It consists of contiguous operations and the algorithm necessary for the initiation, organization, and control of those operations.* *A unit procedure consists of an ordered set of operations that cause a contiguous production sequence to take place within a unit. Only one operation is presumed to be active in a unit at any time. An operation is carried to completion in a single unit. However, multiple unit procedures of one procedure may run concurrently, each in different units. Examples of unit procedures include the following:* — *Polymerize VCM.* — *Recover residual VCM.* — *Dry PVC.*
Automatic Mode	everything	The component does not respond to human commands, except as explicitly required by the automation sequence. Responds only to commands from other logic while executing its own internal logic.
Semiautomatic Mode	n/a	The component responds to human commands in lieu of commands from other logic while still executing its own internal logic. Procedural components will typically advance to the next step in its sequence at human command. In some applications this mode may be known as Maintenance.
Manual Mode	n/a	The component responds to human commands to activate its outputs and does not execute its own internal logic. For procedural components manual mode overrides the procedure and human commands affect the process directly. Manual mode is optional.

Annex B Questions and Answers

Annex C PID Example

C.1 PID Implementation

The purpose of this example is to show how the Control Shell can be used for a PID controller, implemented as a compound control module.

C.2 MEC PID Control Module

Figure 10 illustrates the MEC elements that make up a PID Control Module. It contains a Resource Manager (RM), Functional Manager (FM), and a Functional Strategy (FS). The FS illustrates that the FS is made up of contained or referenced Control Modules.

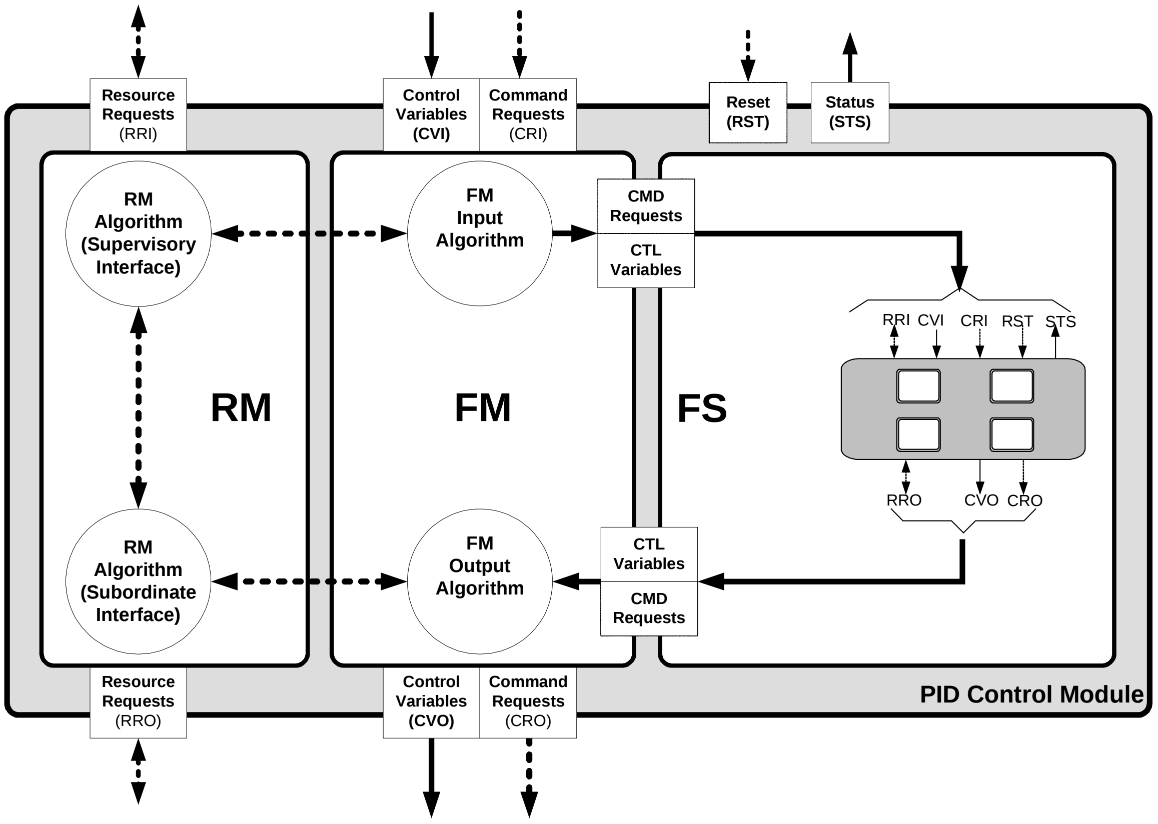

Figure 10 - MEC PID Control Module

Figure 11 illustrates the Functional Strategy of the PID Control Module. It is made up of Control Modules for Analog Inputs, Virtual Inputs, Analog Outputs, and Control Shell around a PID Function Block.

Virtual Inputs are inputs from HMI systems or other equipment entities.

All of the inputs to the basic function block have a Control Shell wrapper. The output from the basic function block has a Control Shell wrapper, and the basic PID function block also has a Control Shell wrapper.

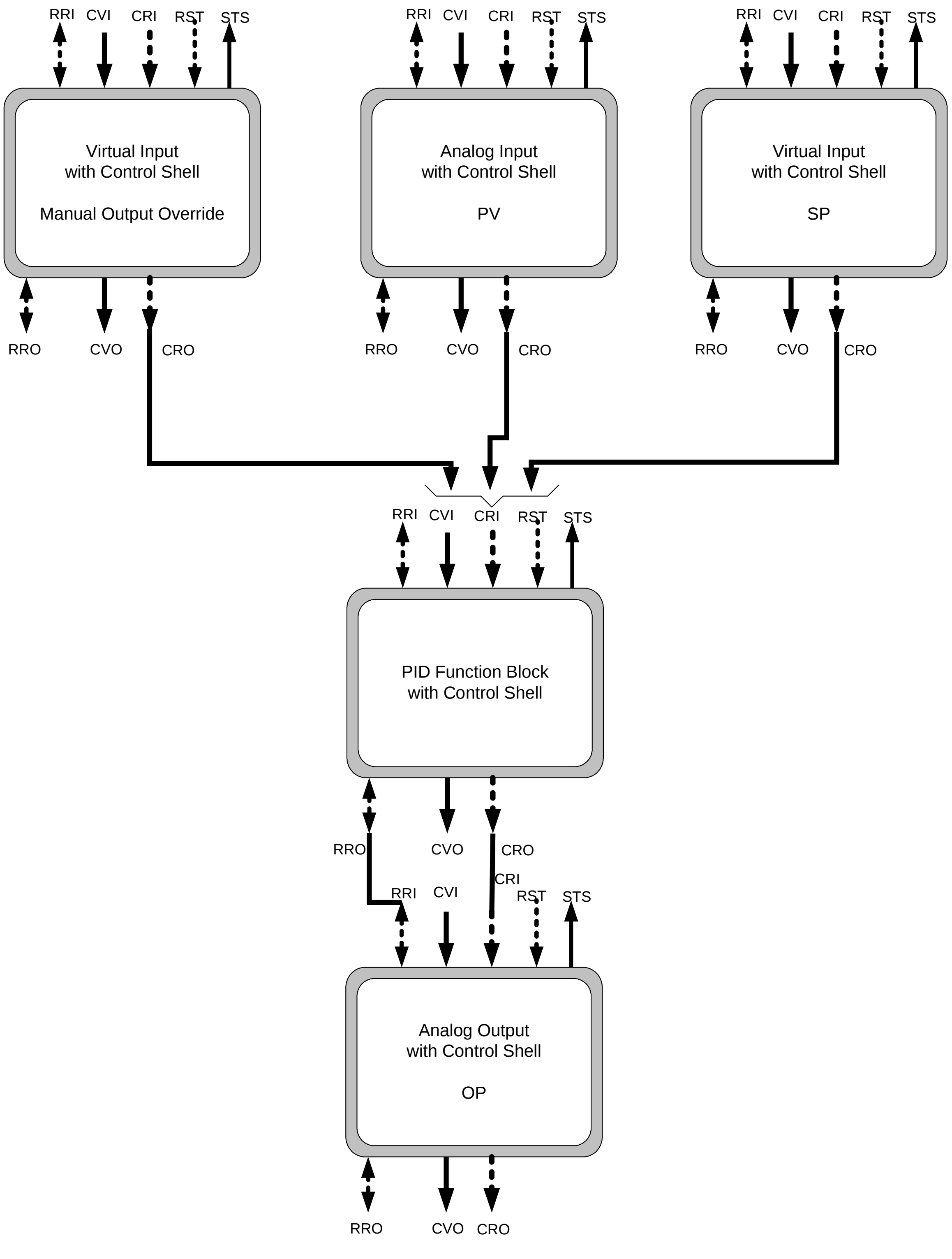

Figure 11 – PID basic controller in MEC format

C.3 PID Control Element with Control Shell

Figure 12 illustrates the encapsulation of the PID function block with a control shell. This provides the resource management and switching capability contained in the shell with the functional strategy defined in the IEC 61499 PID Function Block.

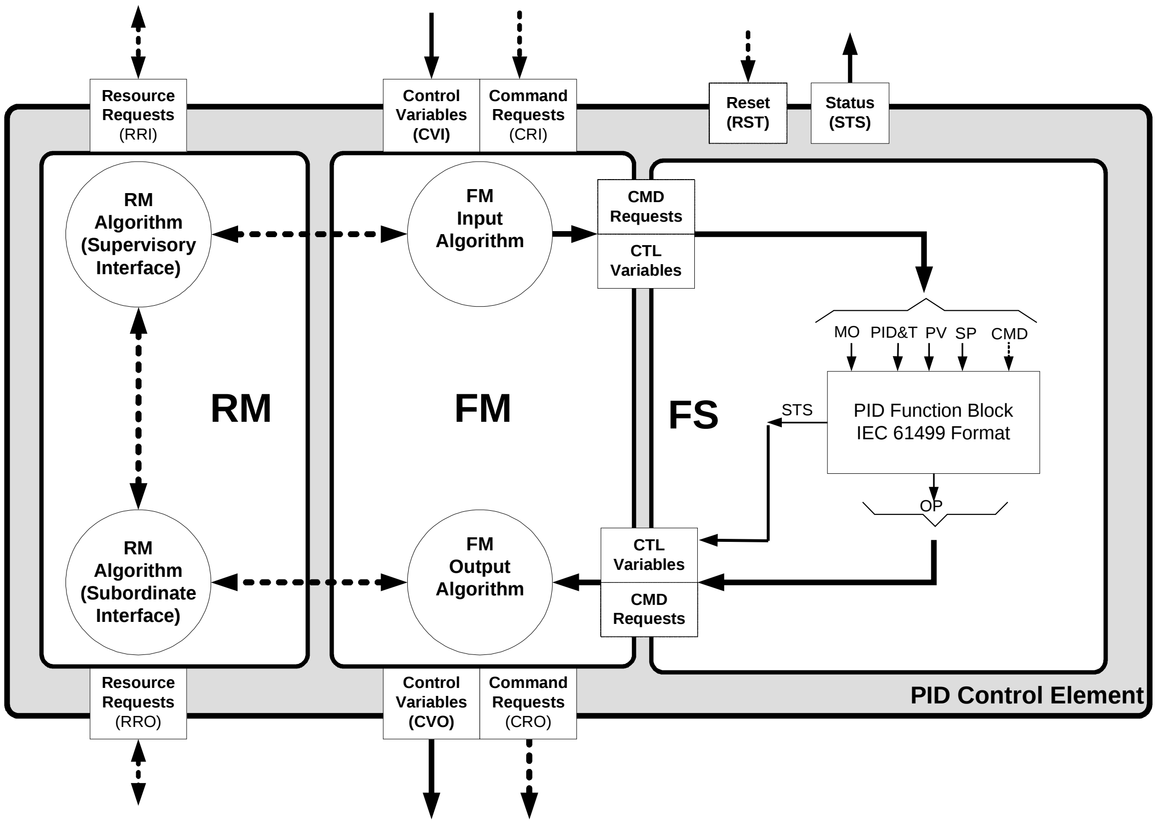

Figure 12 - PID Control Element

Figure 13 below shows Functional Strategy represented using an IEC 61499-1 Function Block for a simple PID control with typical PID variables exposed. Figure 13 also shows a typical implementation for Auto/ Manual functionality that may be included with the function block.

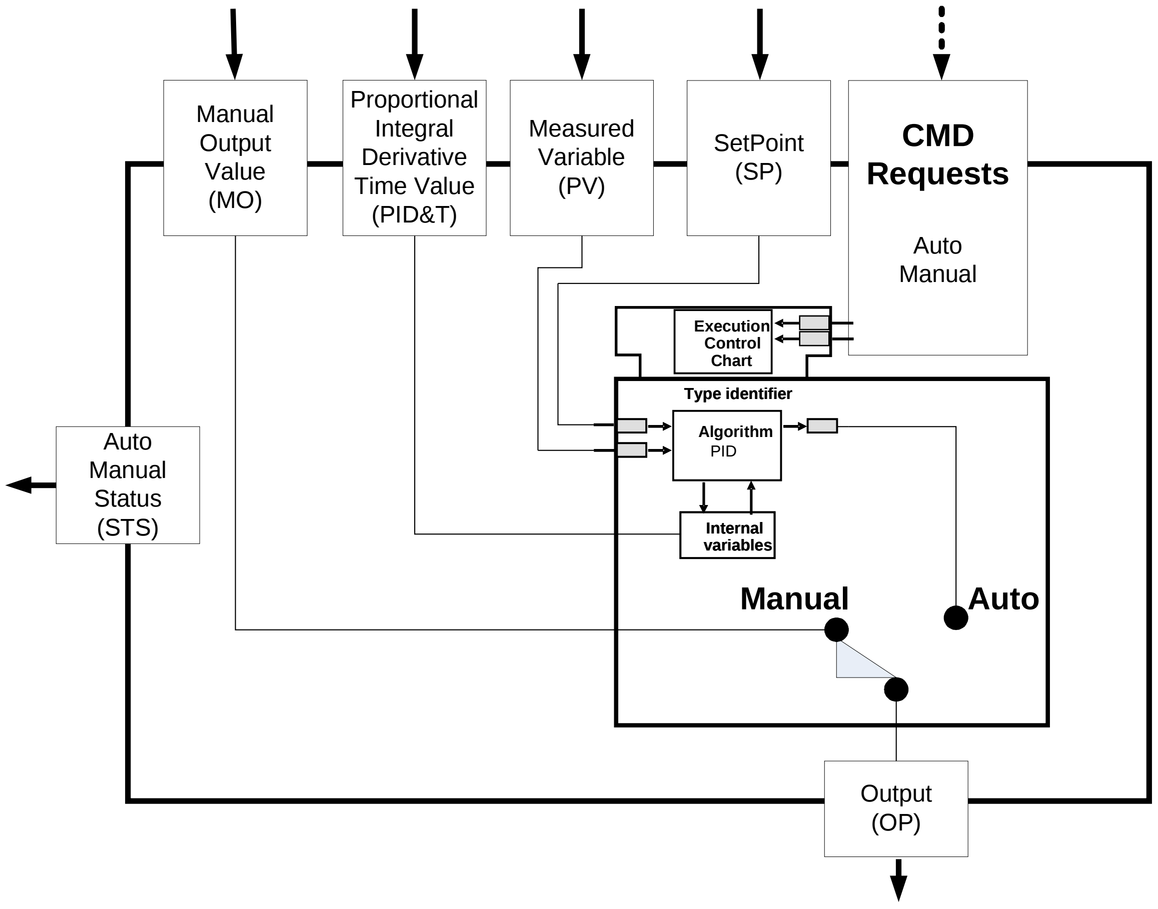

Figure 13 – PID Functional Strategy

C.4 MEC Analog Input Functional Strategy for PID Measured Variable Input

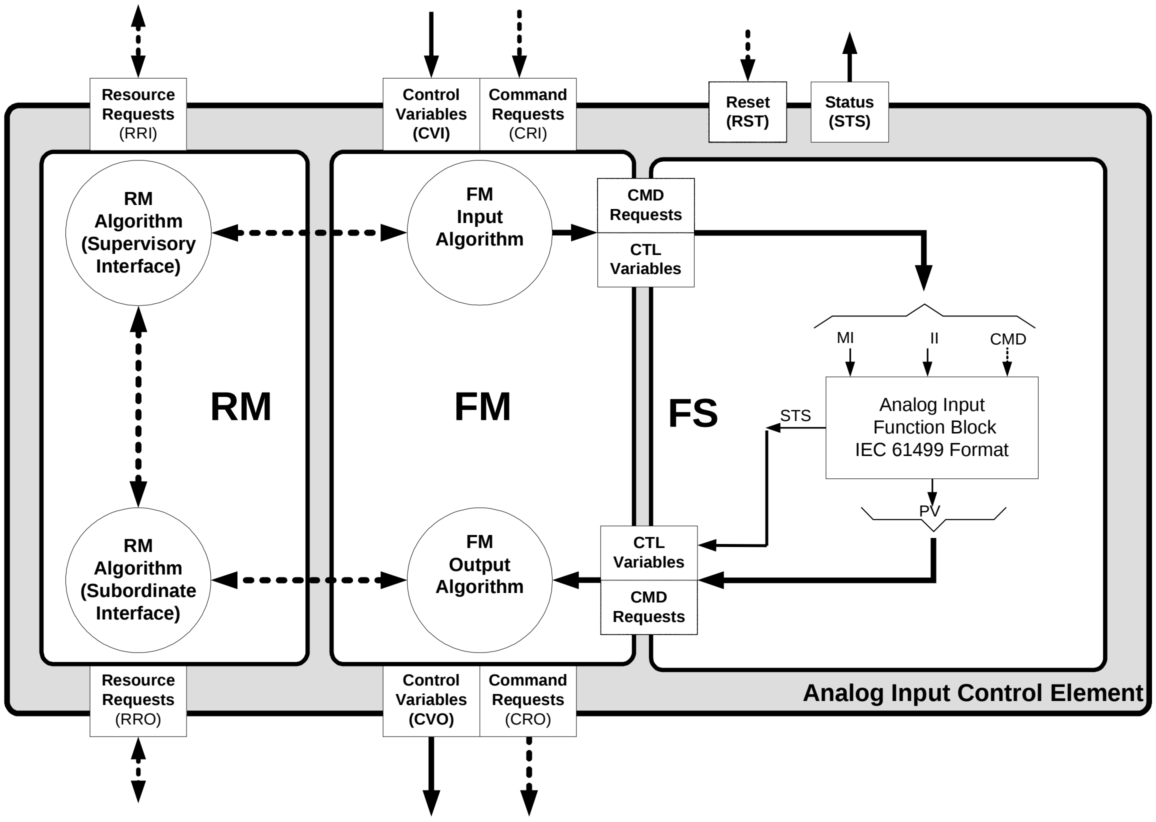

Figure 14 - MEC Analog Input

Figure 15 below shows the IEC 61499-1 Function Block for a simple Analog Input control with typical Analog Input variables exposed. This Analog Input is designed to connect to real inputs in the IO system. Figure 15 also shows the classical implementation for Auto / Manual functionality for an Analog Input.

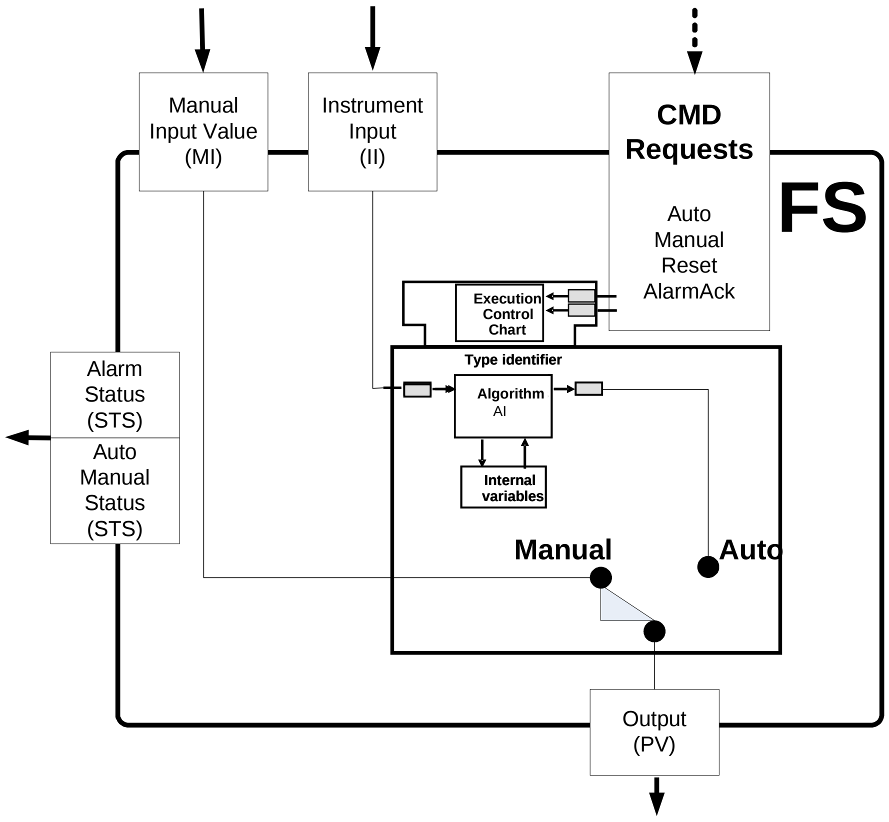

Figure 15 - Analog Input

C.5 MEC Virtual Input Functional Strategy for PID Manual Output and PID Setpoint Inputs

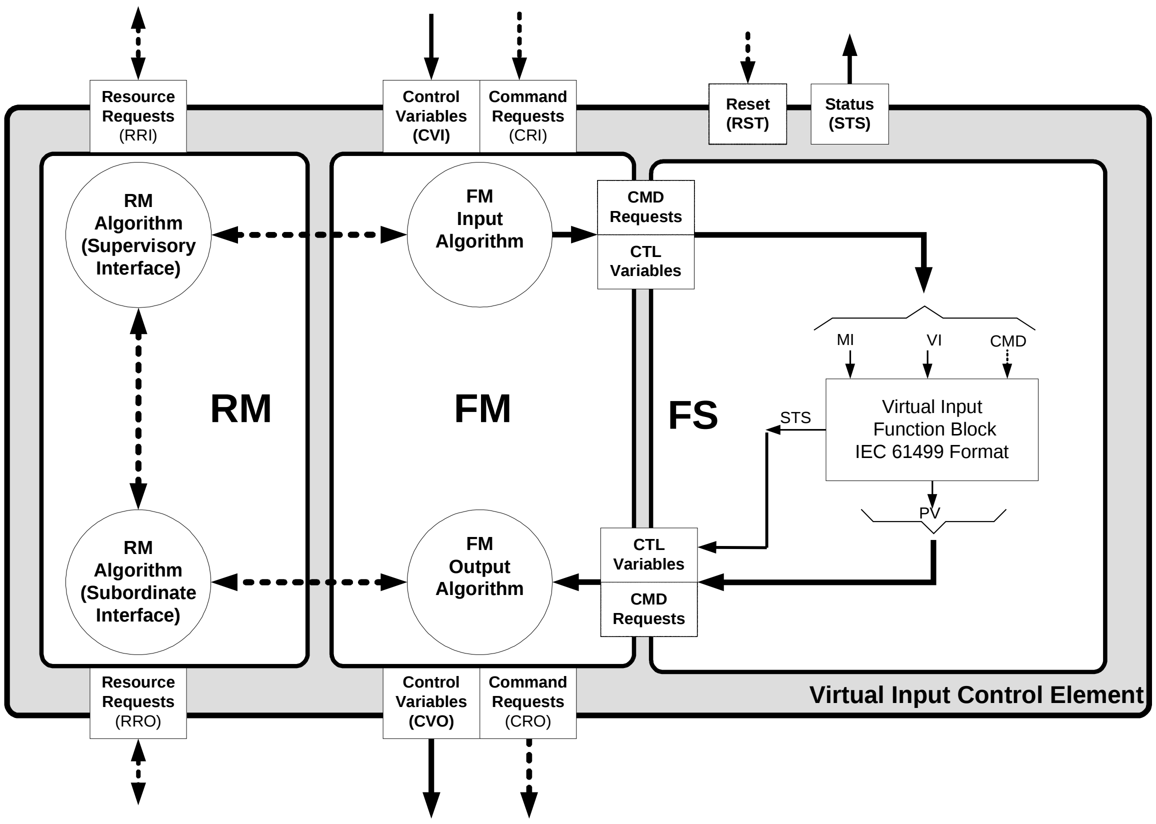

Figure 16 – MEC Virtual Input

Figure 17 below shows the IEC 61499-1 Function Block for a simple Virtual Input control with typical Virtual Input variables exposed. This Virtual Input is designed to connect to outputs from other control components. Figure 17 also shows the classical implementation for Auto / Manual functionality for an Virtual Input.

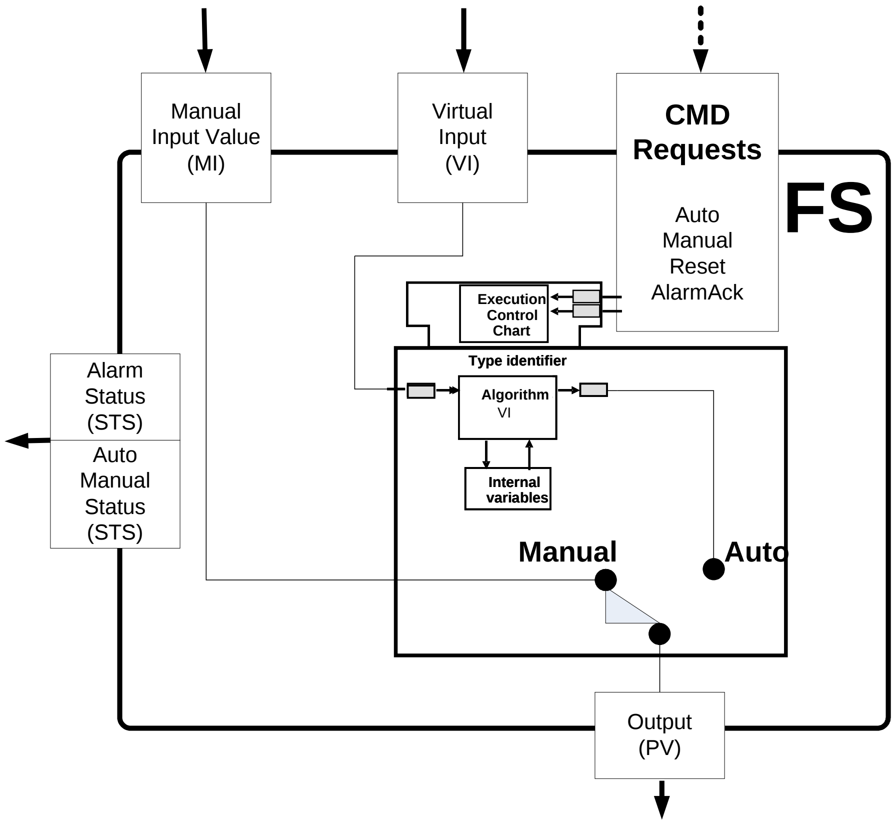

Figure 17 - Virtual Input

C.6 MEC Analog Output Functional Strategy for PID Output

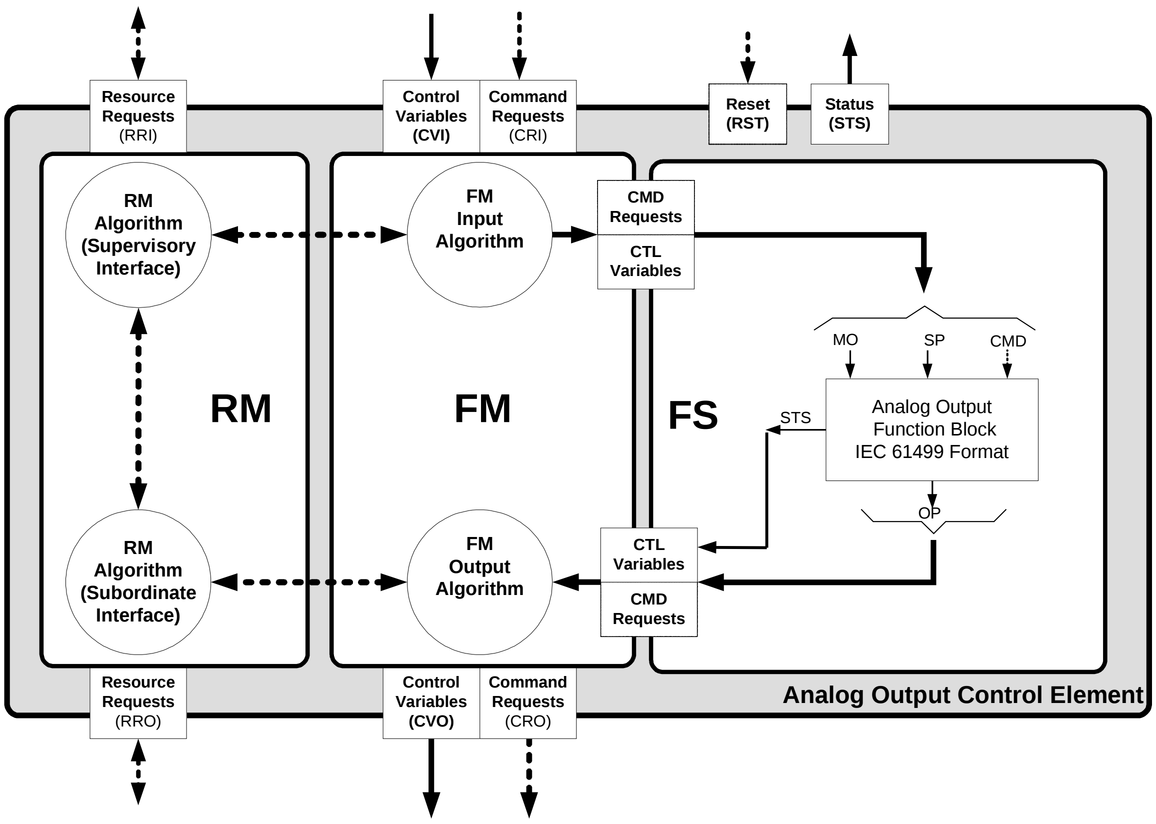

Figure 18 – MEC Analog Output

Figure 19 below shows the IEC 61499-1 Function Block for a simple Analog Output control with typical Analog Output variables exposed. This Analog Output is designed to connect to outputs in the IO system. Figure 19 also shows the classical implementation for Auto / Manual functionality for an Analog Output.

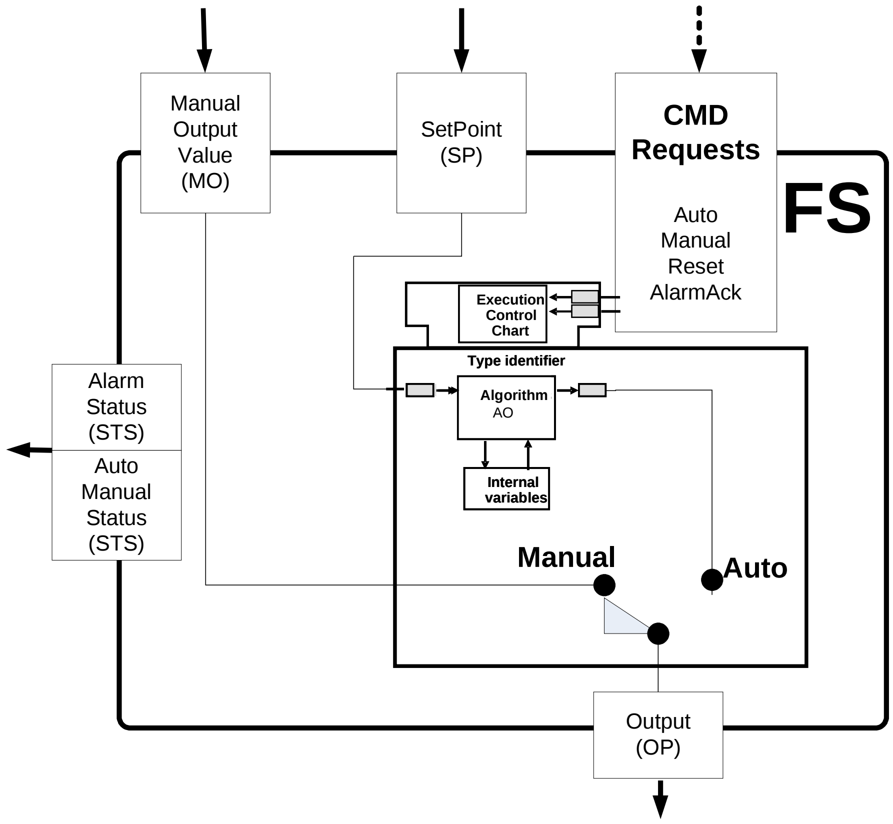

Figure 19 - Analog Output

Annex D - Cascaded PID Implementation

6.4 MEC Cascaded PID Implementation

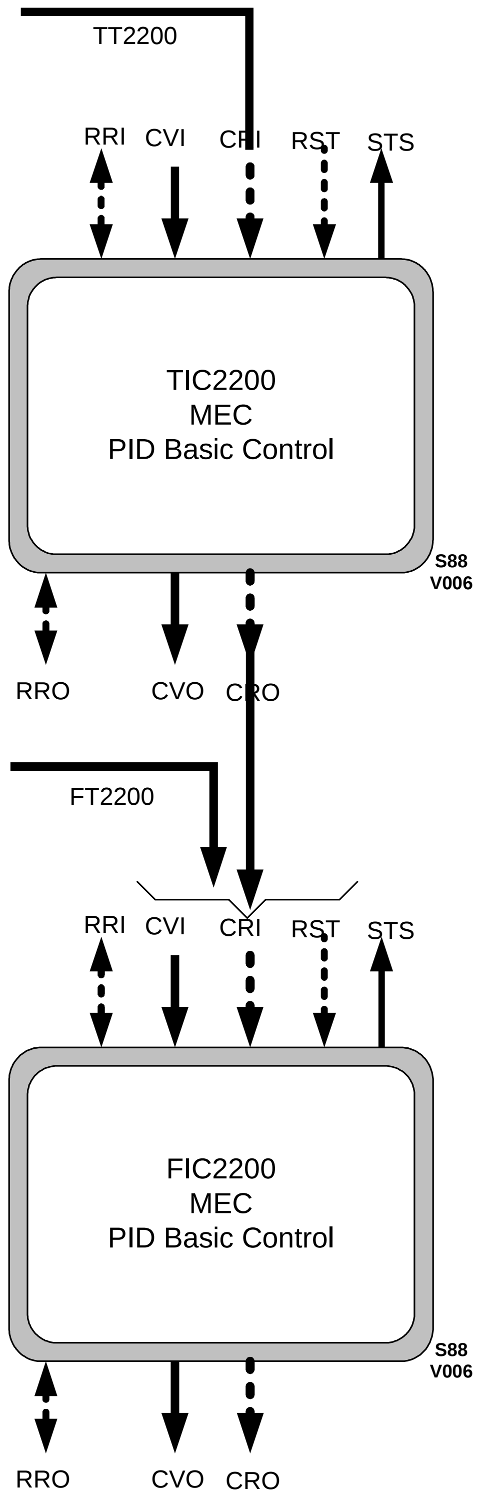

Figure 13

Annex E – Prototype Automation Object Example

1. **Resource Manager (RM) Supervisory Interface**

   MEC.RRI.Command.RequestID – (INT) – This input tells the Resource Manager who is requesting a change in the resource that will be controlling the Automation Module. This value is verified against the MEC.RM.Admin.ValidRequestID[] to validate the sending Module ID is allowed to send a value. This input is the Module ID of the specific equipment entity that is requesting to change the value of the resource that is controlling the Automation Module. This input is implemented using an integer value and has a range from 1 to 32767.

   MEC.RRI.Command.ResourceID – (DINT) – This input tells the Resource Manager what resource will be controlling the Automation Module. This input is only set by the equipment entity specified by the Module ID in MEC.RRI.Command.RequestID, when the supervisor verifies the Automation Module is listening to it. The upper INT is the Module ID of the Automation Module sending the value and must be the same as the value of the MEC.RRI.Command.RequestID. The lower INT is the Module ID of the specific equipment entity that will be controlling the Automation Module. The value for this Resource Module ID is 1 to 32767. However there is a special value, -1, that will reset the Resource Manager back to a default state. See the Erreur : source de la référence non trouvée below. This input is implemented using a double integer value.

Table 1- MEC.RRI.Command.ResourceID (Lower INT)

-1	Resource Manager Reset Request
0	No Module in Control
1	Module ID 1
•	
•	
•	
32767	Module ID 32767

6.4.1.1.1 Resource Request Inputs (RRI) Status

MEC.RRI.Status.RequestID – (INT) – This status value indicates the Module ID of the supervising equipment entity that has been acknowledged by the Resource Manager to change the controlling resource of the Automation Module. Once this value is set and read by the supervisor, the supervisor can then send the value for the resource it wants to control this Automation Module. This input is implemented using an integer value.

MEC.RRI.Status.ResourceID – (INT) – This status value indicates the Module ID of the equipment entity that has been allowed to control the Automation Module. This input is implemented using an integer value.

MEC.RRI.Status.RequestStatus – (INT) – This status value indicates the status of the request to change the resource controlling the Automation Module. This input has many different values that indicate the specific states of the request. At a minimum there will be five (5) values as shown in Erreur : source de la référence non trouvée:

Table 2 - MEC.RRI.Status.RequestStatus

-2	Failed to Release
-1	Failed to Control
0	Initialized
1	Success
2	Pending Request

MEC.RRI.Status.LastResourceID – (DINT) – This status value is the last ResourceID command that was sent. This input is a copy of the incoming command and is implemented using a double integer value. This value is updated everytime a new ResourceID is received.

MEC.RRI.Status.LastRequestStatus – (INT) – This status value is the returned status for the last ResourceID command that was sent. This input is a copy of the status, MEC.RRI.Status.RequestStatus, when the last command was executed. This value is updated everytime a new ResourceID is received.

2 Resource Manager (RM) Subordinate Interface

6.4.1.1.2 Resource Request Outputs (RRO) Status

MEC.RRO.Status.RequestID – (INT) – This input tells the Resource Manager who is allowed to request a change in the resource controlling the subordinate Automation Module. If this input matches the Automation Module's ID, then the Resource Manager can send the MEC.RRO.Command.ResourceID to the subordinate Automation Module. This input is implemented using an integer value.

MEC.RRO.Status.ResourceID – (INT) – This input tells the Resource Manager what resource has control of the subordinate Automation Module. This is implemented using an integer value.

MEC.RRO.Status.RequestStatus – (INT) – This input tells the Resource Manager the value of the subordinate Automation Module’s request status. This is implemented using an integer value and will support at least the following values.

-2	Failed to release Subordinate Automation Module
-1	Failed to control Subordinate Automation Module
0	Initialized
1	Success
2	Pending Request
3	Acquiring Subordinate Automation Modules
4	Releasing Subordinate Automation Modules

6.4.1.1.3 Resource Request Outputs (RRO) Commands

MEC.RRO.Command.RequestID – (INT) – This output is sent to the Subordinate Resource Manager to request access to change the resource that is supervising the Subordinate Automation Module. This output is the Module ID of this Automation Module. This output is implemented using an integer value and has a range from 1 to 32767.

MEC.RRO.Command.ResourceID – (DINT) – This output is sent to the Subordinate Resource Manager to change what resource will be controlling the Subordinate Automation Module. This output is only set if the MEC.RRO.Status.RequestID is the Module ID of this Automation Module. The upper INT is the Module ID of the Automation Module sending the value and would be the Module ID of this Module. The lower INT is the Module ID of the specific equipment entity that will be controlling the Automation Module. Typically this will be the Module ID of this Automation Module. This lower INT can also be a -1 which will send a reset command to the subordinate modules. See Table below. This output is implemented using a double integer value.

-1	Resource Manager Reset
0	No Module in Control
1	Module ID 1
•	
•	
•	

32767	Module ID 32767

3 Resource Manager (RM) Administrative Interface

6.4.1.1.4 Resource Manager (RM) Administration

MEC.RM.Admin.ValidRequestID[] – (INT Array) – This value is usually configured at design time, but can be exposed to allow runtime changes. It lists the valid Module IDs that can supervise this Automation Module. The valid values follow the supported types of the Command Processor “WHO” array, see section XXXXX Editors Note: Enter correct Section. Using the instance format, one set of values are as follows:

0	No Module in Control
1	Module ID 1
•	
•	
•	
32767	Module ID 32767

MEC.RM.Admin.SubordinatePassThru – (Boolean) – This value is usually configured at design time, but can be exposed to allow runtime changes. It determines whether the Resource Manager will automatically pass thru the resource requests or whether the subordinate interface will be controlled by the Function Manager.

MEC.RM.Admin.TimeoutPreset – (INT) – This value is typically configured at design time, but can be exposed to allow runtime changes. It indicates the preset for the timeout timer. This is the time that the Resource Manager will wait for the acquisition or release of subordinate Automation Modules before returning an error to MEC.RRI.Status.RequestStatus. This value is the number of milliseconds to wait.

MEC.RM.Admin.ReleaseDelay – (INT) – This value is typically configured at design time, but can be exposed to allow runtime changes. It indicates the preset for the release timer. This is the time the Resource Manager will wait before commanding a release of the subordinate Automation Module. This will keep the subordinate Automation Modules from being released and re-acquired very quickly. This value is the number of milliseconds to wait.

MEC.RM.Admin.ModuleID – (INT) – This value is typically configured at design time, but can be exposed to allow runtime changes. This value is the unique identifier for this Automation Module. This value is implemented as an integer.

MEC.RM.Admin.AsyncPreset – (INT) – This value is typically configured at design time, but can be exposed to allow runtime changes. This value is the Asynchronous communications timer preset that is used to delay further processing of the Resource Manager. This allows for a slow communications link between the parent Automation Module and the Resource Manager. This value is the number of milliseconds to wait.

MEC.RM.Admin.AllowChangeNoRelease – (BOOL) – This value is typically configured at design time, but can be exposed to allow runtime changes. This value is used to allow Resource Manager to bypass the requirement for ResourceID to be zero (0 = No Resource in control), when determining when to process a request to change RequestID or owner.

MEC.RM.Admin.SubordinateAsyncComms – (BOOL) – This value is typically configured at design time, but can be exposed to allow runtime changes. This value would be configured to allow the Subordinate interface to send both the RequestID and ResourceID at the same time rather than using the handshaking that is defined above.

MEC.RM.Admin.SubordinateChangeNoRelease – (BOOL) – This value is typically configured at design time, but can be exposed to allow runtime changes. This value would be configured to allow the Subordinate interface to request ownership when its Subordinate Automation Modules ResourceID is not zero.

6.4.2 Function Manager Incoming Commands

Control Request Inputs (CRI) Commands

6.4.2.1.1 Control Request Inputs (CRI) Commands

MEC.CRI.Command – (DINT) – This input allows the resource that is in command to request that the Automation Module execute a command. The upper integer will hold the ModuleID of the sending Automation Module. This value is validated against the MEC.FM.Admin.ValidModuleID[] array. The lower integer will hold the Command value which will be one of the following:

0	Undefined
1	Auto Mode of Action Request
2	Semi-Auto Mode of Action Request
3	Manual Mode of Action Request
4	Alarm Acknowledge
5	Reset Function Manager and Functional Strategy
6	Activate Request
7	Deactivate Request

The Mode of Action determines the action of the Functional Strategy. For example when a PID loop is put in to manual the input setpoint is written directly to the output of the PID function bypassing the PID algorithm altogether.

6.4.2.1.2 Control Request Inputs (CRI) Status

MEC.CRI.Status – (INT) – This is an output back to the commanding Automation Module to indicate the status of the command. The value will be one of the following:

-2	Command Failed – Not valid command
-1	Command Failed – Not Authorized
0	Not Defined
1	Command Succeeded and Processed
2	Command Succeeded – Not in Control of Module

MEC.CRI.LastCommand – (DINT) – This value is a copy of the last command executed regardless of command status. This value is updated every time a new command is received.

MEC.CRI.LastCommandStatus – (INT) – This value is a copy of the status after the last command was executed. This value is updated every time a new command is received.

6.4.2.1.3 Control Variable Inputs (CVI) Inputs

This is typically where physical inputs, interlocks, permissives, parameters or feedback signals are input to the Automation Module.

MEC.CVI.Permissive – (BOOL) – This input determines if the Automation Module has the appropriate permissions to activate or deactivate. This is input does not generate a fault when not enabled. This input controls the operation of the device. If there is a command and the permissive is off, the Functional Strategy will be ready to execute the command as soon as the permissive becomes true. If device is running and the permissive goes False, the device will stop running until the permissive becomes true again, at which time the device will start running. The Permissive logic for multiple permissive signals is done outside this Automation Module because this logic is custom for every implementation.

MEC.CVI.Interlock – (BOOL) – This input determines if the Automation Module is blocked from activating or deactivating by any other signals. The interlock will generate a fault if the interlock is false. To clear the fault and restart the device will require some external action. The Interlock logic for multiple interlock signals is done outside this Automation Module because this logic is different for every implementation.

MEC.CVI.Feedback.Activated – (BOOL) – This input is the status of the module output command from the Functional Strategy. This input lets the module know if the output actually came on.

MEC.CVI.Feedback.Deactivated – (BOOL) – This input is the status of the module output command from the Functional Strategy. This input lets the module know if the output actually went off.

4 Functional Manager (FM) Outgoing Command Outputs

6.4.2.1.4 Control Request Outputs (CRO) Commands

MEC.CRO.Command – (DINT) – This output allows the Automation Module to request a Subordinate Automation Module to change the Mode of Action. The upper integer will hold the ModuleID of this Automation Module. The lower integer value for ModeRequest will be one of the following:

0	Undefined
1	Auto Mode of Action Request
2	Semi-Auto Mode of Action Request
3	Manual Mode of Action Request
4	Alarm Acknowledge
5	Reset Request

Then Mode of Action determines the action of the Functional Strategy. For example when a PID loop is put in to manual the input setpoint is written directly to the output of the PID function bypassing the PID algorithm altogether.

6.4.2.1.5 Control Request Outputs (CRO) Status

MEC.CRO.Status – (INT) – This is an output back to the commanding Automation Module to indicate the status of the last command. The value will be one of the following:

-2	Command Failed – Not valid command
-1	Command Failed – Not Authorized
0	Not Defined
1	Command Succeeded and Processed
2	Command Succeeded – Not in Control of Module

6.4.2.1.6 Control Variable Outputs (CVO) Outputs

Currently there are no control outputs defined as being common to all Automation Modules. This is typically where physical outputs, or outputs to other control modules are specified. Since every Automation module is different, this specification is not defining inputs of this type.

If there were control outputs defined the attributes would have the following naming:

MEC.CVO.XXXXXX

where the XXXXXX would be replaced with the attribute name.

6.4.2.1.7 Functional Manager (FM) Administrative Inputs

This is typically configuration type data. The developer of the Automation Module may not even expose any attributes of this type.

6.4.2.1.8 Function Manger (FM) Administration

MEC.FM.Admin.ValidModuleID[] – (INT Array) – This value is typically configured at design time, but can be exposed to allow for runtime changes. It lists the valid Module IDs that can send commands to this Automation Module. The valid values follow the supported types of the Command Processor "WHO" array, see section XXXXX Editors Note: Enter correct Section. Using the instance format, one set of values are as follows:

0	No Module in Control
1	Module ID 1
•	
•	
•	
32767	Module ID 32767

MEC.FM.Admin.Alarm.ActionOnAcknowledge – (BOOL) – This input allows the resource that is in command to request that the Automation Module activate or deactivate after an alarm acknowledgement. If the value is 1 then the Automation Module will activate after an alarm acknowledgement and if the value is a 0 it will deactivate after an alarm acknowledgement.

MEC.FM.Admin.Alarm.ActionOnAlarm – (BOOL) – This input allows the resource that is in command to request that the Automation Module activate or deactivate after an alarm is present. If the value is 1 then the Automation Module will activate after an alarm is present and if the value is a 0 it will deactivate after an alarm is present.

MEC.FM.Admin.Alarm.ActionOnClear – (BOOL) – This input allows the resource that is in command to request that the Automation Module activate or deactivate after an alarm is cleared. If the value is 1 then the Automation Module will activate after an alarm is cleared and if the value is a 0 it will deactivate after an alarm is cleared.

MEC.FM.Admin.ModeTransition.ActivateRequest – (BOOL) – This input allows the resource that is in command to request that the Automation Module activate after the procedure mode has changed.

MEC.FM.Admin.ModeTransition.DeactivateRequest – (BOOL) – This input allows the resource that is in command to request that the Automation Module deactivate after the procedure mode has changed.

MEC.FM.Admin.ModeTransition.MaintainLastRequest – (BOOL) – This input allows the resource that is in command to request that the Automation Module to maintain the last command, activate or deactivate, after the procedure mode has changed.

MEC.FM.Admin.StateChangeAlarmTime – (INT) – This input allows the resource that is in command to configure the Automation Module’s alarm timer. This value indicates the maximum time that it should take to change from activate to deactivate. If this time is exceeded an alarm will be generated. This time is specified in milliseconds to wait.

5 MEC Status Overview

Every Automation Module will have status data that will indicate the state of the Automation Module. This data is an overall status and not limited to any one internal algorithm.

6.4.2.1.9 MEC Status

MEC.Status.AlarmPresent – (BOOL) – This status signal indicates that the Automation Module is in an alarm condition.

MEC.Status.AlarmAcknowledged – (BOOL) – This status signal indicates that the Automation Module has acknowledged all alarm.

MEC.Status.Reset – (BOOL) – This status signal indicates that the Automation Module has been reset to a default state.

MEC.Status.ModeCurrent – (INT) – This status value indicates the current Mode of Action for the Automation Module. The value will be one of the following values:

0	Undefined
1	Auto Mode of Action
2	Semi-Auto Mode of Action
3	Manual Mode of Action

MEC.Status.ModeRequested – (INT) – This status value indicates the Mode of Action that has been requested. The value will be one of the following values:

0	Undefined
1	Auto Mode of Action
2	Semi-Auto Mode of Action
3	Manual Mode of Action

6 MEC Reset Command

Every Automation Module will have a global command to reset the entire Automation Module.

MEC.Reset.ResetRequest – (DINT) – This input allows the resource that is in command to request that the Automation Module be reset to a default state. The upper integer will hold the ModuleID of the sending Automation Module. The lower integer holds the commanded state for the Reset, True or False. This is used throughout the Automation Module to reset all pieces, Resource Manager, Functional Manager and Functional Strategy, to a known default state.

7 MEC Control Shell Variable List

								Data Type
DeviceName							DeviceName	
	MEC						DeviceName.MEC	S8805
		RRI					DeviceName.MEC.RRI	Supervisory
			Command				DeviceName.MEC.RRI.Command	Command
				RequestID			DeviceName.MEC.RRI.Command.RequestID	Int (16 bit)
				ResourceID			DeviceName.MEC.RRI.Command.ResourceID	Int (32 bit)
			Status				DeviceName.MEC.RRI.Status	Status
				RequestID			DeviceName.MEC.RRI.Status.RequestID	Int(16 bit)
				ResourceID			DeviceName.MEC.RRI.Status.ResourceID	Int(16 bit)
				RequestStatus			DeviceName.MEC.RRI.Status.RequestStatus	Int(16 bit)
				LastResourceID			DeviceName.MEC.RRI.Status.LastResourceID	Int(32 bit)
				LastRequestStatus			DeviceName.MEC.RRI.Status.LastRequestStatus	Int(16 bit)
		RRO					DeviceName.MEC.RRO	Subordinate
			Command				DeviceName.MEC.RRO.Command	Command
				RequestID			DeviceName.MEC.RRO.Command.RequestID	Int (16 bit)
				ResourceID			DeviceName.MEC.RRO.Command.ResourceID	Int (32 bit)
			Status				DeviceName.MEC.RRO.Status	Status
				RequestID			DeviceName.MEC.RRO.Status.RequestID	Int(16 bit)
				ResourceID			DeviceName.MEC.RRO.Status.ResourceID	Int(16 bit)
				RequestStatus			DeviceName.MEC.RRO.Status.RequestStatus	Int(16 bit)
		RM					DeviceName.MEC.RM	RM
			Admin				DeviceName.MEC.RM.Admin	Admin
				ValidRequestID[#]			DeviceName.MEC.RM.Admin.ValidRequestID[#]	Array of Int (16 bit)
				SubordinatePassThru			DeviceName.MEC.RM.Admin.SubordinatePassThru	Bool
				TimeOutPreset			DeviceName.MEC.RM.Admin.TimeOutPreset	Int(16 bit)
				ReleaseDelay			DeviceName.MEC.RM.Admin.ReleaseDelay	Int(16 bit)
				ModuleID			DeviceName.MEC.RM.Admin.ModuleID	Int(16 bit)
				AsyncPreset			DeviceName.MEC.RM.Admin.AsyncPreset	Int(16 bit)
				AllowChangeNoRelease			DeviceName.MEC.RM.Admin.AllowChangeNoRelease	Bool
		CRI					DeviceName.MEC.CRI	CommandRequestInput
			Command				DeviceName.MEC.CRI.Command	Int (32 bit)
			Status				DeviceName.MEC.CRI.Status	Int (16 bit)
			LastCommand				DeviceName.MEC.CRI.LastCommand	Int (32 bit)
			LastCommandStatus				DeviceName.MEC.CRI.LastCommandStatus	Int (16 bit)
		CVI					DeviceName.MEC.CVI	ControlVariable
			Permissive				DeviceName.MEC.CVI.Permissive	Bool
			Interlock				DeviceName.MEC.CVI.Interlock	Bool
			Feedback				DeviceName.MEC.CVI.Feedback	Bool
				Activated			DeviceName.MEC.CVI.Feedback.Activated	Bool
				Deactivated			DeviceName.MEC.CVI.Feedback.Deactivated	Bool
		CRO					DeviceName.MEC.CRO	CommandRequestOutput
			Command				DeviceName.MEC.CRO.Command	Int (32 bit)
			Status				DeviceName.MEC.CRO.Status	Int (16 bit)
		CVO					DeviceName.MEC.CVO	ControlVariable
			XXXXXXX				DeviceName.MEC.CVO.XXXXXXX	Any
		FM					DeviceName.MEC.FM	FM
			Admin				DeviceName.MEC.FM.Admin	Admin
				ValidModuleID[#]			DeviceName.MEC.FM.ValidModuleID[#]	Array of Int (16 bit)
				Alarm			DeviceName.MEC.FM.Admin.Alarm	Alarm
					ActionOnAcknowledge		DeviceName.MEC.FM.Admin.Alarm.ActionOnAcknowledge	Bool
					ActionOnAlarm		DeviceName.MEC.FM.Admin.Alarm.ActionOnAlarm	Bool
					ActionOnClear		DeviceName.MEC.FM.Admin.Alarm.ActionOnClear	Bool
				ModeTransition			DeviceName.MEC.FM.Admin.ModeTransition	ModeTransition
					ActivateRequest		DeviceName.MEC.FM.Admin.ModeTransition.ActivateRequest	Bool
					DeactivateRequest		DeviceName.MEC.FM.Admin.ModeTransition.DeactivateRequest	Bool
					MaintainLastRequest		DeviceName.MEC.FM.Admin.ModeTransition.MaintainLastRequest	Bool
				StateChangeAlarmTime			DeviceName.MEC.FM.Admin.StateChangeAlarmTime	Int (16 bit)
		Status					DeviceName.MEC.Status	Status
			AlarmPresent				DeviceName.MEC.Status.AlarmPresent	Bool
			AlarmAcknowledged				DeviceName.MEC.Status.AlarmAcknowledged	Bool
			Reset				DeviceName.MEC.Status.Reset	Bool
			ModeCurrent				DeviceName.MEC.Status.ModeCurrent	Int(16 bit)
			ModeRequested				DeviceName.MEC.Status.ModeRequested	Int(16 bit)
			RequestStatus				DeviceName.MEC.Status.RequestStatus	Int(16 bit)
			Activated				DeviceName.MEC.Status.Activated	Bool
			Deactivated				DeviceName.MEC.Status.Deactivated	Bool
			Indeterminent				DeviceName.MEC.Status.Indeterminent	Bool
		Reset					DeviceName.MEC.Reset	Reset
			ResetRequest				DeviceName.MEC.Reset.ResetRequest	Int(32 bit)