

SEMI E133-0704

PROVISIONAL SPECIFICATION FOR AUTOMATED PROCESS CONTROL SYSTEMS INTERFACE

This provisional specification was technically approved by the Global Information & Control Committee and is the direct responsibility of the North American Information & Control Committee. Current edition approved by the North American Regional Standards Committee on March 14, 2004 and April 22, 2004. Initially available at www.semi.org May 2004; to be published July 2004. Originally published March 2004.

1 Purpose

1.1 In order to more efficiently facilitate the integration of Process Control Systems (PCS) such as run-to-run (R2R) control, fault detection (FD), fault classification (FC), fault prediction (FP), statistical process control (SPC), etc. into current and future fabs, there is a need to define interfaces for Process Control Systems that enable them to interact effectively and share data 1) among themselves and 2) with the other interdependent factory systems (including equipment data collection). This standard is intended for use by equipment suppliers, semiconductor manufacturers, equipment subsystem suppliers and control system suppliers who will utilize the PCS standards in future developed products.

2 Scope

2.1 This standard focuses on defining PCS capabilities, functional groups, and functional group interfaces. The standard is structured so that new functional groups can be added to the standard through future balloting. The components in this document include:

- Definitions of key terms and concepts pertinent to this domain,
- Description of the specification conventions used,
- Identification of the major functional groups within the scope of PCS,
- Description of the capabilities expected in each functional group (required and optional), and
- Definition of interfaces for these functional groups, including:
 - Data that is available across these interfaces. This includes the interface data descriptions (inputs and outputs). Future enhancements to this specification will include detailed data structures in the form of inheritance and aggregation models that are common to all PCS functional groups, as well as specific models for each,
 - Services supported and Interface behavior assumed, and

- Description of an approach for compliance verification testing.

2.2 The specific format and protocol implementation of these interface specifications, which are technology specific, will be delineated in supporting “dot” specifications to this standard. As an example a possible format is XML (which would include data type definitions and metamodel schemas).

2.3 In addition to the information embodied in this standard, the PCS TF will describe a validation process (prototyping) for the standard to identify and address real implementation issues as early as possible; results of this work will be included in future versions.

2.4 The provisional status of this specification will be removed when the following enhancements are added:

- Interface data structures and services specifications for the R2R, FD, FC, FP and SPC functional groups, and
- Additional compliance verification procedures for the R2R, FD, FC, FP and SPC functional groups.

2.5 Note that conformance with this standard as defined in Section 11 is not enforceable until the complete data structures have been defined for at least one functional group.

NOTICE: This standard does not purport to address safety issues, if any, associated with its use. It is the responsibility of the users of this standard to establish appropriate safety and health practices and determine the applicability of regulatory or other limitations prior to use.

3 Limitations

3.1 The proposed PCS standards will **not** focus on:

- Low-level communication mechanisms (for example, hardware connection schemes, transport layer software, etc.) between PCS functional groups,
- Methodology for implementing actual equipment or factory level control strategies,

- Specifying internal structure or implementation methods of PCS functional groups,
- Capabilities of complementary applications (recipe management, data collection, engineering analysis, data storage, scheduling, change control, ...),
- The management of information and collaboration between functional groups. This is left to implementation,
- Communications within equipment or within factory systems,
- Determination of the proper control algorithm, model, or runtime selectable behavior. It is optional for a PCS function to contain “context matching”,
- The creation and management of Process Control Jobs, or
- Defining test procedures required for validating the standard.

4 Referenced Standards

4.1 SEMI Standards

SEMI E81 — Provisional Specification for CIM Framework Domain Architecture

4.2 Other Sources

The Unified Modeling Language Reference Manual, James Rumbaugh, Ivar Jacobson, Grady Booch, 1999 / 0-201-30988-X / Addison Wesley Professional.

The Unified Modeling Language User Guide, Grady Booch, James Rumbaugh, Ivar Jacobson, 1999 / 0-201-57168-4 / Addison Wesley Professional.

Object-Oriented Modeling and Design, Prentice Hall, Englewood Cliffs, NJ, 1991 by James Rumbaugh, et. al.

Equipment Engineering Capabilities (EEC) Guidelines (Phase 2.0), Version 2.5, International SEMATECH, July 2002.

IEC 61158-5, “Digital data communications for measurement and control – Fieldbus for use in industrial control systems – Part 5: Application layer service definition,” Final Draft International Standard, 2003.

NOTICE: Unless otherwise indicated, all documents cited shall be the latest published versions.

5 Terminology

5.1 Abbreviations & Acronyms

5.1.1 XML — eXtensible Markup Language

5.2 Definitions

5.2.1 *Advanced Process Control (APC)*¹ — the manufacturing discipline for applying control strategies and/or employing analysis and computation mechanisms to recommend optimized machine settings and detect faults and determine their cause.

5.2.2 *Analysis Engine (AE)* — a process that utilizes data and possibly operational instructions to produce a response. In the context of this standard, this term is used to encompass the characteristics common to all PCS functional groups.

5.2.3 *context* — a series of attributes that uniquely identifies a manufacturing entity (e.g., wafer, lot, module, tool, reticle) and its status in the manufacturing operation.

5.2.4 *Context Matching (CM)* — the process of comparing and matching the values of a set of attributes that represent the state of a system (e.g., process, product and equipment) to a set of stored or computed values. This is usually done so that a unique action can be specified by the context matching system.

5.2.5 *Fault Classification (FC)* — the technique of determining the cause of a fault once it has been detected.

5.2.6 *Fault Detection (FD)* — the technique of monitoring and analyzing variations in tool and/or process data to detect anomalies. Fault detection includes both univariate and multivariate statistical analysis techniques.

5.2.7 *Fault Detection and Classification (FDC)* — combination of FD & FC.

5.2.8 *Fault Prediction (or Prognosis) (FP)* — the technique of monitoring and analyzing variations in process data to predict anomalies.

5.2.9 *Functional Group (FG)* — a collection of closely related software capabilities that one would expect to be provided as an integrated product.

5.2.10 *Process Control Job (PCJ)* — a unit of work that could be tracked and have data associated with it. In the context of this standard a process control job is usually associated with a unit of work that a PCS Analysis Engine performs; a unique identifier such as an ID number is often utilized to track the process control job.

¹ APC and PCS terms are used interchangeably in some companies. Both can be seen as the umbrella of components for process control.

5.2.11 *Process Control System (PCS)*¹ — a system capable of performing Process Control, which includes one or more of R2R Control, FD, FC, FP, SPC, or any future Process Control functionality defined in this standard for a PCS functional group.

5.2.12 *Run-to-Run (R2R) Control* — the technique of modifying recipe parameters or the selection of control parameters between runs to improve processing performance. A “run” can be a batch, lot, or an individual wafer.

5.2.13 *Statistical Process Control (SPC)* — the technique of using statistical methods to analyze process or product metrics to take appropriate actions to achieve and maintain a state of statistical control and continuously improve the process capability.

6 Background

6.1 With the development of process control applications at the factory level and inside the equipment, it is now apparent that the lack of interface standards for these applications has inhibited widespread adoption of process control at the factory level. Moreover, adding process control and integrated metrology capabilities inside the tool requires the ability for the factory level applications to collaborate with tool level applications.

7 Conventions

7.1 This section defines conventions followed in this document.

7.2 Object Modeling

7.2.1 *Unified Modeling Language (UML)* — This specification uses UML notation for all class diagrams and for object diagrams provided as examples. No other types of diagrams are used in this specification.

7.2.2 UML class diagrams have clearly defined meanings and are a part of this specification. Detail contained in these diagrams is not always repeated in the text.

7.2.3 *Class Diagrams* — UML Class Charts.

7.2.4 *Name of a Class* — The text capitalizes class names.

7.2.5 *Abstract and Concrete Classes* — Each class is specified as Abstract or Concrete. Abstract classes are not directly implemented. In this specification “implemented” means represented to the factory through the communications interface. All classes defined as concrete may be directly implemented. This specification does not attempt to define equipment control system implementation. In UML class diagrams, abstract class names are shown in italics.

7.2.6 *Class Attribute Definition* — The attributes of a class are defined in table format as illustrated in the table below. Note that the “+” sign in front of attributes in UML class diagrams indicates “public” attributes. All attributes defined in this specification are public.

Table 1 Attribute Table Format

Attribute Name	Definition	Access	Reqd	Form
		RO or RW	Y or N	See list below

7.2.6.1 *Access* — Attributes may be settable (ReadWrite or RW) or not settable (ReadOnly or RO) through the interface services.

7.2.6.2 *Reqd* — Is this attribute required? Y — Yes, or N — No.

7.2.6.3 *Form* — Defines the data type of the attribute. Data types specified in the Process Control Systems specification are described in the subsections that follow. Data types in this specification are high-level definitions and should be mapped to the data types of a specific technology as appropriate. Extensions to the Process Control System interface specifications are not limited to the list of data types defined in this specification. See the definition of “Form” in the SEMI Compilation of Terms for other Form values commonly used in SEMI documents.

7.2.6.3.1 *Enumeration* — A list of named values used as the range of a particular attribute type. In this document, these values are represented as text strings, but may be implemented differently (e.g. as integers that correspond to the named values).

7.2.6.3.2 *String* — A text string. Limitations on length are left to the implementation technology unless otherwise specified in this document.

7.2.6.4 *Constraints*

7.2.6.4.1 Constraints in UML are shown with dotted lines, typically with a note attached to explain the constraint. Constraints shown in this way tend to clutter large UML diagrams. Therefore, in this specification, constraints are shown only on the diagrams focusing on individual classes.

7.2.7 *UML Associations*

7.2.7.1 The mechanism used for representing associations in UML between classes is implementation dependent. This document is abstract in nature, and does not specify or imply any such mechanism. Any adjunct standard that provides an implementation of this specification must include a description of the mechanism used for representing UML associations used in this document.

7.2.8 *Class Association Table* — This table provides an example of the tables used to list and describe associations between classes defined in this specification.

Table 2 Association Table Format

<i>Association Role Name</i>	<i>Role Definition</i>	<i>Comments</i>

7.2.8.1 *Association Role Name* — The name of the association role being defined.

7.2.8.2 *Role Definition* — Describes the function or purpose of the association.

7.2.8.3 *Comments* — Any additional comments or notes regarding the association.

7.2.9 *Method Definition* — The methods (in other words, the messages that comprise the external interface to an object) supported by a PCS functional group are defined in table format as illustrated below.

Table 3 Method Definition Table Format

#	<i>Method</i>	<i>Arguments</i>	<i>Returns</i>	<i>Requirement</i>	<i>Behavior/Comment</i>
				Y or N	

7.2.9.1 *Method* — Method name.

7.2.9.2 *Arguments* — Input parameters.

7.2.9.3 *Returns* — Output parameters.

7.2.9.4 *Requirement* — Is this method required of a compliant implementation? Y – Yes, or N – No.

7.2.9.5 *Behavior/Comment* — Summary description of the method's purpose and behavior, explanation of expected system response to this method, etc.

7.3 *Sequence Behavior*

7.3.1 Sequence behavior description is specified in the form of UML Sequence Diagrams.

7.4 *State Behavior*

7.4.1 State behavior description is specified in the form of UML State Charts.

8 Overview

8.1 Using the simple definition of a Functional Group given in the Terminology section, the initial set of PCS functional groups has been identified. To identify the proper scope (and therefore the boundaries) for these groups, the following criteria are applied to test whether or not a given set of capabilities constitutes a valid PCS functional group. Specifically, a functional group should satisfy one or more of the following requirements:

- Support a coherent set of tasks
- Share common set of data and must use it under similar constraints (performance, retention, etc.)
- Internal objects are tightly coupled
- Legacy implementation/integration constraints exist (for example, the capabilities are inseparable because an external system must interact with them as an integrated whole)
- Strong associations exist in the market (that is, one would expect these capabilities to be provided as an integrated product)

8.2 Based on these criteria, and the stated implementation priorities of the semiconductor manufacturers and suppliers involved on the task force, the following initial PCS functional groups have been identified:

- R2R (Run-to-Run) Control
- FD (Fault Detection)
- FC (Fault Classification)
- FP (Fault Prediction)
- SPC (Statistical Process Control)

8.3 Note that it is understood that other PCS functional groups can be added as needed. Future versions of the document will describe procedures for adding a new group.

8.4 The aforementioned groups can be implemented at the tool or factory level or both. As a requirement, functional group interfaces should be defined in a consistent way. The following diagram represents the proposed functional groups and an example of their interactions within the factory and tool environments.

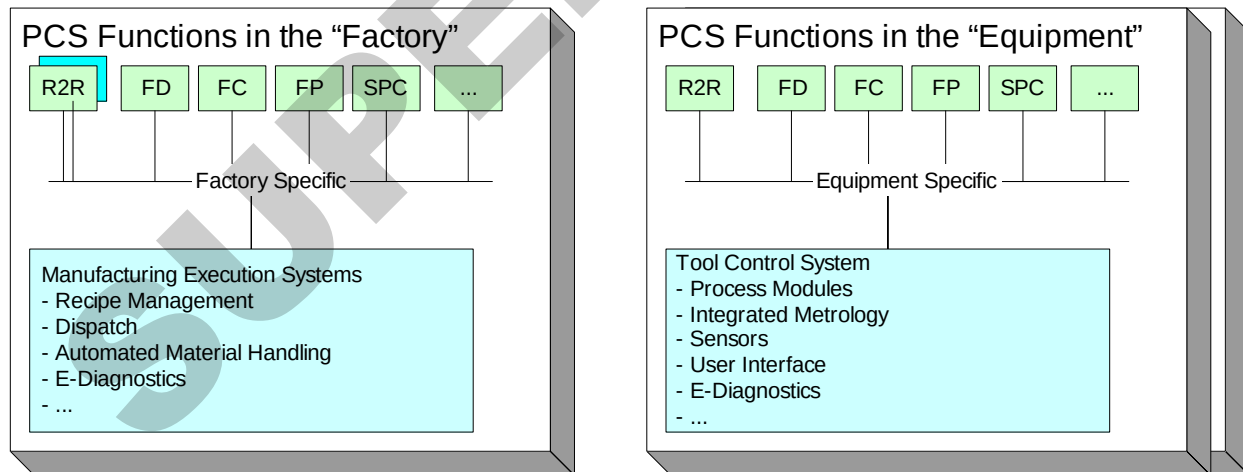


Figure 1
PCS Functions

8.4.1 Process Control Systems may consist of a variety of capabilities or functions at both the factory and equipment level as shown in Figure 1 below. Moreover, this may include multiple instances of the same function type from different suppliers. In the example above, several run-to-run controllers are running at the factory level with differences that are only visible as internal special capabilities. This interface standard will facilitate easier

collaboration between the tool level functions and factory level systems independent of the supplier that developed the internal function's capabilities.

9 Process Control Systems Functional Group Capabilities and Interface Data Structures

9.1 Functional Group Partitioning

9.1.1 Process control systems are partitioned into functional groups using the criteria defined in Section 8. This standard defines the interface data structures, services and behavior associated with these functional groups. In this section the capabilities of the PCS functional groups are defined in more detail. First of all, the basic capabilities common to all functional groups are presented. Then the additional basic capabilities unique to each functional group are presented. A high level description of the specifications of data structures, services and behavior are described in Section 10.

9.1.2 It is important to note that the specific implementations of these various functional groups may differ greatly in the level of capability delivered, performance, quality of service, and internal architecture. Moreover, in a given factory, there may be multiple instances of the same functional group running concurrently, from one or more suppliers. Some functional groups may be completely independent of other PCS functional groups (in other words, they can operate in relative isolation, depending only on information from systems external to PCS); others may depend heavily on other functional groups, collaborating closely to provide a higher level of capability than they could on their own.

9.1.3 The goal of the PCS interface standards is to enable all of these implementation scenarios (and more) without dictating internal product architectures.

9.2 PCS Analysis Engine

9.2.1 As defined in Section 8, all functional groups can be thought of as specializations of a PCS Analysis Engine and therefore inherit the structure and behavior of a PCS Analysis Engine. This section defines the common capabilities of a PCS Analysis Engine.

9.2.2 General Capability Description

9.2.2.1 All PCS Analysis Engines shall have inputs and outputs as shown in Figure 2. This means that the PCS Analysis Engines shall support the consumption of the specified inputs and production of the specified outputs as necessary to achieve the required capabilities of that analysis engine as specified in this standard. These inputs and outputs are specified as aggregate data structures. A definition of these inputs and outputs is provided in Tables 4 and 5, respectively. The PCS Analysis Engine shall have the general behavior of utilizing specified inputs, performing tasks based to some degree on these inputs, and generating outputs. Any additional structure and behavior of various PCS Analysis Engines are outlined in the remainder of this section and specified in detail in Section 10.4.

9.2.2.2 Note that, in addition to limitations of specification defined in Section 4, the following are beyond the scope of definition of the PCS Analysis Engine:

- the level of synchronization of inputs with performance of tasks and/or generation of outputs, and
- the specification of any internal states of the PCS Analysis Engine.

9.2.2.3 All PCS Analysis Engines belonging to this functional group shall have the capabilities listed in Table 6.

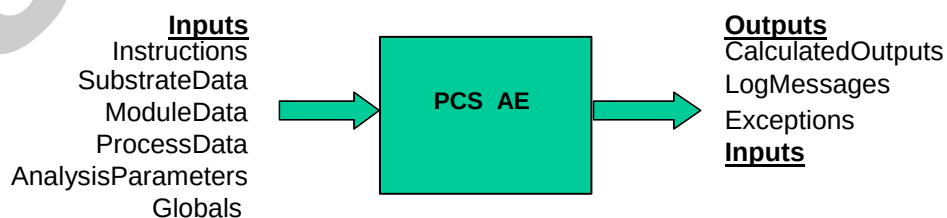


Figure 2
PCS Analysis Engine Common Input/Output Structure

9.2.2.4 Note that based on the simple data model shown in Figure 2, Inputs are available as an Output from an Analysis Engine. Data provided as input to an Analysis Engine should be accessible as output from the Analysis Engine for historical purposes.

Table 4 PCS Analysis Engine Inputs Definitions

<i>Name</i>	<i>Definition</i>	<i>Comments</i>
Instructions	Data consisting of computational commands/services, data collection plans, algorithms, and models, used by an Analysis Engine.	An Instruction could be a specific service to be performed by the Analysis Engine.
SubstrateData	Information about the substrate being associated with the Analysis Engine. This could include data representing a single wafer, multiple wafers, or one or more lots, as well as attributes of the material.	Profile or topology data, and other characteristics measured physically or electrically.
ModuleData	Information representing the current or last known physical state of a process module that will be used by an Analysis Engine to process material.	Machine, or module E10 state, RF hours, number of wafers processed, etc.
ProcessData	Information representing the process itself, such as the name and type of the process and the current or last known process state based on measurements from sensors of the processing environment.	Name of the process (CMP), in-situ process measurements, trace data, and summary statistics etc.
AnalysisParameters	Information that identifies numeric data used by the Analysis Engine to perform various calculations.	Equation coefficients, feedforward data, etc.
Globals	Information that applies globally across an entire job.	A lot id, for example could be specified once as a global so that it would not have to be specified with many other data sets.

Table 5 PCS Analysis Engine Outputs Definitions

<i>Name</i>	<i>Definition</i>	<i>Comments</i>
CalculatedOutputs	Data representing the results of analysis from an Analysis Engine.	A calculation derived from various inputs to the Analysis Engine indicating wafer state, process state or model state information, recommended machine settings, etc.
Instructions	Data consisting of computational commands/services, data collection plans, algorithms, and models, used by an Analysis Engine.	An Instruction could be a specific service to be performed by the Analysis Engine.
SubstrateData	Information about the substrate being associated with the Analysis Engine. This could include data representing a single wafer, multiple wafers, or one or more lots, as well as attributes of the material.	Profile or topology data, and other characteristics measured physically or electrically.
ModuleData	Information representing the current or last known physical state of a process module that will be used by an Analysis Engine to process material.	Machine, or module E10 state, RF hours, number of wafers processed, etc.
ProcessData	Information representing the process itself, such as the name and type of the process and the current or last known process state based on measurements from sensors of the processing environment.	Name of the process (say, CMP), in-situ process measurements, trace data, and summary statistics etc.
AnalysisParameters	Information that identifies numeric data used by the Analysis Engine to perform various calculations.	Equation coefficients, feedforward data, etc.
Globals	Information that applies globally across an entire job.	A lot id, for example could be specified once as a Global so that it would not have to be specified with many other data sets.
Exception	Data representing erroneous conditions resulting from an Analysis Data describing the activity of an Analysis Engine as it executes its tasks.	A status based on constraints placed on the calculations, indicating calculations are out of bounds, including level of severity. A text message describing the activities of the Analysis Engine.