

# Introduction into IEC 61131-8

## Guidelines for the application and implementation of programming languages

(From the Working Draft 3)

### **Scope**

This technical report applies to the programming of programmable controller systems using the programming languages defined in IEC 61131-3. It also provides guidelines for the implementation of these languages in programmable controller systems and their programming support environments (PSEs).

IEC 61131-4 should be consulted for other aspects of the application of programmable controller systems.

### **Overview**

The intended audience for this technical report consists of:

- users of programmable controller systems as defined in IEC 61131-3, who must program, configure, install, and maintain programmable controllers as part of industrial process measurement and control systems; and
- implementors of programming languages, as defined in IEC 61131-3, for programmable controller systems. This may include vendors of software and hardware for the preparation and maintenance of programs for these systems, as well as vendors of the programmable controller systems themselves.

IEC 61131-3, in contrast, is mainly oriented toward the implementors of programming languages for programmable controllers. Users who wish a general introduction to these languages and their application should consult any of several generally available textbooks on this subject. Subclause 1.4 of IEC 61131-3 should be consulted by those who wish a "top-down" overview of the contents of IEC 61131-3.

Clause 2 of this technical report provides a general introduction to IEC 61131-3, while clause 3 provides complementary information about the application of some of the programming language elements specified IEC 61131-3. Clause 4 provides information about the intended implementation of some of these programming language elements, while Clause 5 provides general information about requirements for hardware and software for program development and maintenance. Hence, it is expected that users of programmable controllers will find Clauses 2 and 3 of this document most useful, while programming language implementors will find Clauses 4 and 5 more useful, referring to the background material in Clauses 2 and 3 as necessary.

## TABLE OF CONTENT IEC 61131-8

|         |   |    |
|---------|---|----|
| 1.      | General   | 7  |
| 1.1     | Scope   | 7  |
| 1.2     | Normative references  | 7  |
| 1.3     | Overview  | 7  |
| 2.      | Introduction to IEC 61131-3   | 8  |
| 2.1     | General considerations  | 8  |
| 2.2     | Overcoming historical limitations                                   | 10 |
| 2.3     | Basic features in IEC 61131-3                                       | 11 |
| 2.4     | New features in the second edition of IEC 61131-3                   | 12 |
| 2.5     | Software engineering considerations                                 | 12 |
| 2.5.1   | Application of software engineering principles                      | 12 |
| 2.5.1.1 | Encapsulation and hiding  | 12 |
| 2.5.1.2 | Explicit representation of state                                    | 13 |
| 2.5.1.3 | Mapping to the application domain                                   | 13 |
| 2.5.1.4 | Mapping of design to implementation                                 | 13 |
| 2.5.1.5 | Structured programming  | 14 |
| 2.5.1.6 | Software reuse  | 14 |
| 2.5.2   | Portability   | 15 |
| 2.5.2.1 | Inter-language portability  | 15 |
| 2.5.2.2 | Inter-system portability  | 15 |
| 3.      | Application guidelines  | 16 |
| 3.1     | Use of data types   | 16 |
| 3.1.1   | Type vs. variable initialization                                    | 16 |
| 3.1.2   | Use of enumerated and subrange types                                | 17 |
| 3.1.3   | Use of BCD data   | 17 |
| 3.1.4   | Use of REAL data types  | 19 |
| 3.1.5   | Use of character string data types                                  | 19 |
| 3.1.6   | Use of time data types  | 20 |
| 3.1.7   | Use of multi-element variables                                      | 21 |
| 3.1.8   | Use of bit string data types  | 21 |
| 3.2     | Data passing  | 21 |
| 3.2.1   | Global and external variables                                       | 22 |
| 3.2.2   | In-out (VAR_IN_OUT) variables                                       | 23 |
| 3.2.3   | Formal and non-formal invocations and argument lists                | 25 |
| 3.3     | Use of function blocks  | 29 |
| 3.3.1   | Function block types and instances                                  | 29 |
| 3.3.2   | Scope of data within function blocks                                | 30 |
| 3.3.3   | Function block access and invocation                                | 31 |
| 3.4     | Differences between function block instances and functions          | 32 |
| 3.5     | Use of indirectly referenced function block instances               | 32 |
| 3.5.1   | Establishing an indirect function block instance reference          | 33 |
| 3.5.2   | Access to indirectly referenced function block instances            | 35 |
| 3.5.3   | Invocation of indirectly referenced function block instances        | 35 |
| 3.5.4   | Recursion of indirectly referenced function block instances         | 38 |
| 3.5.5   | Execution control of indirectly referenced function block instances | 38 |
| 3.5.6   | Use of indirectly referenced function block instances in functions  | 38 |
| 3.6     | Recursion within programmable controller programming languages      | 39 |
| 3.7     | Single and multiple invocation                                      | 39 |
| 3.8     | Language specific features  | 40 |

|          |  |    |
|----------|--|----|
| 3.8.1    | Edge triggered functionality                               | 40 |
| 3.8.1.1  | Edge triggering in LD language                             | 40 |
| 3.8.1.2  | Use of edge triggered function blocks                      | 41 |
| 3.8.2    | Use of EN/ENO in functions and function blocks             | 42 |
| 3.8.3    | Use of non-IEC 61131-3 languages                           | 43 |
| 3.9      | Use of SFC elements  | 43 |
| 3.9.1    | Action control   | 44 |
| 3.9.2    | Boolean actions  | 45 |
| 3.9.3    | Non-SFC actions  | 50 |
| 3.9.4    | SFC actions  | 51 |
| 3.9.5    | SFC function blocks  | 53 |
| 3.9.6    | "Indicator" variables                                      | 53 |
| 3.10     | Scheduling, concurrency, and synchronization mechanisms    | 54 |
| 3.10.1   | Operating system issues                                    | 54 |
| 3.10.2   | Task scheduling  | 56 |
| 3.10.2.1 | Performance effects  | 57 |
| 3.10.2.2 | Concurrency effects  | 57 |
| 3.10.3   | Semaphores   | 57 |
| 3.10.3.1 | General  | 57 |
| 3.10.3.2 | Deadlocks  | 58 |
| 3.10.4   | Messaging  | 58 |
| 3.10.4.1 | Global storage   | 59 |
| 3.10.4.2 | Mailboxes and queues                                       | 59 |
| 3.10.5   | Time stamping  | 59 |
| 3.11     | Communication facilities in ISO/IEC 9506/5 and IEC 61131-5 | 59 |
| 3.11.1   | Communication channels                                     | 60 |
| 3.11.2   | Reading and writing variables                              | 60 |
| 3.11.3   | Communication function blocks                              | 60 |
| 3.12     | Recommended programming practices                          | 61 |
| 3.12.1   | Global variables   | 62 |
| 3.12.2   | Jumps in Function Block Diagram (FBD) language             | 62 |
| 3.12.3   | Multiple invocations of function block instances in FBD    | 62 |
| 3.12.4   | Coupling of Sequential Function Chart (SFC) networks       | 62 |
| 3.12.5   | Dynamic modification of task priorities                    | 63 |
| 3.12.6   | Execution control of function block instances by tasks     | 63 |
| 3.12.7   | Use of RTC (Real Time Clock) Function Blocks               | 63 |
| 4.       | Implementation guidelines                                  | 64 |
| 4.1      | Resource allocation  | 64 |
| 4.2      | Implementation of data types                               | 64 |
| 4.2.1    | REAL and LREAL data types                                  | 64 |
| 4.2.2    | Character strings  | 64 |
| 4.2.3    | Time data types  | 65 |
| 4.2.4    | Multi-element variables                                    | 65 |
| 4.3      | Execution of functions and function blocks                 | 65 |
| 4.3.1    | Functions  | 66 |
| 4.3.2    | Function blocks  | 66 |
| 4.4      | Implementation of Sequential Function Charts (SFCs)        | 67 |
| 4.5      | Task scheduling  | 67 |
| 4.5.1    | Classification of tasks                                    | 68 |
| 4.5.2    | Task priorities  | 68 |
| 4.6      | Error handling   | 69 |
| 4.6.1    | Error handling mechanisms                                  | 69 |
| 4.6.2    | Run-time error handling procedures                         | 71 |
| 4.6.2.1  | Reporting of errors  | 71 |
| 4.6.2.2  | System-defined error handling procedures                   | 71 |

|          |   |     |
|----------|---|-----|
| 4.6.2.3  | User-defined error handling procedures                                | 73  |
| 4.7      | System interface  | 73  |
| 4.8      | Compliance  | 73  |
| 4.8.1    | Compliance statement  | 73  |
| 4.8.2    | Controller instruction sets   | 73  |
| 4.8.3    | Compliance testing  | 74  |
| 4.9      | Compatibility with IEC 617-12, 617-13, and 848                        | 74  |
| 5.       | Programming support environment (PSE) requirements                    | 74  |
| 5.1      | User interface  | 74  |
| 5.2      | Programming of programs, functions and function blocks                | 75  |
| 5.3      | Application design and configuration                                  | 76  |
| 5.4      | Separate compilation  | 76  |
| 5.5      | Separation of interface and body                                      | 77  |
| 5.5.1    | Invocation of a function from a programming unit                      | 77  |
| 5.5.2    | Declaration and invocation of a function block instance               | 78  |
| 5.6      | Linking of configuration elements with programs                       | 79  |
| 5.7      | Library management  | 82  |
| 5.8      | Analysis tools  | 83  |
| 5.8.1    | Simulation and debugging  | 83  |
| 5.8.2    | Performance estimation  | 83  |
| 5.8.3    | Feedback loop analysis  | 83  |
| 5.8.4    | SFC analysis  | 83  |
| 5.9      | Documentation requirements  | 86  |
| 5.10     | Security of data and programs   | 87  |
| 5.11     | On-line facilities  | 87  |
| Annex A  | Changes to IEC 61131-3 2nd Edition                                    | 88  |
| A.1      | Reasons for the 2nd edition of part 3                                 | 88  |
| A.2      | Corrigendum   | 88  |
| A.3      | Amendment   | 89  |
| A.3.1    | Numeric literals (2.2.1) - typed literals                             | 90  |
| A.3.2    | Elementary data types (2.3.1) - double-byte strings                   | 90  |
| A.3.3    | Derived data types (2.3.3) - enumerated data types                    | 90  |
| A.3.4    | Single element variables (2.4.1.1) - 'wild-card' direct addresses     | 91  |
| A.3.5    | Declaration (2.4.3) - Temporary variables                             | 91  |
| A.3.6    | Type assignment (2.4.3.1) - RETAIN and NON_RETAIN Variable attributes | 92  |
| A.3.7 of | Function ( 2.5.1) - Use EN/ENO  | 92  |
| A.3.8    | Declaration (2.5.1.3) - Function invocation with VAR_IN_OUT           | 92  |
| A.3.9    | Type conversion functions (2.5.1.5.1)                                 | 93  |
| A.3.10   | Functions of time data types (2.5.1.5.6)                              | 93  |
| A.3.11   | Function blocks (2.5.2) - Extended initialisation facilities          | 93  |
| A.3.12   | Pulse action qualifiers (2.6.4.4)                                     | 94  |
| A.3.13   | Action control (2.6.4.5)  | 94  |
| A.3.14   | Configuration initialisation (2.7.1)                                  | 94  |
| A.3.15   | Instruction List (3.2)  | 95  |
| A.3.16   | Formal specification of language elements (Annex B)                   | 97  |
| A.3.17   | Further amendments  | 97  |
| ANNEX B  | SOFTWARE QUALITY MEASURES   | 99  |
| ANNEX C  | INDEX   | 101 |