Towards Standardising PID Controllers in the Process Industry

Margret Bauer*, Emil Sundström, José Luis Guzman***, Tore Hägglund** and Kristian Soltesz****

*Hamburg University of Applied Sciences, Germany (Margret.bauer@haw-hamburg.de)
**Department of Automatic Control, Lund University, Sweden (emil.sundstrom@control.lth.se, tore.hagglund@control.lth.se, Kristian.soltesz@control.lth.se)
*** University of Almería, Department of Informatics, ceiA3, (joguzman@ual.es)

Abstract: The current situation relating to PID control in the process industry is highly unsatisfactory. Every process control system (DCS, PLC, SCADA, etc.) vendor has hidden their PID code in a function block. There is very little documentation, often consisting of only simple continuous time equations, sometimes using the Laplace transform. Editing or augmenting the code is not possible. As a result, many control systems users code their own PID implementation if they want to build a structure around the PID function block. This paper addresses the situation by discussing recent standardization efforts by industry bodies in the context of a recent publication of a reference PID implementation published by the authors.

Keywords: PID control, implementation, numerical algorithm, process control system, process automation, standardisation.

1. INTRODUCTION

Recently, the PID has received renewed attention by standardization bodies and academic researchers alike (Hägglund and Guzman, 2024). The attention is driven by the fact that the actual PID controller is sold by an automation vendor who hides it in graphical programming building blocks called 'function blocks'. The PID function block is then configured, setting inputs, outputs and parameters, but cannot be accessed or altered.

The concept of a function block is intuitive and useful, particularly because there are usually hundreds if not thousands of instances where the PID block is used. While the PID controller works in this context, there are situations where you would like to access the code even when you have a fully functioning PID controller. These include the following:

- Understanding the workings of the controller and the individual P-, I- and D-parts;
- Troubleshooting problems when getting unexpected control action;
- Implementing advanced control structures such as ratio control, feedforward and further control logic;
- Switching from one control system vendor to another.

In the current situation, the vendor's function blocks do not allow for any interaction with the code. As a result, it is impossible to understand or reproduce the function block output. While some vendors provide detailed user manuals, these will only give a general transfer function of the PID controller, not the code line by line. As a result, automation software users often program their own PID function despite the fact that there is an existing function block. However, this come with its own hazards as important features may not be included in the code.

It is not in the interest of automation vendors to disclose any information on the PID implementation or work towards a unified implementation that is not their own. Only in recent years, two standardization bodies, the International Society of Automation (ISA) and the German Process Automation User Association NAMUR have published technical reports that aim at providing a standard PID controller. These are not norms or standards.

One key advantage of the PID controller is that it is very simple to implement. However, the devil is in the detail. For example, there are different implementation forms, specifically the positional and the velocity form. Both are described by the authors in Sundström et al. (2026), presenting a reference implementation of the PID controller for the first time. Even with a polished and finished reference implementation available, the road to standardisation is still long. There is more to PID in the process industry than the algorithm alone.

The purpose of this paper is revisit the reference implementation in the context of process control systems to drive the standardization further. In addition, it is put side by side with the recent standardisation efforts of the ISA and NAMUR. The final goal – which will not be achieved here – is to standardize the PID controller so that automation user companies can easily switch between automation vendors, to facilitate easy building of advanced control structures and to allow engineers insight into the PID controller workings.

Section 2 explains how the PID controller is embedded in the different aspects of the process control system, using ABB's control system 800xA as an example. Section 3 gives an overview of recent standardization efforts by ISA and NAMUR. Section 4 compares the reference implementation published by the authors to the proposed standards.

2. BACKGROUND: PID IN THE PROCESS CONTROL SYSTEM

The PID controller in the process industry has many aspects because of the integration into production and the resulting safety and reliability requirements. The PID controller is

therefore a complex software object required in two environments, namely in the

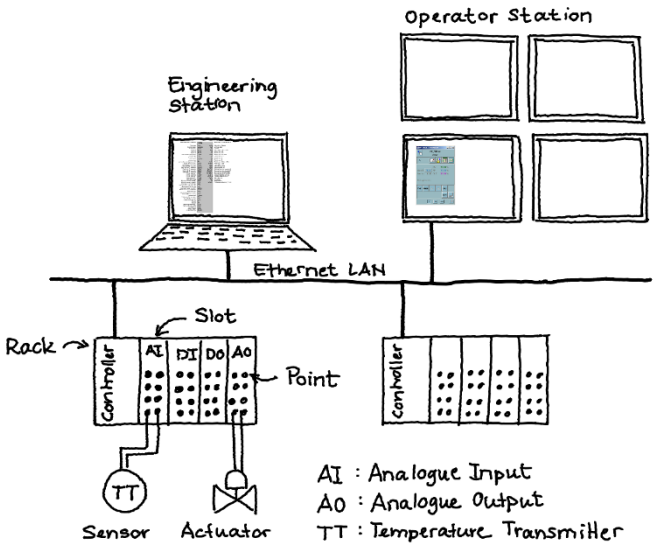

Figure 1. Architecture of a distrusted control system (DCS) in the process industry.

- Engineering station
- Operator station

The term 'engineering' in 'engineering station' can be misleading, as it does not relate to engineering in general, but specifically to automation engineering which is the configuration of the process control system.

Fig. 1 shows the architecture of a typical process control system or distributed control system. The PID controller is configured in the engineering station and executed on the controller. During operation, a faceplate on the operating station visualizes the current value of the four signals related to the PID controller: process variable, setpoint, controller output and mode (automatic or manual).

2.1 PID in the engineering station context

The IEC61131-3 standard prescribes the format in which the engineering station should be programmed. After the discontinuation of the instruction list, there are now four languages:

- Structured text (ST)
- Function block diagrams (FBD)
- Ladder diagrams (LD)
- Sequential function charts (SFC)

While ST is a text-based programming language, FBD, LD and SFC are graphical configuration tools. However, it is possible to integrate ST within a FBD and it is also possible to translate from more modern programming languages such as Python to ST (Zhao et al., 2024).

The PID controller is implemented in a function block (FB) in the control system libraries. Often, there are several versions of PID controllers within one control system due to historical reasons. If engineering station users are dissatisfied with the functioning or flexibility of the predefined function block, they can implement their own version of a PID controller in ST.

Table 1. In- and outputs of the PID function block.

I/O	Description
PV	Process variable, measurement, sometimes referred to as controlled variable (CV) or measurement value (MV)
SP	Setpoint or reference value
OP	Controller output
FF	Feedforward input signal
mode	The controller can be in automatic mode (AUTO), manual mode (MAN) or tracking mode where it follows an external signal. There may be several tracking modes for different applications.
uman	Controller output when the controller is in manual mode
utrack	Controller output of an external tracking signal when the controller is in tracking mode.

An example of a PID function block is shown in Fig 2. The main function block is labelled FC101, indicating that it represents a flow controller. The in- and outputs are defined in Table 1.

The PID block receives its input signal PV from the measurement signal which is passed through an analogue input (AI) signal block from the configured measurement input address R01.S04.P06. The address reflects the physical connection of Rack 01, Slot 04 and Point 06. The other inputs – setpoint, mode and uman – are received from the operator interface. The output of the PID controller is passed to an analogue output (AO) block to the designated actuator address R01.S05.P02.

2.2 PID in the operator station context

The operator interacts with the PID controller through a faceplate. Generally, a process flow diagram represents the most important equipment and measurement of the controlled process. A faceplate is a separate window that opens when the operator double-clicks a controlled measurement value indicated on the process flow diagram. The faceplate in Fig. 3 shows the measurement or process variable (PV) on the same scale as the setpoint or reference value (SP). In automatic mode, the setpoint SP can be adjusted through arrow buttons or via entering a value. In manual mode, the controller output OP can be adjusted through the same buttons.

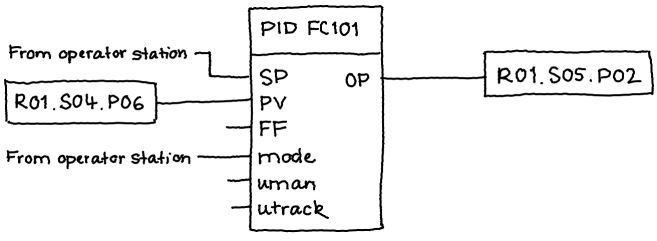

Figure 2. Example of a basic function block diagram (FBD) for implementing a single PID controller.

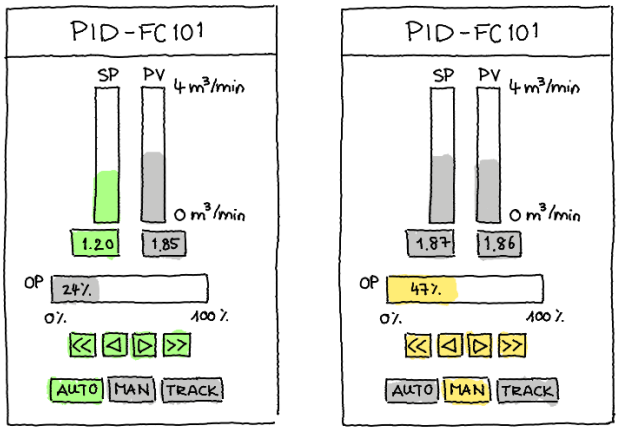

Figure 3. Faceplate of a PID controller when the controller is in automatic mode (left) and when it is in manual mode (right).

ISA101 is the standard that defines human machine interfaces (HMIs) for process automation systems. Its emphasis lies on safety, usability and efficiency and includes recommendations for PID faceplates.

2.3 Example: PID control within ABB's 800xA

ABB was chosen as an example control system provider because its systems are widely found within the Nordic countries, having been partly developed by ABB in Västeras, Sweden. In addition, ABB provides detailed information in publicly available user manuals (Persson, 2023, and ABB, 2013).

ABB's flagship distributed control system is called 800xA. It was introduced in 2004 and evolved from previous control systems Master, Advant and Symphony/Harmony. The letters xA stand for 'Extended Automation'. The standard hardware controller, that is the physical computer, is called AC 800M. The operating station is called 'Operator Workplace', the engineering station is 'Engineering Workplace' or previously Control Builder M. 800xA is object oriented and as a result, the PID controller is an object.

In 800xA, the main PID object, that is, the function block type, is called PID01A. It provides engineering and operator interfaces, alarms and event handling, interlocks, as well as the control algorithm. PID01A has 52 possible inputs, 29 outputs, 37 input parameters and 44 output parameters, highlighting the complexity of the block, see Fig. 4. In comparison, the basic function block described in Fig. 1 has six inputs and one output. There are six control modes: MAN and AUTO as well as modes that allow an external setpoint. A further mode is the non-regulatory control mode, where the setpoint is derived from program logic and directly applied to the output. Balance control mode is used to force the controller output to a value present at the input terminal BalRef. The operator cannot override this mode.

Fig. 5 gives the parameterisation block and the faceplate of PID01A.

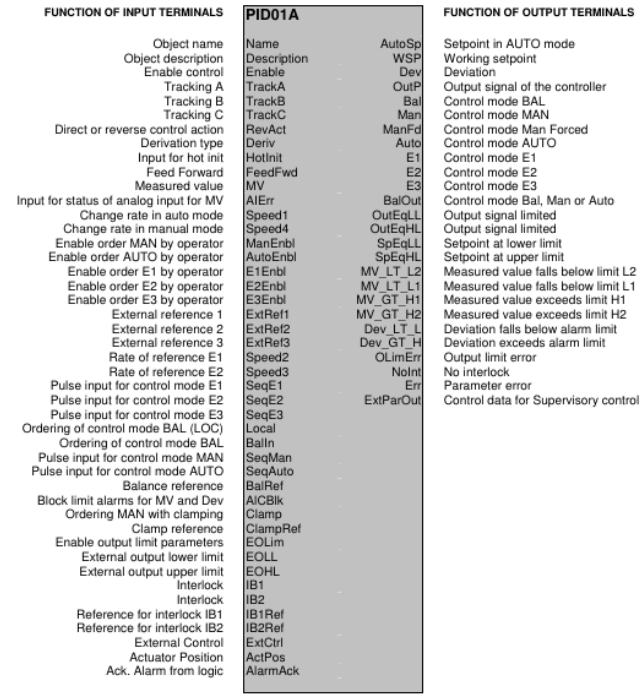

Figure 4. Function block type of PID01A with inputs and outputs.

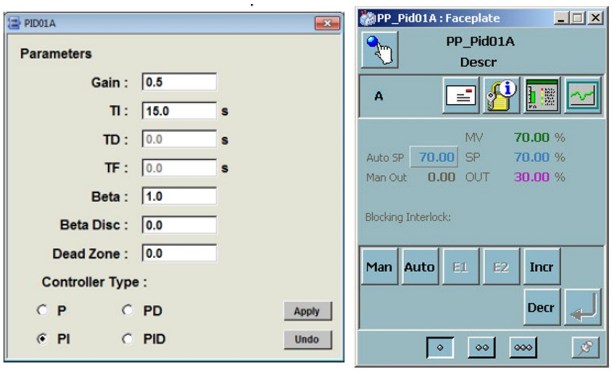

Figure 5. Parameterisation window (left) and simple faceplate (right) of the PID01A control function block.

The actual PID algorithm is contained in a function block called PidCC. The block is often embedded within the higher-level function block PID01A, but can also be used separately, too. Fig. 6 from the user manual describes the algorithm. The P-part is the middle path and includes a setpoint weight β. The I-part is the bottom path and the top path represents the D-part which is applied to the PV and not the controller error and is filtered.

While this representation explains the main algorithmic use, it lacks in detail, as it does not e.g. reflect the implementation for a P-only controller which would include a bias. It gives no indication about whether the positional or the velocity form is used. In short, the details of the algorithmic implementation is hidden from the user.

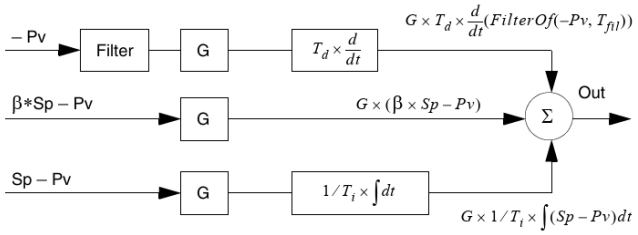

Figure 6. PID algorithm as documented in ABB's 800xA PID01A controller, contained in the PidCC function block, (ABB, 2013).

3. PID STANDARDS

Section 2 has highlighted that standardising the PID controller is much more complex than standardising the PID algorithm alone. However, the core of any PID controller is the algorithm and the standardization effort should include this. This section compares the different efforts that have been made to standardize the PID algorithm by the International Society of Automation (ISA) and the German Process Automation User Association NAMUR.

Both standardization bodies, ISA and NAMUR, have not released a standard. The IEC has issued a standard on PID performance, but not on the PID controller itself. A further IEC norm (IEC 60050-351) standardizes the vocabulary used in control technology in general. Different names and letters for PV, OP and SP are used in different languages which does not assist in a global standardization effort.

3.1 ISA 5.9 Technical Report

The ISA technical report documents common PID algorithms used in industrial control systems, including the various algorithms' implementation methods, options and performance measurements. The authors note that a standard describing fundamentals, terminology, best practices and special functions does not exist. The purpose of the technical report lays the foundation for that standard. The technical report is 167 pages long and thus constitutes a significant source of information.

The report distinguishes between the position(al) form and the velocity form of the algorithm as well as between parallel and series form. It includes a thorough discussion of the following implementation details:

- Anti-windup
- Output tracking
- Integral tracking
- Feed-forward
- PID tuning parameter equivalence and conversion
- External reset feedback (tracking) for cascade and advanced control
- Slow actuators (valve or variable frequency drive)
- PID performance

Fig. 7 depicts the velocity algorithm implementation, one of many versions mentioned in the ISA technical report. The velocity implementation uses the delayed error signal (value last execution cycle) for the P-part and a further delay (rate of change last execution cycle) for the D-part. Note that here, the

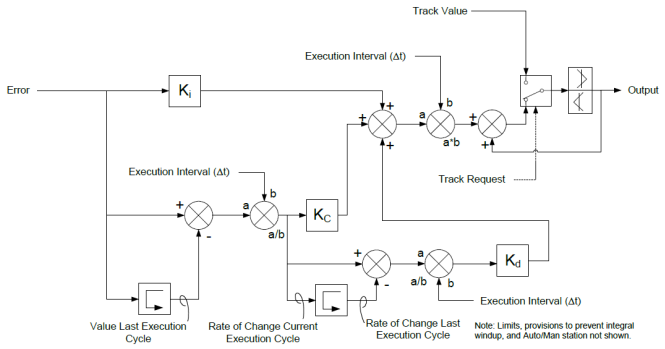

Figure 7. Velocity algorithm implementation as described in the ISA 5.9 technical report.

D-part is executed on the error signal and not the negative process variable in this particular implementation. Other implementations described in ISA 5.9 are executed on the negative process variable.

3.2 NAMUR Standard-PID-Controller

In comparison to the ISA technical report, the NAMUR position paper is relatively short (6 pages). It defines the inputs and parameters as well as outputs and provides a transfer behaviour. The PID controller is defined in the Laplace domain as

$$Y = RD\,\frac{OP_{range}}{PV_{range}}\,GAIN\left(G_e\left(W - \frac{X}{1+\tau s}\right) - G_x\,\frac{X}{1+\tau s}\right)$$

where

$$G_e = P_{on}\left(1 - P_{fb}\right) + I_{on}\,\frac{1}{T_I s} + D_{on}\left(1 - D_{fb}\right)\frac{T_D s}{1 + \frac{T_D}{\alpha}s}$$

and

$$G_x = P_{on}P_{fb} + D_{on}D_{fb}\,\frac{T_D s}{1+\frac{T_D}{\alpha}s}\,.$$

Here, the variables are named Y=OP, X=PV and W=SP, based on the German version of IEC 60050-351. The process variable is filtered with time constant τ. Only the process variable acts to form the derivative part, which is additionally filtered with time constant T_D/α. The tuning parameters are *GAIN*, T_I and T_D. Boolean parameters P_{on}, I_{on} and D_{on} specify whether P, I and D are used.

The modes defined in the NAMUR position paper distinguishes between AUTO, MAN and CASCADE modes. The basic functionality includes

- Automatic reaction to bad inputs
- Anti-Windup
- Bumpless variation of GAIN during PID operation
- First-order lag for PV filter
- Setpoint ramp
- Feed-forward
- Split-range / valve linearization
- Gain scheduling

The working group is currently discussing an updated version of the position paper.

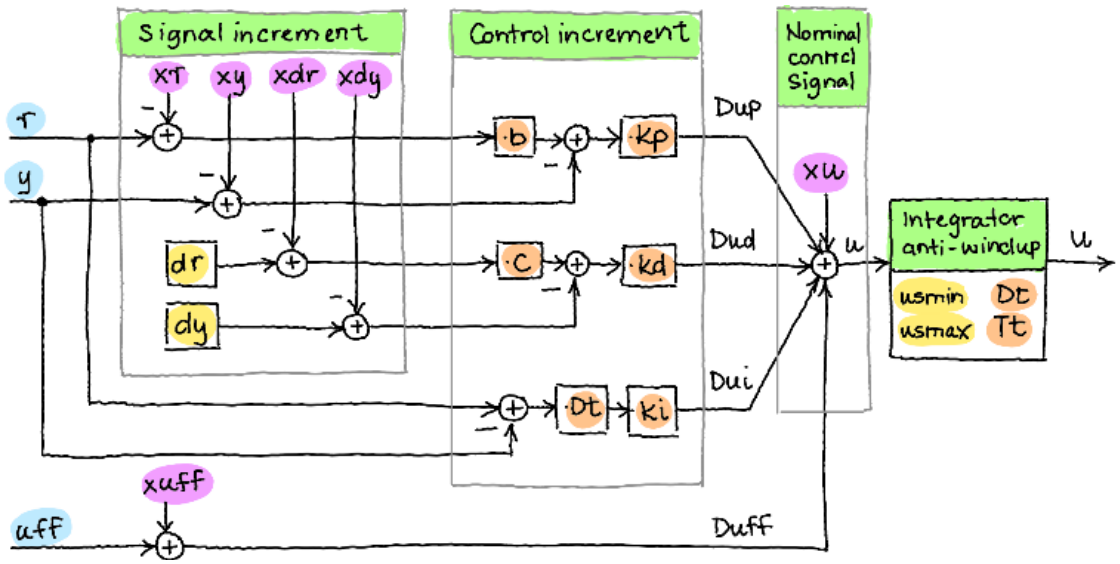

Figure 8. Schematic representation of the PID controller with integral action as described in Sundström et al., 2026. The code is freely available.

4. PID REFERENCE IMPLEMENTATION

While the ISA 5.9 technical report has listed many variations of the PID controller, the NAMUR working group has decided to focus only on one version. The NAMUR working group has provided the general form of the PID controller, focusing on the definition of the input and output variables. As both documents are pre-norm or pre-standard report, the work is necessarily incomplete. Both documents do not yet give any details regarding the impact on the engineering and operator stations.

A recent publication by the authors of this paper has presented a PID control algorithm that is suitable for all situations. The proposed algorithm includes

- Control with and without integral action
- Setpoint handling
- Feed-forward control
- Controller output limitation
- Anti-windup
- External setpoint tracking
- Bumpless mode changes and parameter changes
- Sampling rate variations and jitter
- Process variable and setpoint filtering

The algorithm combines positional and velocity form. For PID control including integral action (PI- or PID-control), a depiction of the implementation is shown in Fig. 8. The implementation includes the previous measurements which are labelled with the letter 'x'. In the ISA technical report, the symbol ⍇ was used. The middle path reflects the D-part and uses the setpoint increment dr from one time step to the next and the process variable increment dy.

This reference implementation is a parallel implementation because most of the series implementation are maintained due to historical reasons.

The reference implementation thus takes the standardization effort further as it provides both only one implementation option which includes all desired features relevant to the process industry and provides the actual implementation code which can be accessed and adapted if necessary.

5. CONCLUSIONS

Now the problem of a unified PID controller architecture has been solved. Standardising the PID controller in the process industry, however, remains a difficult task. The reasons are that the vendors continue to have no interest to improve on or open up their working algorithms, disclosing proprietary information.

The software architecture will also make it difficult for vendors to adopt the proposed implementation. An object-oriented implementation will work different to a normal function based one.

At a bare minimum, a PID standard for the process industry needs to agree on an algorithm. Furthermore, it has to define which signals are inputs, which are outputs and which are parameters.

The standard should also include the function block properties for the engineering station and the representation of the PID controller in the operator station. Ideally, faceplates as well as function blocks should be defined in accordance to existing standards on function blocks, human machine interface and nomenclature.

The development of a standard cannot be driven by automation vendor companies. These companies have the expertise but have no interest in pushing for any implementation other than their own. Thus, the development lies in the hand of automation user companies, retired personnel without personal investment and academics.

REFERENCES

ABB 2013. System 800xA Control, AC800M binary and analog handling. 670 p. https://library.e.abb.com/public/01044bb4edcc4608b3a61391722b72fe/3BSE035981-

600_A_en_System_800xA_Control_6.0_AC_800M_Binary_and_Analog_Handling.pdf

Hägglund, T., & Guzmán, J. L. (2024). Give us PID controllers and we can control the world. IFAC-PapersOnLine, 58(7), 103-108.

IEC 60546-1:2010. Controllers with analogue signals for use in industrial-process control systems – Part 1: Methods of evaluating the performance iTeh STANDARD PREVIEW.

IEC 61131-3:2025. Programmable controllers – Part 3: Programming languages.

ISA 101:2015. Enhancing Human-Machine Interface Design for Safer, Smarter Automation

NAMUR Standard-PID-Controller, 2018. Position paper of the working group on process operation WG 2.2. https://www.namur.net/en/work-areas-and-project-groups/wa-2-automation-systems-for-processes-and-plants/wg-22-process-operation.html

Persson, B., 2023. Functional Description Pid01A, closed loop control with auto tuner 3AST001596D013, https://library.e.abb.com/public/756cb472a7ce622dc1257b0c0055f591/Functional%20Descr%20Pid01ARevD.pdf

Sundström, E., Bauer, M., Guzmán, J. L., Hägglund, T., & Soltesz, K. (2026). A Practical Guide to PID Controller Implementation. *arXiv preprint* arXiv:2604.15918.

Zhao, Y., Eve, A., Miny, T., & Kleinert, T. (2024, September). Bridging the gap: A python-to-structured text compiler with IEC 61131-3 compliance. *In 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)* (pp. 1-7). IEEE.