Controls

Smart solutions for comfort and safety

SAIA®PCD operating system

Breaking the boundaries of PLC based controller technology

SAIA®PCD – the versatile controller for factory and building automation

- **The open architecture of the SAIA®PCD operating system** will, in the future as in the past, allow the seamless integration of the latest technologies, such as Ethernet-TCP/IP, web servers, etc. This guarantees the greatest possible security of investment.
- **The same operating system, from the smallest control device up to a multiprocessor system,** minimizes programming expenses and allows easy portability.
- **The operating system manages modular program structures in BLOCTEC and GRAFTEC,** which guide the user during the editing of efficient, reusable, well documented programs. The "connoisseur's" PCD thereby achieves unrivalled maximum performance levels.
- **The comprehensive instruction set** includes the traditional PLC instructions based on IEC 1131-3, and also high-level instructions with various addressing modes, e.g. for data and text manipulation or PID control.
- **A crucial strength of the SAIA®PCD operating system is its ability to run in parallel up to 6 communications channels** with different protocols, for example: single character, PROFIBUS FMS/DP, LONWORKS® etc. and including open protocols like Ethernet-TCP/IP and web server.
- **The SAIA®PCD operating system ensures reliable management and rapid access** to contents of user memory up to 1 MByte in size. This can be divided up by the user as required into sectors for program, text and data blocks.

Efficient project planning, programming and commissioning

- **The user-friendly PG5 tool for project management and programming** has been precisely matched to the PCD operating system. Each of the different editors has powerful commissioning and diagnostic tools.
- **Comprehensive software libraries** offer fully developed solutions that are easy to adapt, expand or integrate within function blocks.
- **The operating system also supports access for remote diagnosis, telemaintenance and firmware updates** via modem or Ethernet-TCP/IP.

Optimum interaction of operating system, hardware and programming tools

Saia-Burgess Controls has over 20 years of experience in the field of PLC technology and is among the established pioneers of the PLC industry. As the first PLC supplier to offer programming tools under Microsoft Windows and to develop CPUs based on 32-bit microprocessors, Saia-Burgess Controls can demonstrate comprehensive expertise in the fields of programming tools, CPU hardware and operating systems for programmable controllers.

High level of functionality - all included

The SAIA®PCD operating system is impressive in its functionality, which elsewhere might only be obtainable with additional hardware and software. For example, widely used communications protocols and the free ASCII drivers are all integral parts of the operating system and can be used without additional expense. This operating system also supports telemaintenance and teleservice by modem from a home base.

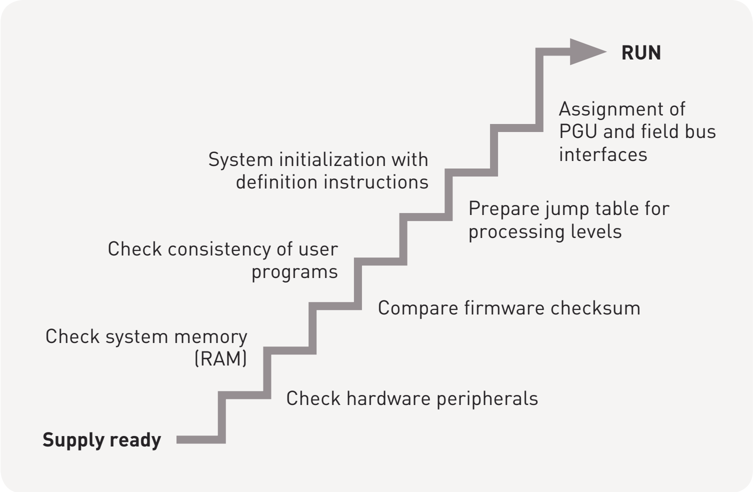
Reliability built in

The demands placed on an operating system used to control processes and machines are high. The SAIA®PCD operating system has the following distinctive features:

- **High stability and robust function**
  Perfect operation under all operating conditions, safety shut-down when faults arise.
- **Hardness of real-time capability**
  Deterministic time response and computable reaction times, due to sequence control with direct access to inputs/outputs.
- **Constant product life-cycles and reproducibility**
  Control solutions based on the SAIA®PCD operating system can be (re)produced, even after many years – without software updates, service packs or new hardware components.
- **Maintainability**
  Programming and test functions that do not interrupt the process or mechanical sequence being controlled, ensure that care and maintenance meet practical requirements – even when production is in full swing.
- **Perfect match**
  Specially tailored for PCD controllers, the SAIA®PCD operating system makes efficient, optimum use of hardware resources.

Operating system start-up

At start-up, power supply logic checks all system voltages and enables the SAIA®PCD controller to start the operating system. After extensive hardware and software checks, the operating system switches to RUN mode and processing of the user program begins.


RUN
Assignment of PGU and field bus interfaces
System initialization with definition instructions
Prepare jump table for processing levels
Check consistency of user programs
Compare firmware checksum
Check system memory (RAM)
Check hardware peripherals
Supply ready

How the interpreter works

Signals from a wide range of sensors are connected to input modules and processed by the user program in the CPU. The results are passed to the output modules and the actuators connected to them are activated accordingly.

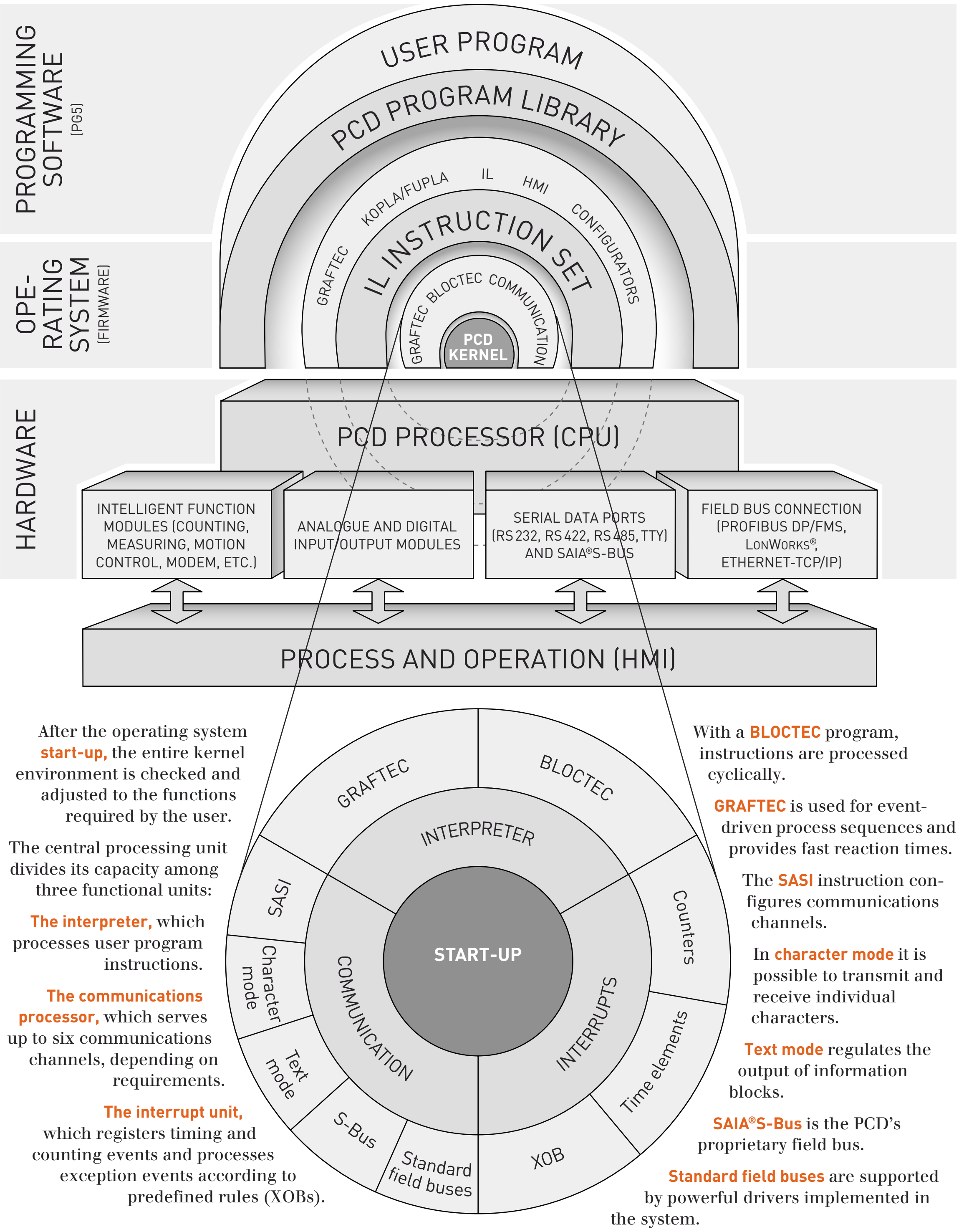
The CPU consecutively processes instructions in user memory. Program and function blocks are called conditionally as required.

When the last instruction in the program has been executed, processing starts again from the beginning. This cyclic processing is typical of all PLCs compatible with EN 61 131-1.

Structure of operating system, hardware and programming environment

The SAIA®PCD operating system is the link between the programming tools (or user programs written with them) and the controller hardware. It ensures perfect controller function – as specified by the programming.

The top position is occupied by the user program, which may be written with different methods and editors. This is executed at the appropriate processing level. Individual modules, working in fine coordination, guarantee that PLC function and communications run perfectly, while allowing access to onboard functions (fast counter, interrupt inputs, etc.) of the PCD1/2 hardware. Finally, the BIOS establishes a connection to the CPU hardware and I/O module drivers allow the smooth operation of I/O modules



After the operating system **start-up,** the entire kernel environment is checked and adjusted to the functions required by the user.

The central processing unit divides its capacity among three functional units:

The interpreter, which processes user program instructions.

The communications processor, which serves up to six communications channels, depending on requirements.

The interrupt unit, which registers timing and counting events and processes exception events according to predefined rules (XOBs).

With a **BLOCTEC** program, instructions are processed cyclically.

GRAFTEC is used for event-driven process sequences and provides fast reaction times.

The **SASI** instruction configures communications channels.

In **character mode** it is possible to transmit and receive individual characters.

Text mode regulates the output of information blocks.

SAIA®S-Bus is the PCD's proprietary field bus.

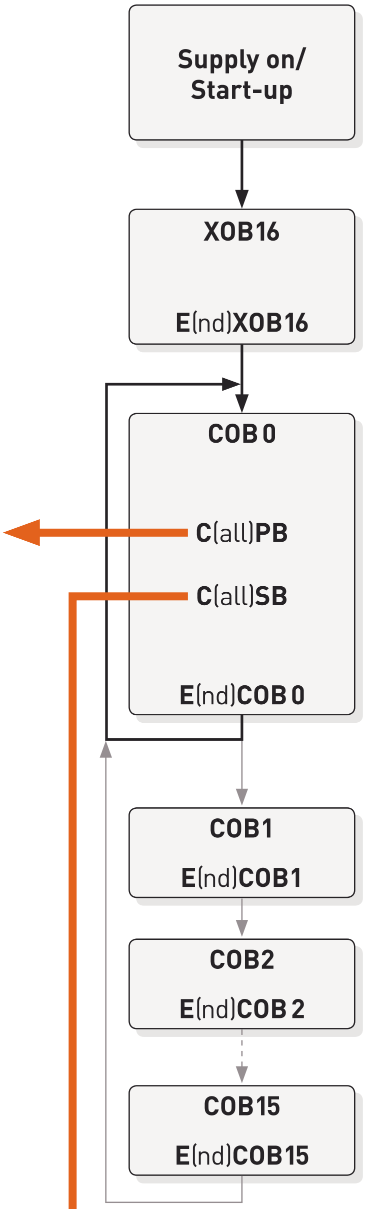
Standard field buses are supported by powerful drivers implemented in the system.

Uniform structure

Integrated program structures and routines

Processing levels

Processing levels are produced by organization blocks, which are called automatically by the operating system in RUN mode. The event-controlled XOB blocks have the highest priority, with XOB16 being called when the controller is switched on or after a “Restart Cold”, after which it cannot be repeated. The purpose of this cold-start block is to execute the process initialization that is needed for the application. It is followed by cyclical organization blocks COB0…COB15 and sequential blocks SB0…SB31, which deal with progressive process sequences.



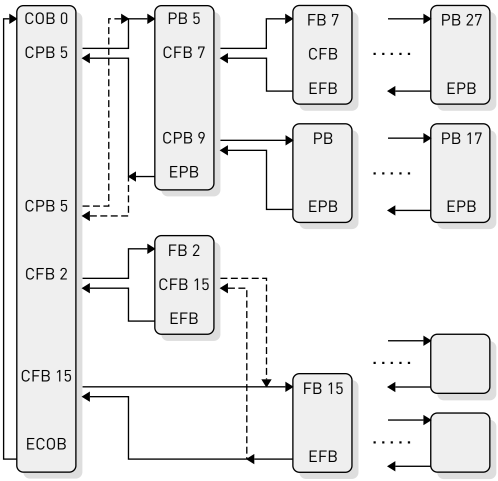
Program structure in BLOCTEC

With structured programming the user program is broken down into small, self-contained program and function blocks, with each block forming part of the overall program. This structure saves time and costs when writing a program. In addition, a structured program makes error detection easier during commissioning or maintenance.

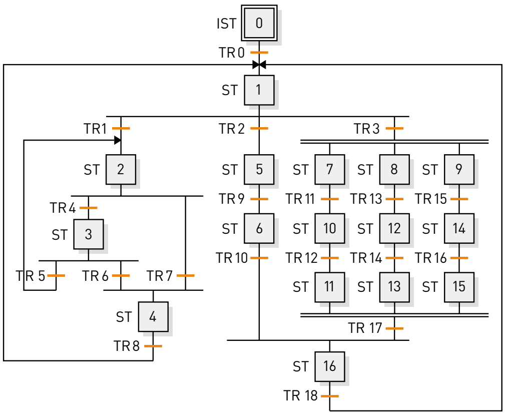
Cyclic organization block COB
In a BLOCTEC structure, the COB organization block defines the sequence in which the program will be processed. It contains calls to the program and function block subroutines. When the called block has been processed, the program returns to the COB and continues from there. When the last instruction in a COB has been processed, the processor starts on the first instruction in the next COB.

Program block PB
A PB contains operations which belong together because of their functionality. For practical reasons, any process to be controlled should always be divided into its mechanical or electrical functional units. Each of these units will be assigned to a PB.

Function block FB
Frequently recurring functions are programmed in FBs. They can be viewed as subroutines and contain frequently used, elementary processing sequences. Since they can be used in many applications, FBs are often stored in a library. Instances of FBs can be called with different parameters.



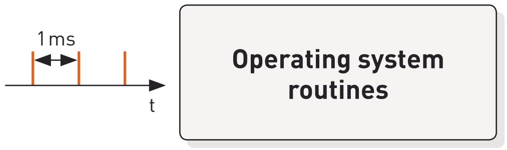
Program structure in GRAFTEC



GRAFTEC enables structured programming for sequential processes (also referred to as “sequential function charts”). Its characteristic features are linkages or conditions, actions, branches, junctions and synchronisations.

In a sequential block (SB), the user program is subdivided into events called ‘transitions’ (TRs) and actions called ‘steps’ (STs). This makes programming in GRAFTEC very simple, and programs retain their clarity even for complex tasks.

Since every step is dependent on a transition, the operating system processes only the pending transitions. This procedure allows very fast reaction times when an awaited event occurs.

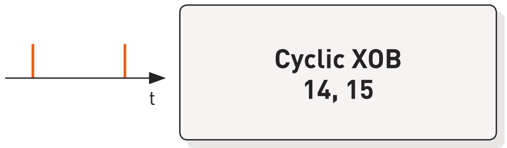


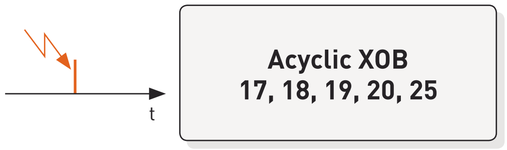
Operating system routines

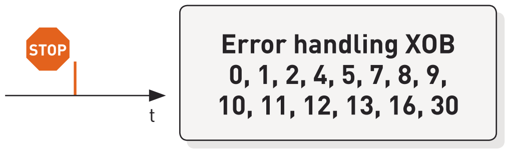
Most operating system functions run in the background during the PCD's cyclical operation. The main tasks are:

- processing time function elements, error messages and the real-time clock
- managing communications with external devices
- monitoring program cycle times, power supply and battery states
- processing interrupts, e.g. for counting functions
- managing and recording particular events
- time-dependent activities, e.g. delayed output switching

So that it can look after these background tasks, the CPU provides a small time window once every millisecond.







Event controlled exception block XOB

XOBs are called by the PCD operating system using interrupts, i.e.: when the SAIA®PCD is started up, when an interrupt input is activated, or whenever any irregularity is detected in hardware or software.

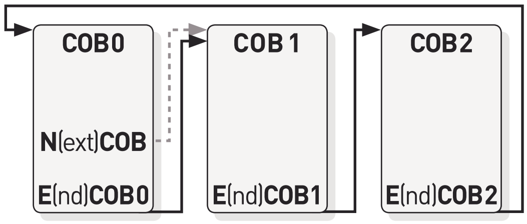
In the XOBs, the programmer is largely at liberty to define the reaction of the system and the measures to be taken. However, if a fault occurs in the system that has not been programmed with an appropriate XOB exception block, the CPU error lamp turns on and the user program will continue (if the fault permits).

Every error is entered in an error log with the date, time, error type and the relevant line of the program. If the same error recurs, its frequency is registered (rather than entering it again every time). This log (history) can be read at any time by the programming tools.

Each XOB is called for a specific event:

XOB	Description	Priority
14	Cyclic XOB	3
15	Cyclic XOB	3
17	S-Bus XOB interrupt request	3
18	S-Bus XOB interrupt request	3
19	S-Bus XOB interrupt request	3
20	Interrupt input INB1	3
25	Interrupt input INB2	3
0	Power down	4
1	Power down in extension rack	2
2	Low battery	2
4	Parity error on main bus (PCD6 only)	1

XOB	Description	Priority
5	No response from I/O module	1
7	System overload	3
8	Invalid opcode	4
9	Too many active tasks (GRAFTEC)	1
10	PB/FB nesting depth overflow	1
11	COB supervision time exceeded	3
12	Index register overflow	1
13	Error flag set	1
16	Executed on PCD start-up	1
30	RIO connection master↔slaves	1



Working with several COB organization blocks

Organization blocks COB0…COB15 are mainly used to structure an large user program. By monitoring the cycle time of each COB, it is possible to ensure that the program runs without trouble. If an unforeseen event interrupts its course, a subsequent COB can be activated to instigate the appropriate safety measures. This interruption can also be triggered deliberately when specific functions must be processed at precise, cyclical intervals, e.g. PID control.

System resources

All SAIA®PCD control devices have the same internal data structure. Only the number of inputs and outputs varies depending on the chosen system. The porting of entire user programs or their functions is made easy by this compatibility of resources, bringing the user decisive advantages.

The CPU processes each resource directly at its physical address, i. e. without a process image. This procedure saves the unnecessary copying of data and allows instant dialogue between the user program and the process.

Assignment of system resources is static and they are accessible to the user, who can read and change them when the controller is either stopped or in operation. This procedure makes commissioning easier, enables effective adjustments and allows speedy error detection.

Description	**Media code**	**Address range**	**Bit info**	**Data / explanatory notes**
Inputs	**I**	**0…8191**	High/Low	—
Outputs	**O**			
Flags	**F**	**0…8191**	High/Low	Non volatile as standard, however, the address range can be partitioned into volatile/non volatile
Timers	**T**	**0…1599**	High/Low	0…2 147 483 647 (31 bits) T/Cs use the same address range. T/C partition can be adjusted (standard: T 0…31/C 32…1599) Timers are volatile and Counters are non-volatile. Timebase selectable between 1 ms…10 s (standard 100 ms)
Counters	**C**			
Registers, non volatile	**R**	**0…4095**	Integer	−2 147 483 648…+2 147 483 647 (−2^{31}…+2^{31}−1)
			Floating point	±9.22357 × 10^{18}…±5.42101 × 10^{-20}
			ASCII	1…4 ASCII characters
			Binary	32 bits
Text block	**X**	**0…3999**	—	All ASCII characters
Data block	**DB**		—	Up to 383 × 32 bits (= 383 register contents) per data block
	X RAM	**4000…7999**	—	X/DBs use the same address range. X/DBs in the address range 4000…7999 are always in RAM, i. e. they can be written to by the user program. DBs 4000…7999 can contain up to 16 383 elements.
	DB RAM		—	
Constant	**K Constant**	**0…16 383**	—	14 bits
	Constant	**± 2 147 483 647**	—	32 bits K constants can be of a variety of types: decimal (e. g. −3456), hexadecimal (e. g. A45Fh), binary (e. g. 11011y) or ASCII (e. g. ʻSTOPʼ ⇒ 32 bits = 4 characters, 14 bits = 1 character) Time constants can be loaded into a time function element (T) and have a special format: **T#**nnn**MS** or **S** (e. g.: T#250MS corresponds to 250 ms, regardless of the chosen time base)
Semaphore	—	**0…99**	—	Latching of parts of the program when there is access by two or more CPUs to the same media
Index register	—	**0…16**	—	1 per COB or XOB
Real-time clock	**R**	**0…4095**	—	hh/mm/ss, week/day of week, month/day of month, year
BLOCTEC structure				
Organization blocks	**COB**	**0…15**	—	COBs are main parts of the program. Every application consists of at least one COB (the number is not important). The operating system processes all available blocks consecutively and repeats this cyclically. PBs and FBs can be nested in up to 7 levels.
Exception blocks	**XOB**	**0…30**	—	
Program blocks	**PB**	**0…299**	—	
Function blocks	**FB**	**0…999**	—	
GRAFTEC structure				
Sequential blocks	**SB**	**0…31**	—	—
Steps	**ST**	**0…1999**	—	
Transitions	**TR**	**0…1999**	—	
Parallel branches	—	**0…31**	—	

Status Flags

Various instructions not only write a result to a medium (register, counter, etc.) but also set 4 status flags (arithmetic status) according to the result. These can be tested by the user program. The 4 status flags are:

P (Positive)
Will be set if the result of an arithmetic instruction is positive.

N (Negative)
Will be set if the result of an arithmetic instruction is negative.

Z (Zero)
Will be set if the result of an arithmetic instruction is zero.

E (Error)
Will be set if an instruction cannot be executed.

Accumulator (A = ACCU)

The accumulator is used for the continuous storage of an intermediate result obtained from the instruction that has just been processed. The following instructions can be used to read the (logical) result of the last operation:

STH I 33
This instruction copies the state (1 or 0) of input 33 to accumulator.

OUT O 49
This instruction reads the state of the accumulator and writes it to output 49.

Apart from being read and set by means of instructions, the accumulator can also be given the state of a status flag:

```
CMP   R   33
      R   34
```

This instruction compares the contents of registers 33 and 34. If their values are the same, status flag Z (zero) is set.

ACC Z
This instruction copies the state of status flag Z (zero) to the accumulator.

Convenient text output

The SAIA®PCD operating system is capable of storing large quantities of text (up to 8000 texts) that, for example, is to be transmitted to a display or a printer. The text can contain strings, control characters and PCD data or symbol references, which allows design freedom.

Character strings are written in quotation marks, control characters (e. g. line space) in brackets and PCD data is preceded by a $ sign (e. g. Contents of register 296 ⇒ $R0296).

Addressing modes

The performance of the PCD instruction set is very substantially affected by the addressing modes available, as most of the 100+ commands allow elements to be addressed in four different ways:

Direct: Element address specified directly.

Indirect: The address is stored in a register.

Indexed: The content of the index register is added to the element address.

Indirect, indexed: A combination of indirect and indexed.

Example of indexed addressing:

```
SEI     10    ; Index = 10
SETX    O32   ; Output 42 is set
```

Configuration memory

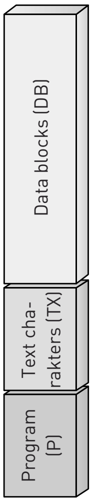
All PCD systems contain an area of memory that keeps important data from being lost, independently of data protection by battery:

- SAIA®S-Bus settings
- Modem configuration
- CPU manufacturing data
- 5 register contents (PCD1) or 50 register contents (PCD2 to PCD6), which can be written with the SYSWR instruction and read with SYSRD.

Flexible user memory

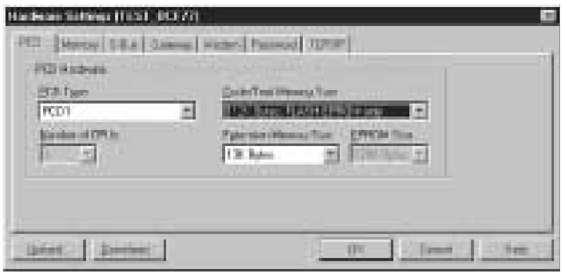
User memory can be partitioned into sectors for code, text and data blocks according to the requirements of the user program. There are instructions for transferring data between different resource types, e.g. flags, registers, timers, counters, data blocks.

- 1 register (32 bits) takes 4 bytes in a data block, or several bytes in a text (e. g. “2147483647”)
- 1 text character occupies 1 byte
- 1 program line occupies 4 bytes



Configuration of a PCD system with the PG5 programming tool

The PCD operating system permits a wide variety of operational possibilities. The user can configure the system’s PCD type, memory size and communications interfaces, using the PG5’s hardware configurator.



Instruction set of the SAIA®PCD

IEC 1131-3

The SAIA®PCD programming languages are based on the international IEC 1131-3 standard. This standard defines a set of basic instructions, which have been extended by the addition of more powerful SAIA-specific instructions. Programming structures defined by IEC 1131-3 can also be created by specific instructions in the SAIA®PCD. This allows the user to obtain the best from a controller, saving valuable development time.

Instruction set

The instruction list language includes 113 instructions. The SAIA®PCD complements usual instruction set of a traditional PLC with a number of high-level instructions, saving long sequences of instructions or time-consuming function blocks. Programs written with the SAIA®PCD need less space in memory and are easier to understand.

All data are memory-oriented. The logical linkages of bit instructions are stored in an accumulator. All word, integer and floating point operations are executed as fast, register-to-register instructions.

Logical operations (1 bit)

Read or link element states[1]:
STH, STL, ANH, ANL, ORH, ORL, XOR

Set elements or set accumulator:
OUT, OUTL, SET, RES, COM, ACC, DYN

Set elements with delay:
SETD, RESD

[1] Elements = inputs/outputs, flags, timer/counter states or individual bits from a register

Word instructions (register operations)

Load, increment, decrement registers, counters and timers:
LD, LDL, LDH, DSP, INC, DEC

Index register handling: SEI, INI, DEI, STI, RSI

Read and write multiple bits (up to 32) or digits from and to registers:
BITI, BITIR, BITO, BITOR, DIGI, DIGIR, DIGO, DIGOR

Logical linking of registers: AND, OR, NOT, EXOR

Shift operations: SHIU, SHID, SHIL, SHIR

Rotate operations: ROTU, ROTD, ROTL, ROTR

Register transfer: MOV, PUT, GET, TFR, TFRI, COPY

Integer and floating point calculations

Basic arithmetic operations: ADD, SUB, MUL, DIV

Square root, compare operations: SQR, CMP

Basic arithmetic operations:
FADD, FSUB, FMUL, FDIV, FSQR, FCMP

Trigonometric operations: FSIN, FCOS, FATAN

Other, higher mathematical operations: FEXP, FLN, FABS

Format conversion: IFP, FPI

Analogue value processing and PID command

Analogue value input and output: BITI, BITIR, BITO, BITOR

Call PID control algorithm: PID

Communications instructions

Interface assignment: SASI

Handling single characters and intelligent text output:
SRXD, STXD/STXT

Handling control lines: SICL, SOCL

Control of telegram traffic for PCD data:
SRXM, SRXMI, STXM, STXMI

Establishing a connection and diagnostics: SCON

Program structure and control instructions

for BLOCTEC: COB/ECOB, PB/EPB, FB/EFB (parameters can be supplied to FBs), XOB/EXOB

for GRAFTEC: SB/ESB, IST, ST/EST, TR/ETR

Relative, direct and indirect jumps: JR+, JR-, JPD, JPI

Call program and function blocks (BLOCTEC) and sequential blocks (GRAFTEC): CPB, CPBI, CFB, CSB

Special instructions

Read and write the real-time clock: RTIME, WTIME

Partitioning of volatile/non-volatile flags, and counters/timers:
DEFVM, DEFTC

Definition of timebase and of write-protected memory:
DEFTB, DEFWPR, DEFWPH

Call special function
The CSF instruction allows frequently occurring or complex user routines can be executed directly by the CPU in its native code. The benefits are fast processing and economies of programming time.

Read, write and compare system data

The special **SYSRD, SYSWR** and **SYSCMP** instructions give the user access to internal system data. Typical applications are:

- Access to the local configuration memory on EEPROM (see page 7)
- Time and date manipulations
- Accurate time measurement in milliseconds
- Access to XOBs or triggering an XOB
- Reading all communications definitions
- Writing and reading the flash-card

Communication

The PCD operating system allows 6 communications channels to be run with different protocols for the physical and logical connections.

Mode/Standard \ Connections	RS 232	RS 422	RS 485	TCP/IP	Modem	FTT 10	Andere
Character and text mode	X	X	X	–	X	–	–
Open data mode	–	–	–	X	–	–	–
SAIA®S-Bus	X	X	X	X	X	–	–
PROFIBUS FMS/DP	–	–	X	–	–	–	–
LonWorks®	–	–	–	–	–	X	–
EIB [1]	–	–	–	–	–	–	X
Belimo MFT	–	–	–	–	–	–	X
M-Bus [1]	–	–	–	–	–	–	X
Modbus	X	X	X	X	X	–	–

[1] These standards are supported by additional modules.

Assigning a serial interface

The SASI instruction tells a communications channel what definition will be used for the serial connection:

UART	Baud rate, character length, parity, stop bits, timeout
MODE	Character/text, S-Bus, PROFIBUS
DIAG	Flags and registers

Protocols

Character mode

This mode allows single characters to be transmitted and received via a register. When mode C is selected, it automatically manages the handshaking control signals.

Mode	Effect
MC0	Without control lines and without control characters
MC1	With control lines RTS and CTS
MC2	With control characters XON and XOFF
MC3	With echo (e. g. for operator devices)
MC4	With transmission reversing for RS 485

Example: "UART: 9600, 7, E, 1;"
"MODE: MC1;"
"DIAG: F100, R4000;"

The user program is notified of the results of communication: receipt of a character, readiness to send, error diagnostics and completion of a job. The 512-character send and receive buffers can match data exchange with the sequence of the user program.

Text mode

Text mode allows a sequence of predefined characters to be transmitted with the same automatic features as character mode. The PCD operating system can store up to 4000 texts. The length of each text is limited only by the amount of user memory. Operation of the user program is not affected during text output.

Texts can contain PCD data, e.g. for outputting the time, date or contents of a register to a printer or screen. Data can be transferred either in decimal or hexadecimal format. This requires accompanying instructions to define the output format of data in the text.

The following example shows how three register contents can be represented differently.

Without formatting	123456 –7890 5
Defined field length	123456 –7890 5
All zeroes displayed	00123456 –0007890 00000005
Field length and decimal point	123.456 –7.890 0.005
Decimal point only	1.23456 –0.789 0.05

SAIA®S-Bus

S-Bus is a proprietary field bus, included in every PCD operating system, which the user can utilize at no additional cost. S-Bus is characterized by the following:

- high transmission reliability, due to CRC16 error detection
- high speed data transfer using binary protocol
- supports remote data transmission and diagnostics via modem and the OPC standard or Ethernet-TCP/IP
- drivers also available for many process management systems

S-Bus can be used to produce master/slave networks, and also multi-master networks with the integral gateway function. For details see technical information 26/370.

PROFIBUS FMS/DP

PROFIBUS is an internationally accepted open field bus standard and is supported by the PCD operating system in two application-specific versions.

- PROFIBUS FMS for communications tasks at field and cell level

- PROFIBUS DP for exchanging data with field devices and decentralized peripherals

The PG5 programming package includes a configurator for PROFIBUS networks, which considerably simplifies the definition of bus parameters, communications relationships and objects.

LonWorks®

PCD systems can be used as host nodes in LonWorks® networks. All standard network variable types currently specified under the LonMark® are supported. The transmission of "explicit messages" also allows connection to Lon nodes with proprietary communications objects. The configuration of a PCD host node is supported by use of the SAIA®Lon configurator within the PG5 programming package.

Ethernet with SAIA®S-Bus or open data mode

Open data mode allows TCP or UDP data packages to be exchanged with foreign systems via Ethernet. It allows very flexible matching, because the specific application protocol is constructed with the user program.

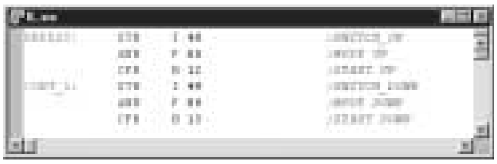
Programming

The programming unit for SAIA®PCD process controllers is a standard personal computer running the SAIA®PG5 programming tools. Apart from programming, the user can also test program sequences, optimise the process and test for errors. Development and configuration programs are therefore provided, which make it much easier to realize complex process sequences and exchange data with foreign devices.

The SAIA®PG5 programming tools will very quickly become familiar to the user, running under Microsoft Windows and using the familiar Windows graphical user interface. Programming languages are based on the international standard IEC 1131-3.

For details about the scope of performance of the PG5 programming tool, see technical information 26/362.

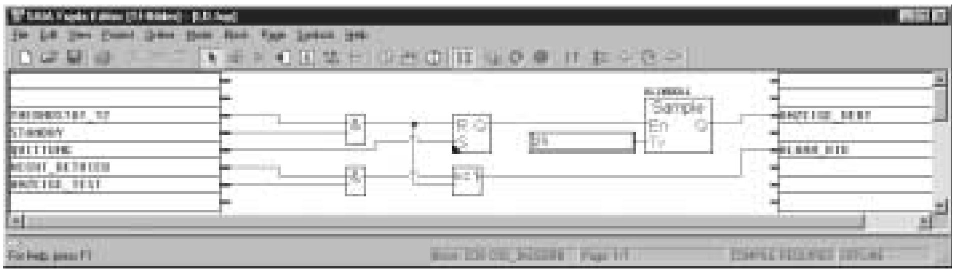
Instruction list (IL)



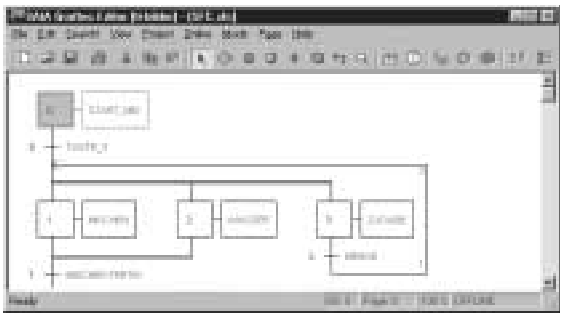
Instruction list is a low-level machine language for general purpose use. It represents the entire PCD instruction set, and is used to implement the other forms of program representation. IL instructions are processed consecutively.

Function block diagram FUPLA (FBD)

The function block diagram is a form of graphical programming. It uses digital engineering symbols as its instructions, selecting them as function boxes from a comprehensive library and connecting them by signal lines. Inputs are shown on the left side of a function box, the right side represents the outputs or the result of the function. Depending on the application, function boxes are configured with data, which can be adjusted on-line by the programming tools.

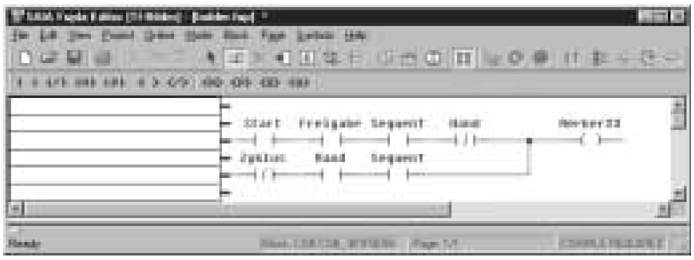


Sequential function chart (SFC)



The sequential function chart is used to define the structures of sequential processes. It represents the time sequence of process execution in the form of steps (actions) and their controlling transitions (events). The PCD operating system provides a virtual processor for SFC programs. It processes sequential structures efficiently, without the time-consuming cycling of a continuous program.

Ladder diagram (LD)



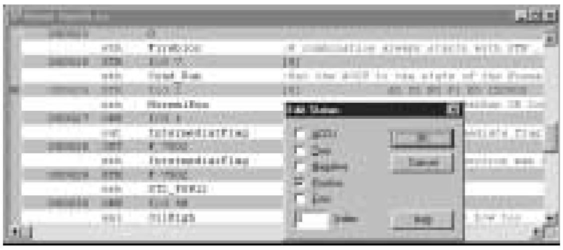
The ladder diagram represents connective networks in a similar way to the circuit path of a relay contact plan. This programming language is essentially limited to Boolean signals. More complex functions can also be integrated within a ladder diagram, but only in the form of closed function boxes.

Commissioning and Debugging

Connection of a programming unit

The operating system of any PCD supports at least 2 serial ports. One of them is used to connect the programming unit (PGU). This connection has the highest priority compared with all other serial communications interfaces (RS485, modem, Ethernet-TCP/IP). The programming unit not only starts up programs for the first time, but can also be used to execute other useful functions:

- Single-step processing of a program up to a defined halt point, or to a specific instruction, or to an error.
- Setting, loading and modifying all elements, data and text.
- Displaying all data in a previously chosen format, with data being continuously refreshed during operation.



The PG5 debugger includes familiar functions, such as step-by-step, breakpoints, the editing of data and instructions, plus the "synchronous data view", which continuously displays the content of the accumulator, status flags, resources and the index register.

System analysis, debugging and maintenance

The SAIA®PCD operating system has several integrated functions for user program analysis and maintenance.

XOB for error handling

XOBs are started by the system as soon as an exceptional event occurs. The reaction of the system to these events is user-programmable (e.g. in response to XOB2 "Battery supply error", an appropriate SMS text message could be transmitted to the maintenance engineer).

Analysis of error log

Errors handled by XOBs are also logged in the history list. This can be displayed and analysed using the PG5 programming tools.

Breakpoints / step-by-step

"Conditional run", which has been integrated into the operating system, allows monitoring of system resources, accumulator, status flags or the index register. As soon as a monitored element assumes a defined value, the CPU stops processing the program at the current line. The program can be executed step-by-step to allow each line to be checked and processed individually.

Loading updated program blocks

The unique download-in-run function does not just allow individual program blocks to be updated during operation. The operating system can simultaneously download up to five changed blocks and start them synchronously. This has the advantage of increased reliability when loading nested programs and complicated program modifications, compared with traditional solutions.

Remote diagnosis and telemaintenance

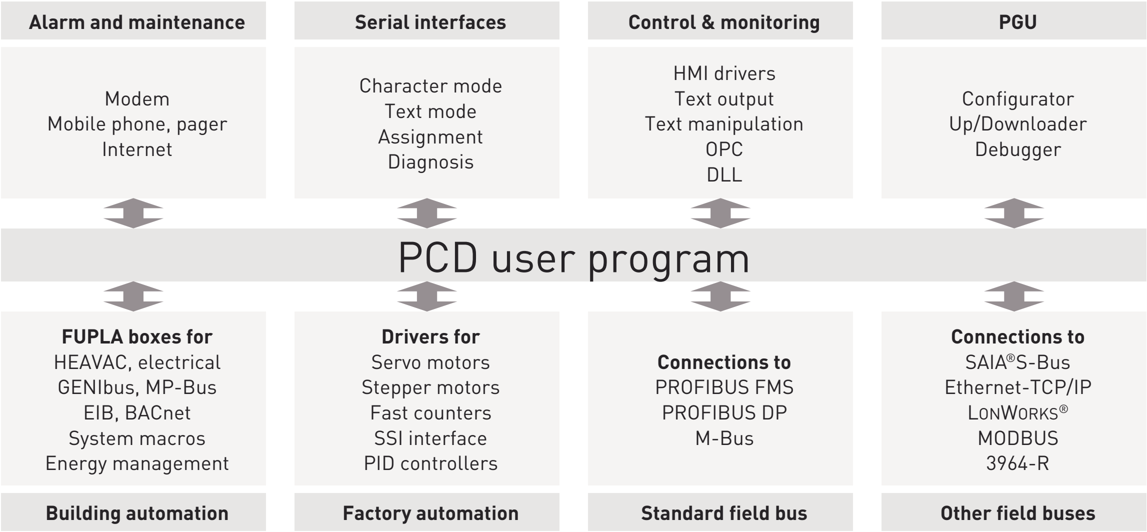
The operating system of every SAIA®PCD has an integral modem connection. No programming of any kind is required to establish a modem connection between the controller and the PG5 programming tools.

Firmware update

The ..M170 CPU allows firmware to be updated by the PG5. This enables the user to take advantage of new features, including the possibility of control & monitoring via Internet/intranet with the web server.

Overview of user library

The PCD operating system supports a large number of pre-coded functions. These are available as function blocks or FBoxes, each of which fulfils a particular function. Other groups serve to connect with communications units or as drivers for intelligent function modules (e.g. the servo drives).



Alarm and maintenance

Modem
Mobile phone, pager
Internet

Serial interfaces

Character mode
Text mode
Assignment
Diagnosis

Control & monitoring

HMI drivers
Text output
Text manipulation
OPC
DLL

PGU

Configurator
Up/Downloader
Debugger

PCD user program

FUPLA boxes for
HEAVAC, electrical
GENIbus, MP-Bus
EIB, BACnet
System macros
Energy management

Building automation

Drivers for
Servo motors
Stepper motors
Fast counters
SSI interface
PID controllers

Factory automation

Connections to
PROFIBUS FMS
PROFIBUS DP
M-Bus

Standard field bus

Connections to
SAIA®S-Bus
Ethernet-TCP/IP
LONWORKS®
MODBUS
3964-R

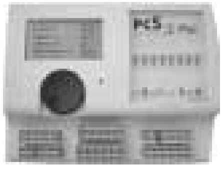
Other field buses

Commissioning und Debugging

Hardware overview

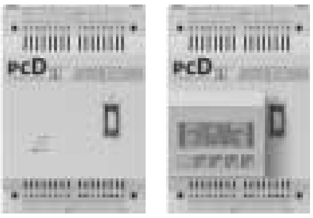
Smart solutions for comfort and safety

PCS1 compact controller



- up to 20 digital inputs and up to 4 digital outputs 24 VDC plus 8 relay outputs 4 A/250 VAC (4 "make" contacts/4 change-over relays) plus up to 16 analogue inputs (up to 4 × 0...10 V plus 12 × Pt/Ni 1000)
- user memory of 384 kBytes
- up to 4 interfaces (serial, SAIA®S-Bus, LONWORKS®)

PCD1 series



- up to 32 inputs/outputs or up to 64 inputs/outputs if using digital modules with 16 I/Os
- up to 140 kBytes user memory
- up to 2 interfaces (serial, PROFIBUS DP, LonWorks® Ethernet-TCP/IP)

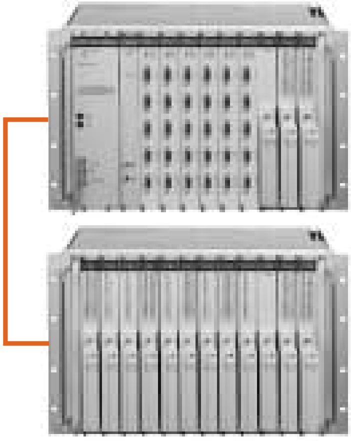
PCD2 series

- up to 128 inputs/outputs or up to 255 inputs/outputs if using digital modules with 16 I/Os
- up to 1MByte user memory
- up to 6 interfaces (serial, SAIA®S-Bus, PROFIBUS FMS/DP, LONWORKS®, Ethernet-TCP/IP)

PCD4 series

- up to 510 inputs/outputs or up to 1020 inputs/outputs if using digital modules with 16 I/Os
- up to 1 MByte user memory
- up to 6 interfaces (serial, SAIA®S-Bus, PROFIBUS FMS/DP, LONWORKS®, Ethernet-TCP/IP)
- up to 2 CPUs

PCD6 series



- up to 5100 inputs/outputs
- up to 1 MByte user memory
- up to 30 interfaces (serial, SAIA®S-Bus, PROFIBUS FMS/DP, Ethernet-TCP/IP)
- up to 6 CPUs

Saia-Burgess Controls Ltd.
Bahnhofstrasse 18
CH-3280 Murten / Switzerland

Telephone ++41 26 672 71 11
Telefax ++41 26 670 44 43

E-mail: pcd@saia-burgess.com
Homepage: www.saia-burgess.com
Support: www.sbc-support.ch

Saia-Burgess Controls Kft.
Liget utca 1
H–2040 Budaörs

Telephone 023 / 501 170
Telefax 023 / 501 180

E-mail: office@saia-burgess.hu
Homepage: www.saia-burgess.hu
Support: www.sbc-support.ch

Your local contact:

Printed in Switzerland 26/354 E3 04. 2002 TA20 Subject to change without notice.