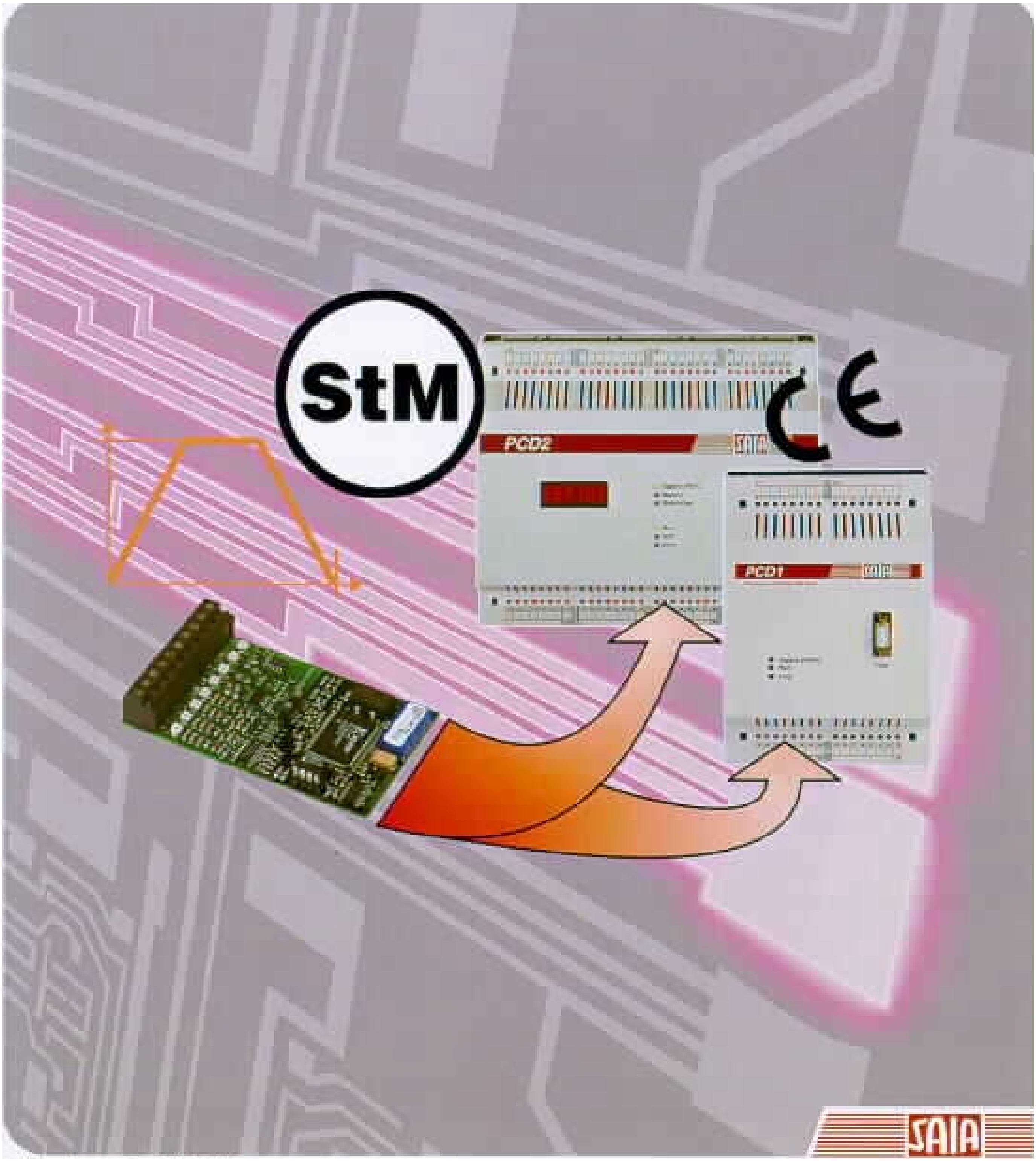
SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA®PCD

Process Control Devices

PCD2.H210 Motion control module for stepper motors



English edition 26/760 E2

SAIA-Burgess Electronics Ltd.
Bahnhofstrasse 18
CH-3280 Murten (Switzerland)
http;//www.saia-burgess.com

BA: Electronic Controllers   Telephone 026 / 672 72 72
Telefax 026 / 672 74 99

SAIA-Burgess Companies

Switzerland
SAIA-Burgess Electronics AG
Freiburgstrasse 33
CH-3280 Murten
☎ 026 672 77 77, Fax 026 670 19 83

France
SAIA-Burgess Electronics Sàrl.
10, Bld. Louise Michel
F-92230 Gennevilliers
☎ 01 46 88 07 70, Fax 01 46 88 07 99

Germany
SAIA-Burgess Electronics GmbH
Daimlerstrasse 1k
D-63303 Dreieich
☎ 06103 89 060, Fax 06103 89 06 66

Nederlands
SAIA-Burgess Electronics B.V.
Hanzeweg 12c
NL-2803 MC Gouda
☎ 0182 54 31 54, Fax 0182 54 31 51

Austria
SAIA-Burgess Electronics Ges.m.b.H.
Schallmooser Hauptstrasse 38
A-5020 Salzburg
☎ 0662 88 49 10, Fax 0662 88 49 10 11

Belgium
SAIA-Burgess Electronics Belgium
Avenue Roi Albert 1er, 50
B-1780 Wemmel
☎ 02 456 06 20, Fax 02 460 50 44

Italy
SAIA-Burgess Electronics S.r.l.
Via Cadamosto 3
I-20094 Corsico MI
☎ 02 48 69 21, Fax 02 48 60 06 92

Hungary
SAIA-Burgess Electronics Automation Kft.
Liget utca 1.
H-2040 Budaörs
☎ 23 501 170, Fax 23 501 180

Representatives

Great Britain
Canham Controls Ltd.
25 Fenlake Business Centre, Fengate
Peterborough PE1 5BQ UK
☎ 01733 89 44 89, Fax 01733 89 44 88

Portugal
INFOCONTROL Electronica e Automatismo LDA.
Praceta Cesário Verde, No 10 s/cv, Massamá
P-2745 Queluz
☎ 21 430 08 24, Fax 21 430 08 04

Denmark
Malthe Winje Automation AS
Håndværkerbyen 57 B
DK-2670 Greve
☎ 70 20 52 01, Fax 70 20 52 02

Spain
Tecnosistemas Medioambientales, S.L.
Poligono Industrial El Cabril, 9
E-28864 Ajalvir, Madrid
☎ 91 884 47 93, Fax 91 884 40 72

Norway
Malthe Winje Automasjon AS
Haukelivn 48
N-1415 Oppegård
☎ 66 99 61 00, Fax 66 99 61 01

Czech Republic
ICS Industrie Control Service, s.r.o.
Modranská 43
CZ-14700 Praha 4
☎ 2 44 06 22 79, Fax 2 44 46 08 57

Sweden
Malthe Winje Automation AB
Truckvägen 14A
S-194 52 Upplands Våsby
☎ 08 795 59 10, Fax 08 795 59 20

Poland
SABUR Ltd.
ul. Druzynowa 3A
PL-02-590 Warszawa
☎ 22 844 63 70, Fax 22 844 75 20

Suomi/ Finland
ENERGEL OY
Atomitie 1
FIN-00370 Helsinki
☎ 09 586 2066, Fax 09 586 2046

Australia
Siemens Building Technologies Pty. Ltd.
Landis & Staefa Division
411 Ferntree Gully Road
AUS-Mount Waverley, 3149 Victoria
☎ 3 9544 2322, Fax 3 9543 8106

Argentina
MURTEN S.r.l.
Av. del Libertador 184, 4° “A”
RA-1001 Buenos Aires
☎ 054 11 4312 0172, Fax 054 11 4312 0172

After sales service

USA
SAIA-Burgess Electronics Inc.
1335 Barclay Boulevard
Buffalo Grove, IL 60089, USA
☎ 847 215 96 00, Fax 847 215 96 06

Issue : 22.11.2000

Subjet to change without notice

SAIA® Process Control Devices

Motion control module for stepper motors

PCD2.H210

Edition 26/760 E2 - 05.99

Subject to technical changes

Updates

Manual : PCD2.H210 - Motion control modules for stepper motors - Edition E2

Date	Chapter	Page	Description
15.05.2000	7.1.2	7-5/6	Description: Zero position
15.05.2000	Appendix A	A-13	Offset for reference position

Table of contents

Page

Notes :

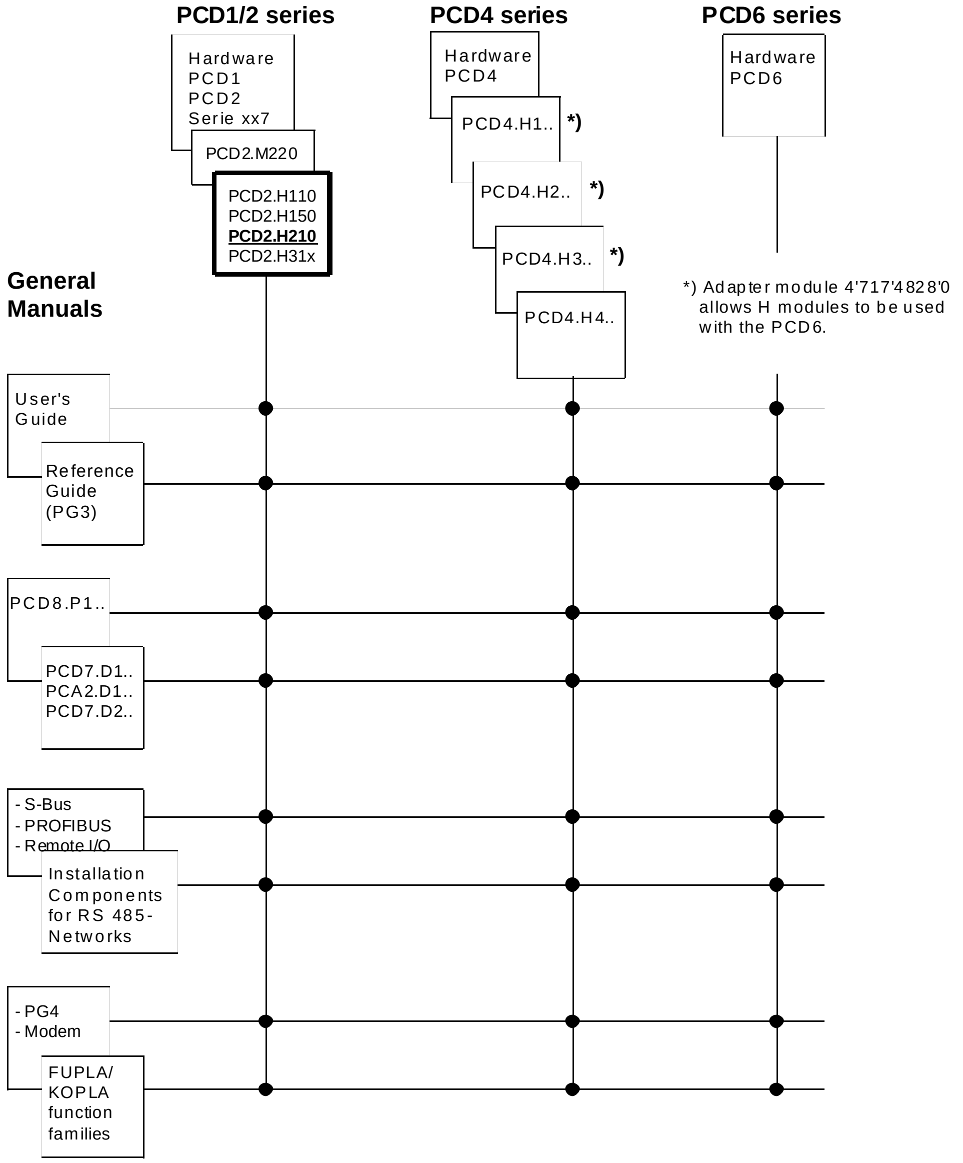
Please note :

A number of detailed manuals are available to aid installation and operation of the SAIA PCD. These are for use by technically qualified staff, who may also have successfully completed one of our "workshops".

To obtain the best performance from your SAIA PCD, closely follow the guidelines for assembly, wiring, programming and commissioning given in these manuals. In this way, you will also become one of the many enthusiastic SAIA PCD users.

If you have any technical suggestions or recommendations for improvements to the manuals, please let us know. A form is provided on the last page of this manual for your comments.

Summary



*) Adapter module 4'717'4828'0 allows H modules to be used with the PCD6.

Reliability and safety of electronic controllers

SAIA-Burgess Electronics Ltd. is a company which devotes the greatest care to the design, development and manufacture of its products:

- state-of-the-art technology
- compliance with standards
- ISO 9001 certification
- international approvals: e.g. Germanischer Lloyd, UL,
  Det Norske Veritas, CE mark ...
- choice of high-quality componentry
- quality control checks at various stages of production
- in-circuit tests
- run-in (burn-in at 85°C for 48h)

Despite every care, the excellent quality which results from this does have its limits. It is therefore necessary, for example, to reckon with the natural failure of components. For this reason SAIA-Burgess Electronics Ltd. provides a guarantee according to the "General terms and conditions of supply".

The plant engineer must in turn also contribute his share to the reliable operation of an installation. He is therefore responsible for ensuring that controller use conforms to the technical data and that no excessive stresses are placed on it, e.g. with regard to temperature ranges, overvoltages and noise fields or mechanical stresses.

In addition, the plant engineer is also responsible for ensuring that a faulty product in no case leads to personal injury or even death, nor to the damage or destruction of property. The relevant safety regulations should always be observed. Dangerous faults must be recognized by additional measures and any consequences prevented. For example, outputs which are important for safety should lead back to inputs and be monitored from software. Consistent use should be made of the diagnostic elements of the PCD, such as the watchdog, exception organization blocks (XOB) and test or diagnostic instructions.

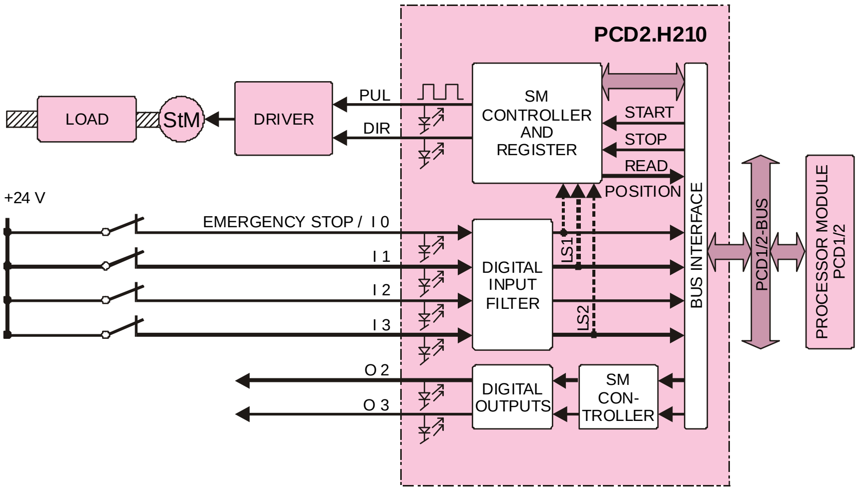
If all these points are taken into consideration, the SAIA PCD will provide you with a modern, safe programmable controller to control, regulate and monitor your installation with reliability for many years.

1. Introduction

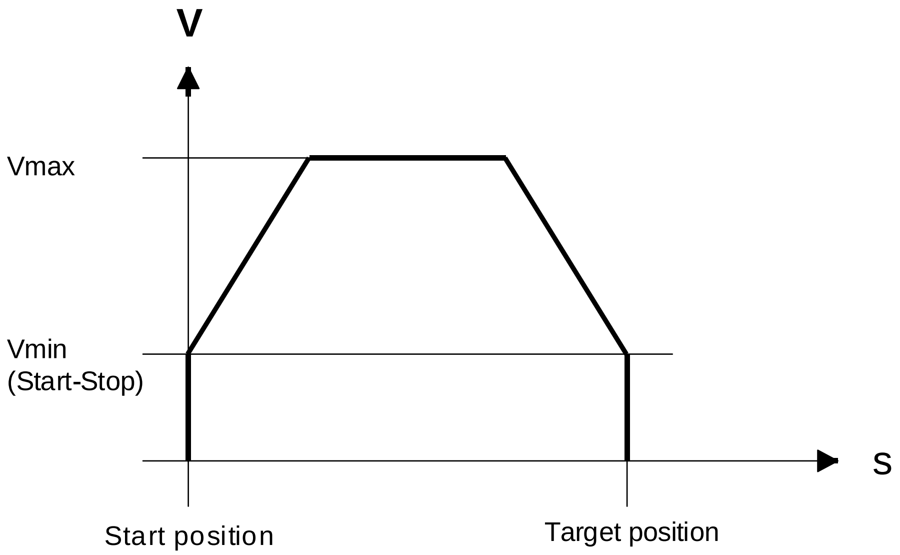
1.1 General

The PCD2.H210 module permits the completely autonomous control and monitoring of motion cycles for one stepper motor with run-up and braking ramps. Each ..H210 module controls an independent axis and supplies a single-phase pulse train to the power stage for a stepper motor.

Block diagram of a stepper motor



Typical moving profile



1.2 Function and application

This low-cost module can be plugged into any I/O socket on a PCD1 or PCD2. It is used to drive the power stage for a stepper motor axis up to a frequency of 19.454 kHz.

This means that up to 16 axes can be controlled with the PCD2.M1../..M2.. and up to 4 axes with a PCD1. Since each ..H210 module additionally has 4 configurable digital inputs and 2 digital outputs, 16 axes provide up to 64 more digital inputs and 32 digital outputs which are available for other process control tasks.

The ..H210 module permits the completely autonomous control and monitoring of motion cycles for one stepper motor with run-up and braking ramps. The commands necessary for the cycle of stepper motor movements are transmitted to the module by function blocks in the user program. During movement, the SM processor monitors the frequency profile, controlling acceleration and braking ramps to drive the axes to their destination position without loss of steps. Each ..H210 module controls an independent axis. However, several axes can be started in coordinated, quasi-synchronous operation.

1.3 Main characteristics

- Low-cost controller for open loop operation, with high accuracy and reliability.
- Frequency (9.5 to 19454 Hz), acceleration and pulse counts are controlled with quartz precision.
- Current axis position or number of output pulses can be read at any time.
- Input I 2 can be used as a normal 24 VDC digital input if not used by the “Home” function block.
- 3 additional inputs are available. These can be configured during initialization as either normal 24 VDC digital inputs, as limit switches (I 1 and I 3, which cause a stop with brake ramp) and/or as emergency stop (I 0).
- 2 digital outputs (0.5 A, 24 VCD) are available on the same module for other process tasks.

1.4 Typical areas of application

- Automatic handling and assembly machines
- Pick and place functions
- Palletizing equipment
- Automatic angle control, e. g. of cameras, headlamps, aerials, etc.
- For the general control of drives requiring high torque from stationary
- Motion control of static axes (set-up)

1.5 Programming

Pre-programmed functional blocks make it possible to simply enter the parameters necessary for the motion control. These FBs (IL) and FBoxes (FUPLA) are used by the PG4 (Windows programming software). A comprehensive manual includes detailed descriptions of each function block, with associated practical examples.

lnitialization commands

INIT	Select the module number Select the frequency range Activate emergency-stop function Activate limit switches Load Vmin (start-stop frequency) Load Vmax Load acceleration

Execution commands

EXEC	Load the relative destination Load the absolute destination Start motion (start pulse output) Stop (interrupt) motion Continue motion Read counter (read position) Read module identity Set/reset digital outputs

Commissioning command

HOME	Home function (seek to reference switch)

Diagnosis and error handling

Recognition of wrong FB parameters and programming errors.
Timeout supervision for FB 'Home'.

Notes :

2. Technical data

2.1 Technical data of the hardware

Digital inputs

Total	4
Nominal voltage Low range : High range :	24 V - 30 ... +5 V +15 ... +30 V
Source operation only	for safety reasons closed contacts should be used.
Input current (typical)	6.5 mA
Switching type	galvanically connected
Input filter	< 1 ms

Digital outputs

Total	4
Current range	0.5 A each in range 10 ... 32 VDC, residual ripple max. 10%
Galvanic isolation	no
Potential drop	max. 0.3 V at 0.5 A
Logic	positive (positive switching)
Output delay	typically 50 µs, max 100 µs at ohmic load

Power supply

Internal supply from PCD1/2 bus	5 VDC, 20 ... 45 mA
External by user for all outputs	24 VDC (10 ... 32 VDC), max. 2 A smoothed ripple max. 10%

Operating conditions

Ambient temperature	operation :	0 ... +50°C without forced ventilation
	storage :	-20 ... +85°C
Interference immunity	CE mark	according to EN 50081-1 and EN 50082-2

Ordering details

PCD2.H210	Motion control module for 1 stepper motor axis
PCD9.H21E	Software library with function blocks

LED displays

Total		8	
	LED 0 :	*) Voltage at input 0	(Emergency stop)
	LED 1 :	*) Voltage at input 1	(LS1)
	LED 2 :	*) Voltage at input 2	(REF)
	LED 3 :	*) Voltage at input 3	(LS2)
	LED 4 :	Voltage at output 0 :	PUL
	LED 5 :	Voltage at output 1 :	DIR
	LED 6 :	Voltage at output 2	
	LED 7 :	Voltage at output 3	

*) status inverted when used as limit switch

Programming Based on PCD user program (PG4) and pre-programmed functional blocks.

2.2 Electrical specification

Internal power consumption

+5 V	20 ... 45 mA
Uext	0 ... 10 mA (without load current)

External power supply

Terminals +/-	10 ... 32 VDC smoothed, residual ripple max. 10% TVS diode 39 V ±10% max. 2 A for outputs, not protected against wrong polarity!

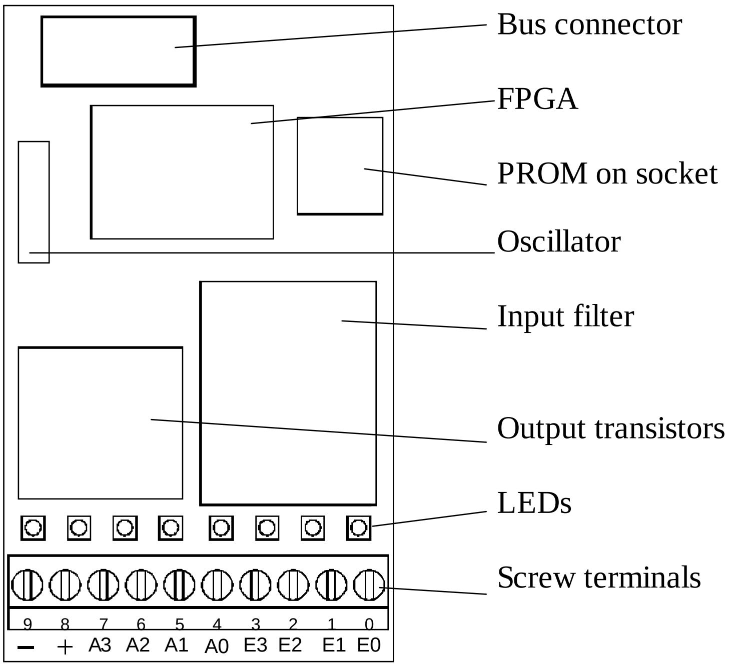
Digital inputs	4 digital inputs (E0 ... E3) (see chapter 2.1)
Digital outputs	4 digital outputs (A0 ... A3) (see chapter 2.1)

2.3 Function specific data

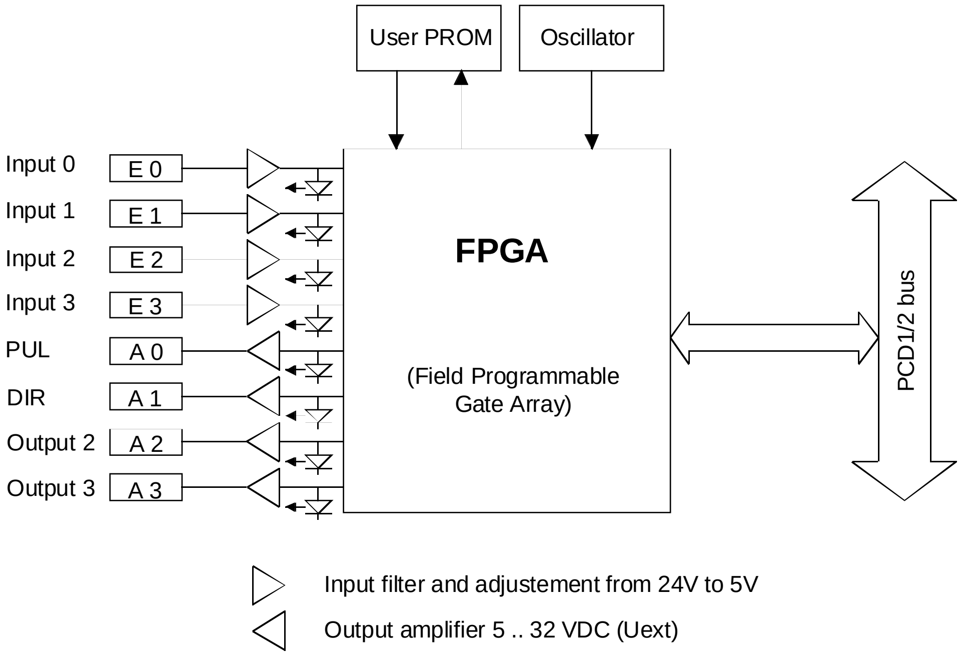
Number of systems	1
Positioning distance (counting range)	0 to 16 777 215 (24 Bit)
Frequency ranges (selectable)	9.5 ... 2 431 Hz 19 ... 4 864 Hz 38 ... 9 727 Hz 76 ... 19 454 Hz
Acceleration	0.6 ... 1 224 kHz/s non-linear range division, depending on the selected frequency range (see table in chapter 7.1.3)
Profile generator	with symmetrical acceleration and braking ramps.
Data protection	All data in this module is volatile (non volatile PCD registers are available)

3. Presentation

Equipped module



Block diagram

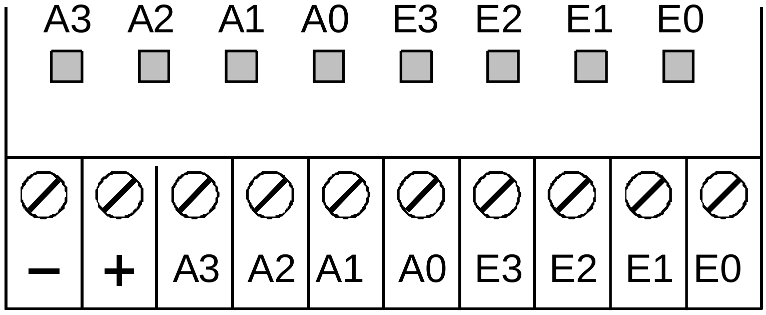

User PROM
Oscillator
Input 0
Input 1
Input 2
Input 3
PUL
DIR
Output 2
Output 3
E 0
E 1
E 2
E 3
A 0
A 1
A 2
A 3
FPGA
(Field Programmable Gate Array)
PCD1/2 bus
Input filter and adjustement from 24V to 5V
Output amplifier 5 .. 32 VDC (Uext)

Notes :

4. Terminals and meaning of the LED's

Screw terminals

This picture shows the text on the print. The I/O connector is standard from 0 ... 9 (from right to left)



Inputs 4

Terminal 0 = E0 : configurable as emergency stop or for general purpose use

Terminal 1 = E1 : configurable as limit switch LS1 or for general purpose use

Terminal 2 = E2 : configurable as reference switch or for general purpose use

Terminal 3 = E3 : configurable as limit switch LS2 or for general purpose use

Outputs 4

Terminal 4 = A0 : Output PUL (pulses for motor)

Terminal 5 = A1 : Output DIR (direction of motor rotation)

Terminal 6 = A2 : programmable as required

Terminal 7 = A3 : programmable as required

Supply

Terminal 8 = + + 24 VDC

Terminal 9 = – GND

LED displays

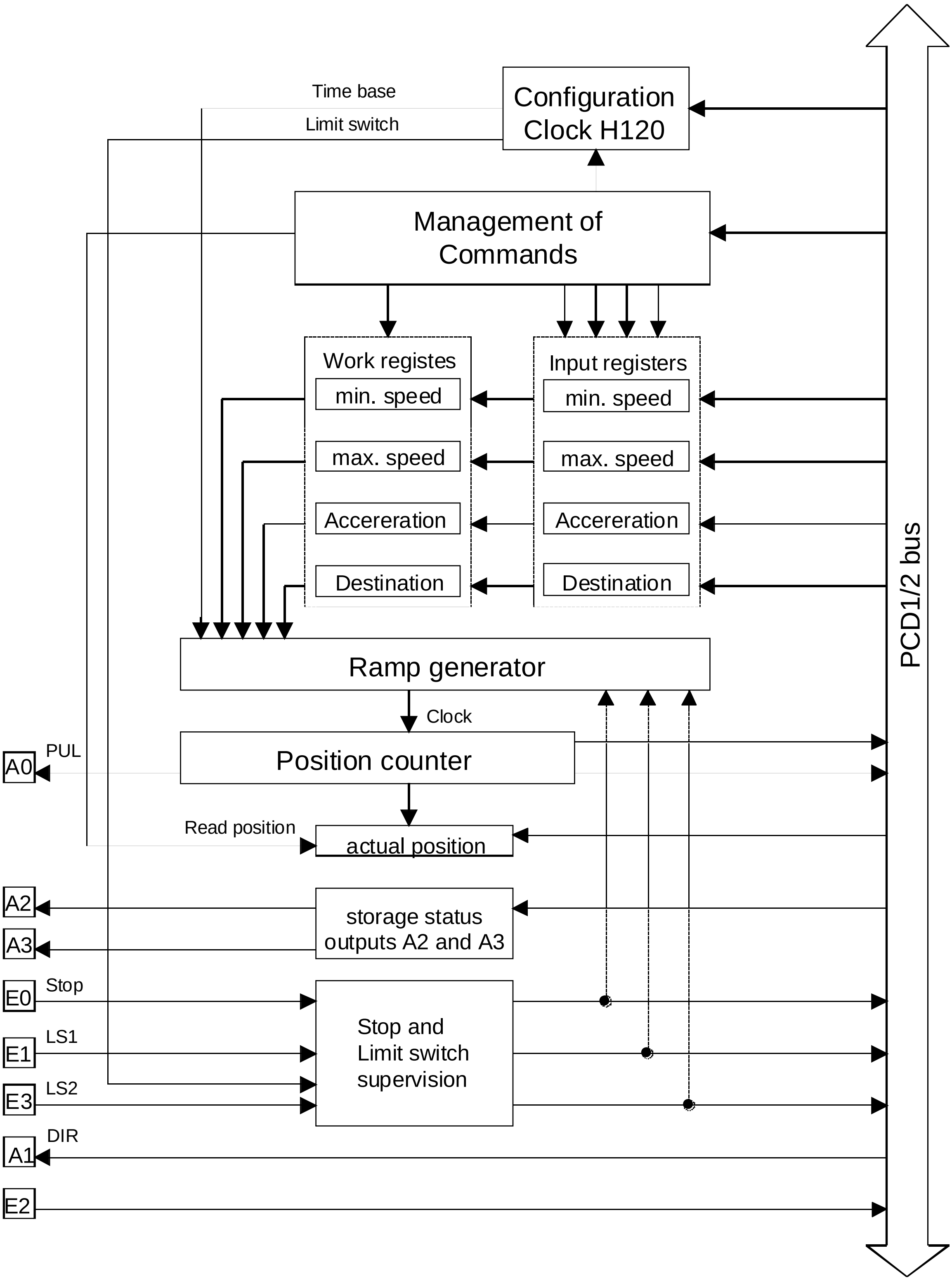
Total 8

LED 0 :	*) Voltage at input 0	(Emergency stop)
LED 1 :	*) Voltage at input 1	(LS1)
LED 2 :	*) Voltage at input 2	(REF)
LED 3 :	*) Voltage at input 3	(LS2)
LED 4 :	Voltage at output 0 :	PUL
LED 5 :	Voltage at output 1 :	DIR
LED 6 :	Voltage at output 2	
LED 7 :	Voltage at output 3	

*) status inverted when used as limit switch

5. Functional description

5.1 Block diagram of the module



5.2 Module description

Successful operation of a stepper motor requires the definition of 4 parameters :

- Start/stop frequency, i.e. the frequency with which a stepper motor can be started and stopped directly without losing steps
- Maximum frequency at which the stepper motor can be accelerated to run under all conditions
- Optimum acceleration at which to change from minimum to maximum frequency and back again.
- Number of steps to be executed in a travel

The first three parameters are motor or system-specific, i.e. they are defined once and then are not usually modified again. The fourth parameter, the number of steps to be executed, depends on the task and must be continuously revised by the user program.

The first three parameters are loaded during intialization of the module (FB INIT / FBox INIT) and can also be changed individually if needed. The number of steps are normally loaded by the user program just before the impulse output. In very fast applications the number of steps for the next movement can be pre-loaded into the input register during the preceding movement, so that it is transmitted into the working register on completion of the movement, thus starting the next movement immediately. (It would be possible to proceed in the same way with the other parameters. However, these normally remain unchanged.)

Another command (EXEC - Start) starts pulse output and motion. Pulses are output to "A0" (PUL).

During initial configuration the time base is defined. A choice is possible between these ranges :

Range 0	→	9.5 ... 2 431 Hz,
Range 1	→	19 ... 4 864 Hz,
Range 2	→	38 ... 9 727 Hz,
Range 3	→	76 ... 19 454 Hz

When the destination position is reached, a flag (ondest_x) is set. Querying this flag controls the sequential processing of the user program.

It is also possible to interrupt a pulse string (EXEC - Stop) and then continue the interrupted movement (EXEC - Continue), concluding it without loss of steps according to the specification. However, this must be planned for in the user program.

Current position, i.e. internal counter status, can be read at any time and output to a display (EXEC - RdPosition). The absolute position is refreshed in the register 'rPosAbs_x' with every instruction 'RdPosition'.

There is a difference between relative and absolute operation (flag Abs_x). In "relative" operation, the path (i.e. the steps to be executed) to approach the destination is always loaded. If 1000 forward steps are to be executed, the destination is loaded as 1000. To return to the starting point, -1000 is then programmed. With negative values output 'A 1' (DIR = direction) is now automatically set low. This can be used to invert the direction of the motor.

In "absolute" operation, the absolute position to be approached is loaded directly as the destination. The FB then calculates the relative movement for the SM controller. Absolute position (Register PosAb_x) must be loaded with the effective position before the first movement. For example, to travel 1000 steps up from a zero position and then return to the starting point, in absolute operation enter position 1000 as the destination and start the motor. To return to the starting position, load position 0.

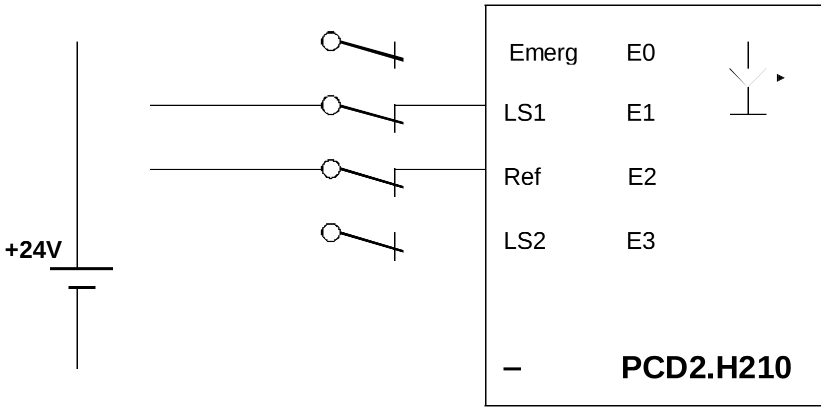
Input "E0" (emergency stop) can be used to stop motion abruptly (without a braking ramp). Afterwards values must be redefined and motion re-initialized.

Apart from the emergency stop input (E0) there are 3 other inputs (E1 - E3) available. 'E1' and 'E3' can be configured (INIT) either as general purpose digital inputs, or as limit switches (LS1 and LS2). Input 'E2' is used by the FB 'Home' as reference switch (see chapter 5.4). Is FB 'Home' not used, 'E2' is configured as general purpose digital input.

Limit switches "LS1" and "LS2", reference switch "Ref" and the emergency stop are normally closed and supply +24 V to the inputs. Please note that a smoothed D.C. voltage must be used (see technical data) because the input circuits have been designed for stops to happen without a delay, i.e. with stepping accuracy (time constant of input filter < 1 ms). If these inputs are not configured for limit or stop the behaviour is the same as for "normal" inputs.

LEDs always show the voltage at the relevant input.

Connection drawing it inputs are configured as emergency stop or limit switches :



When "LS1" or "LS2" become active (voltage = 0, LED off), a braking ramp is triggered directly if the switches have been configured as limit switches.

Input signal 'LSxTrig_x' shows that a motion was stopped by a limit switch (LS). The signal is set 'H' and remains 'H' also if the LS become inactive. By the start of a new movement the 'LSxTrig_x' becomes 'L' again (if no limit switch is active).

When the 'emergency stop' becomes active, the motion is stopped immediately without a braking ramp. The input signal 'EmergTrig_x' records this. If after a emergency stop the position is read (RdPosition) the position where the emergency stop was activated is shown. This is not imperatively the position were the motor is stopped.

Input "Ref". has no direct effect on motion and is only used by the FB 'Home' to approach the reference position.

The polling of the inputs always occurs on the symbolic names independently of the configuration.

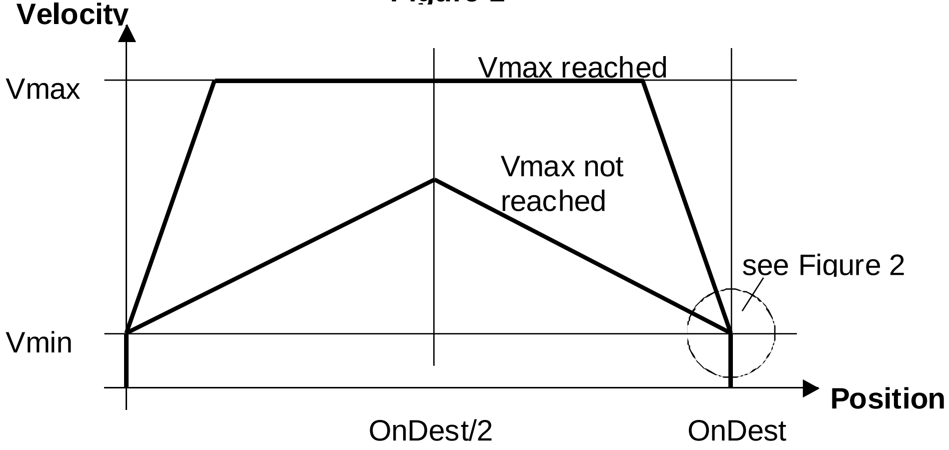
Emerg_x	for input E0
LS1_x	for input E1
REF_x	for input E2
LS2_x	for input E3

where 'x' is the number of the module.

The 2 outputs 'A2' and 'A3' are not for any specific use and can be turned on by the FB in the user program (EXEC - SetOut2, EXEC - ResOut2, EXEC - SetOut3, EXEC - ResOut3). The read back of the logic state of these outputs is not possible.

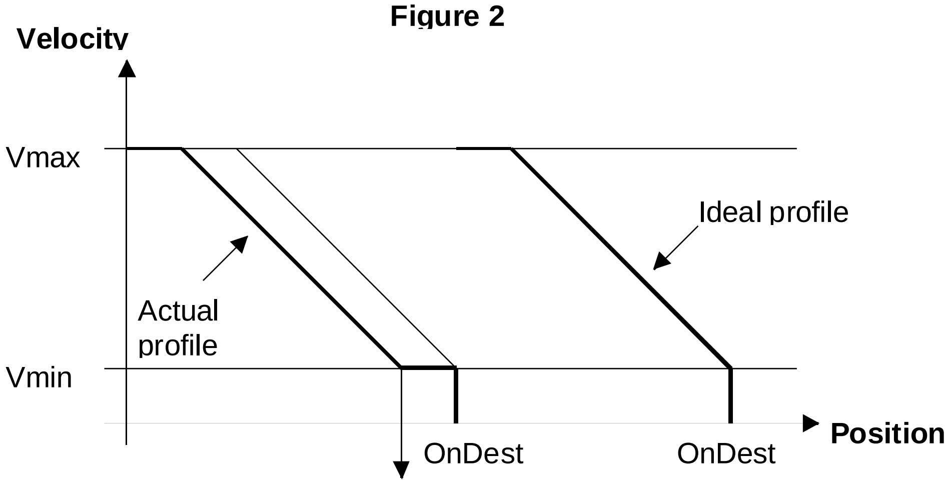
5.3 Additional information : frequency profile

Figure 1



Number of steps for acceleration = Number of steps for braking

Figure 2



Motion completed with some steps at Vmin

The equation shown below enables the maximum time for completion of the movement to be determined.

$$\text{max. time} = \frac{\text{Vmax}}{\text{Vmin}} \times \text{Tacc}$$

where :

Vmax = maximum velocity attained during movement

Vmin = minimum programmed velocity (start-stop frequency)

Tacc = acceleration time: n * 250 µs (1 ≤ n ≤ 255)

Please note : The formula shown corresponds to the worst case. Small modifications to parameters Vmax, Vmin and Tacc can optimize motion and completion time.

5.4 Additional information : homing (FB Home)

Homing can be carried out independently using FB 'Home'. Nine parameters are required to define travel to the reference position. Section 7.1.2 describes how to call FB 'Home'.

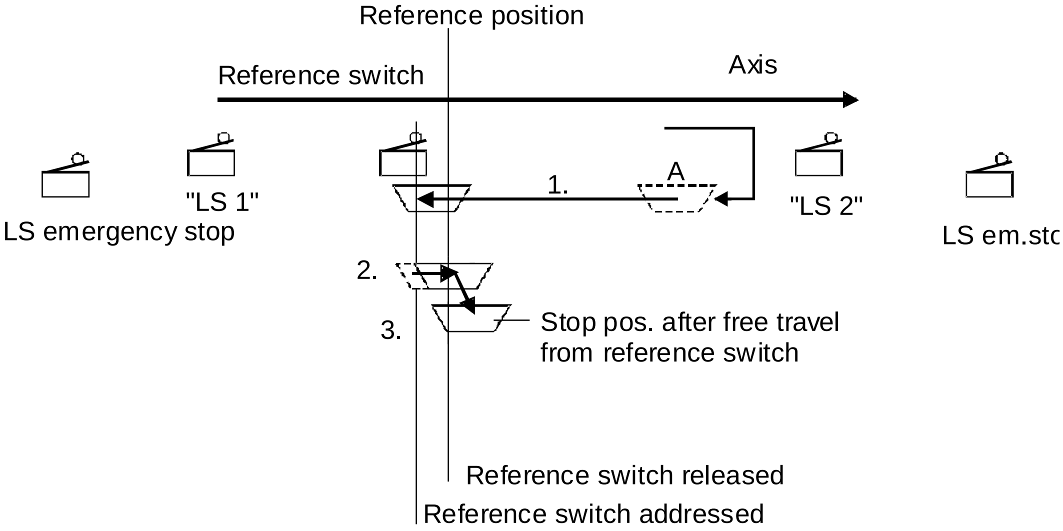
The axis to be referenced must have been initialized (FB Init).
(The following description refers to figure next page).

1. The search for the reference switch takes place at the velocities defined in parameters 6 and 7 and the acceleration defined in parameter 8. Search direction is defined in parameter 3. If the reference switch is not found and the axis encounters a limit switch (HW or SW), search direction is reversed.

2. If the reference switch is found, free travel commences. The direction of free travel is defined in parameter 4; the velocity of free travel is Vmin (parameter 6).

3. After the reference switch has been released, the axis stops and the position defined in parameter 2 is loaded as the absolute position.

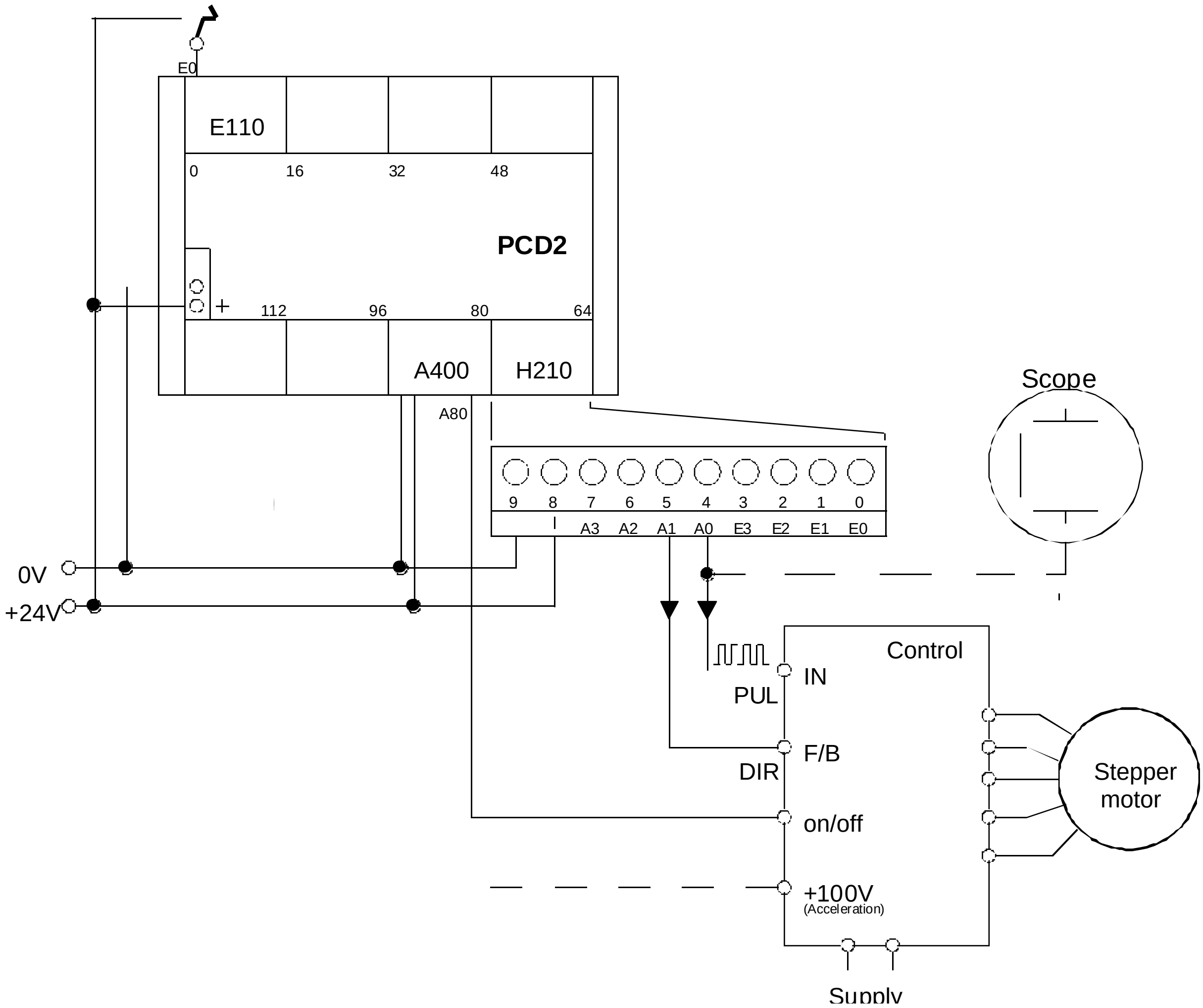
4. The module is configured with its original settings (from FB Init) and FB Home is exited.


Reference position
Reference switch
Axis
A
1.
"LS 1"
"LS 2"
LS emergency stop
LS em.stc
2.
3.
Stop pos. after free travel from reference switch
Reference switch released
Reference switch addressed

Instructions :

- Several modules can be referenced at a time on a single PCD.
- Break contacts must be used as the switches.
- During homing, limit switches are switched off internally. They are only used as direction reversing switches.
- Emergency stop is only active if configured in FB 'Init'.
- If no reference switch is found, error flag 'fHomeErr_x' (x = module no.) is set, absolute position becomes 0 and FB 'Home' is exited.
- The flag 'fEndHome_x' has to be reset in the user program to start the homing procedure. This flag will be set 'H' automatically on the end of the procedure.
- FB'Home' will be ended when homing has been successfully executed or when an error has been detected, this FB can be cancelled with a timeout (parameter 9). Its value corresponds to the time in seconds after which FB 'Home' will be aborted. Therefore, in addition to the 'fHomeErr_x' error flag, diagnostic register 'rDiag' is loaded with code 9 (for parameter 9) in its third byte (for FB Home). Parameter checking takes place as described in chapter 8.

6. Brief introduction



Minimum arrangement to operate a stepper motor for control using a PCD2.H210.

The individual elements are :

- PCD1 or PCD2 equipped with at least 1 PCD2.H210
  1 PCD2.E110
  1 PCD2.A400
- Suitable drive electronics
- Stepper motor, possibly with sliding carriage
- Supply device
- Oscilloscope (not an absolute necessity)

The parameters for 'Vmin', 'Vmax' and 'Acc' are to set according to the used drive and **cannot** be taken from the shown examples.

6.1 Getting started with programming in IL

The following minimal program is suggested to commission a stepper motor in the easiest way.

A proper user program should not contain wait loops. However, for the purposes of demonstrating the main instructions which drive a PCD2.H210, this example has been constructed with wait loops. In practice, a GRAFTEC or for the future a FUPLA structure should always be chosen for programs of this type.

Assume the stepper motor has the following characteristics:

Start-stop frequency :	40 Hz
Maximum frequency :	1000 Hz
Acceleration :	2000 Hz/sec
Steps/revolution :	48

Example task : After power-up of PCD, turning on input 'E0' should cause the stepper motor to move 4800 steps.

Individual parameters and base address settings should be as in section 7.1 below. The user program can then take the following form :

(A similar example but with moving forward and backward can be found as "example1.src" on the FB diskette).

The FBs (IL for PG4) are located on the diskette PCD9.H21E. To install the FBs on the PC follow the indications on the README.TXT on this diskette.

The number of modules (1) and the address of the PCD2.H210 module (64) is to indicate in the file 2D2H210_B.MBA:

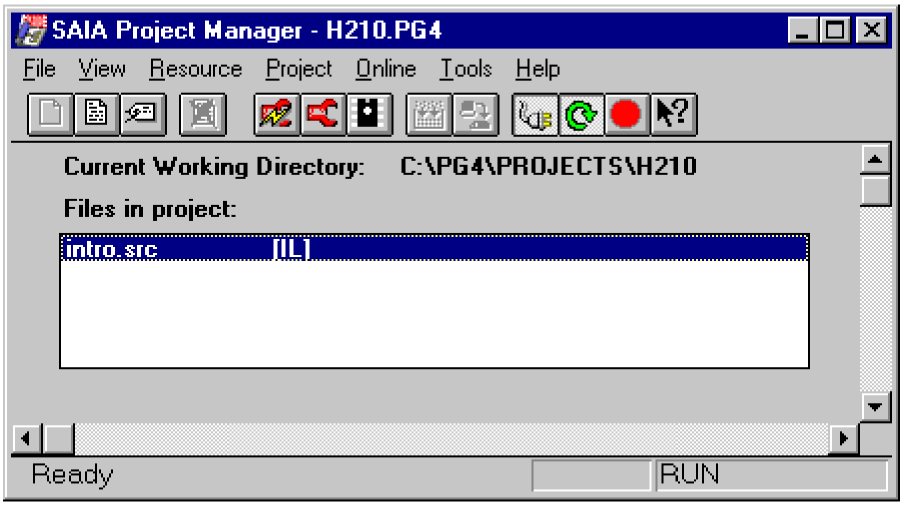
```
NbrModules   EQU   1      ; No. of H210 modules used (0...16)

BA_1         EQU   64     ; Base address of module 1
```

This file (2D2H210_B.MBA) must be located in project directory of this example, i.e. the file is to copy manually from the diskette to the actual project directory.

```
$include d2h210_b.equ
$group H210

       xob         16

       cfb         init        ; intitialisation
             k     1           ;  module number
                   0           ;  frequency range 0
                   0           ;  em stop: no
                   0           ;  limit switches: no
                   4           ;  Vmin
                   105         ;  Vmax
                   77          ;  acceleration

       ld    r     1           ; number of steps
                   4800

 exob
;-------------------------------------------------

       cob         0
                   0

st1:   sth   i     0           ; start ?
       jr    l     st1

       cfb         exec        ; load destination rel.
             k     1           ;  module number
                   LdDestRel   ;  command: load dest rel.
             r     1           ;  number of steps

       cfb         exec        ; start
             k     1           ;  module number
                   start       ;  command: start
                   rNotused    ;  empty register

wait:  sth         OnDest_1    ; finished ?
       jr    l     wait

st2:   sth   i     0           ; start ?
       jr    h     st2

       ecob

$endgroup
;-------------------------------------------------
```

As already mentioned a proper user program should not contain a wait loop. However, for the purposes of demonstrating the main instructions which drive a PCD2.H210, this example has been constructed with a wait loop. In practice, a GRAFTEC structure should always be chosen for programs of this type, see chapter 9.

The program is called "intro.src" and should be located in project "H210".



SAIA Project Manager - H210.PG4
File View Resource Project Online Tools Help

Current Working Directory: C:\PG4\PROJECTS\H210

Files in project:

intro.src [IL]

'Project' - 'Build' processes the program and downloads it to the PCD. Commissioning is done using the debugger.

If continuous display of position is required, e.g. on a PCD2.F5x0 display module, it is simplest to append a new COB to the existing COB 0 and edit as follows :

```
       cob         0
                   0
       . . .
       ecob

;--------------------------------------------------
$group H210

       cob         1
                   0

       cfb         exec         ; read position
             k     1            ;  module number
                   RdPosition   ;  command: RdPosition
             r     99           ;  destination register

       dsp   r     99           ; display dest. register

       ecob

$endgroup
;--------------------------------------------------
```

In this case it is necessary to incorporate a 'Next COB' (NCOB) instruction into each wait loop in COB 0, as otherwise the program only reaches COB 1 very rarely. The obvious problem here becomes, once again, that of spaghetti programming.

6.2 Getting started with programming in FUPLA

In preparation

Notes :

7. Programming

The standard "PG4" programming tools are used to create a user program to manage the PCD2.H.. counting and motion control modules. (To use the older programming tool "PG3", special FBs are available).

Programming is either in IL (instruction list) with FBs (function blocks) or in FUPLA with FBoxes (in preparation). The FBs can be obtained on diskette using reference PCD9.H21E.

Since motion control tasks always concern sequential processes, it is preferable if user programs are written in GRAFTEC, while individual steps and transitions can be edited either in IL with FBs or in FUPLA with FBoxes. User programs, however, can also be written purely in BLOCTEC or FUPLA.

7.1 Programming in IL with FBs

7.1.1 The IL package (Installation of the FBs)

The ordering code of the diskette is PCD9.H21E. The diskette contains the following directories:

- APPSDIR : contains all helps
- FB : contains the .SRC and .EQU files of the H210
- FBOX : contains the FBoxes for the H210
- PG3_FB : contains all files for the FBs of PG3
- PG4_FB : contains examples and the .MBA file
- Readme : contains general information's

The package is provided for the SAIA PG4 from version V2.0.70. For all other versions of PG4 the 'Readme' file is to consult. (The package also contains FBs for use with the older PG3, see 'Readme').

FBoxes for FUPLA are not yet available.

Installation of package for PG4

The simplest method of installation is with the PG4 program 'Setup Extra Files':

Insert the diskette PCD9.H21E into drive A:
<Start> <Programs> <SAIA PG4> <Setup Extra Files>. The FBs and 'Help' file are installed on the hard disk in the 'PG4' directory.

The following files are installed:

D2H210_B.SRC	FB source code	read-only file
D2H210_B.EQU	FB definitions	read-only file

These 2 files are copied from the diskette and located in the PG4 directory ...\PG4\FB.

FB_LIB.HLP	FB library data
D2H210_B.HLP	FB help

These 2 files are located in the directory A:\APPSDIR and are copied into the PG4 directory ...\PG4.

The file **D2H210_B.MBA** (module base addresses) must be copied **by hand** from the diskette into the current project directory.

File: **D2H210_b.mba** (mba = module base address)

```
;
; This file can be modified by the user
;
; Base addresses defined by the user
; ---------------------------------
$group H210
NbrModules      EQU     1       ; No. of H210 modules used (0...16)

;
; Module base addresses (only the used modules must be defined)

BA_1            EQU    32       ;Base address of module 1
BA_2            EQU     0       ;Base address of module 2
BA_3            EQU     0       ;Base address of module 3
BA_4            EQU     0       ;Base address of module 4
BA_5            EQU     0       ;Base address of module 5
BA_6            EQU     0       ;Base address of module 6
BA_7            EQU     0       ;Base address of module 7
BA_8            EQU     0       ;Base address of module 8
BA_9            EQU     0       ;Base address of module 9
BA_10           EQU     0       ;Base address of module 10
BA_11           EQU     0       ;Base address of module 11
BA_12           EQU     0       ;Base address of module 12
BA_13           EQU     0       ;Base address of module 13
BA_14           EQU     0       ;Base address of module 14
BA_15           EQU     0       ;Base address of module 15
BA_16           EQU     0       ;Base address of module 16
$endgroup
```

The number of PCD2.H210 modules must be specified and the hardware base addresses of PCD2.H210 modules used should then be entered.

Since the '.mba' file does not appear in Project Manager’s file list, a text editor (e.g. SEDIT32) must be used for modification.

The modules are to be numbered successively beginning with 'BA_1'. When e.g. three H210 modules should be used in a project, the 'BA_1', 'BA_2' and 'BA_3' are to be used. The places in the PCD can be chosen free.

Example:

```
NbrModules      EQU     3       ; No. of H210 modules used (0...16)
;
; Module base addresses (only the used modules must be defined)

BA_1            EQU    64       ;Base address of module 1
BA_2            EQU   208       ;Base address of module 2
BA_3            EQU   112       ;Base address of module 3
BA_4            EQU     0       ;Base address of module 4
BA_5            EQU     0       ;Base address of module 5
```

The base addresses of registers, flags and FBs are assigned automatically and can be viewed in the resource list under 'View' - 'Resource List'.

The project to be created should have project name "TEST-H2" and the actual user program module should be entitled "move-01.sfc". The files are arranged like this :

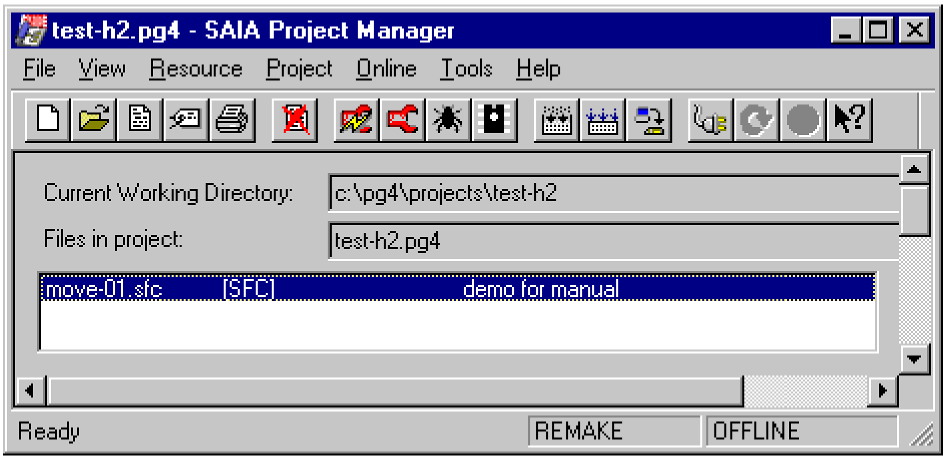
```
C:\PG4 \FB          \D2H210_b.equ
                    \D2H210_b.src
       \...
       \FBOX        \...
       \GALEP3      \...
       \PROJECTS    \FUP_E              (Demo example PG4)
                    \GRAF_E             (Demo example PG4)
                    \TEST-H2    \D2H210_b.mba
                                \move-01.sfc
       \...
       \D2H210_b.hlp
```

The user program for the H210 part is structured as follows :

```
$include D2H210_b.equ
$group H210

XOB     16


PCD-Code


ecob
$endgroup
```

If the program is written in GRAFTEC, the assembler directives "$include" and "$group" are placed in the first step (ST), normally the initial step (IST). "$endgroup" comes at the end of the last transition (TR).

If everything has been correctly installed, the user program edited and all parameters defined, the program can be processed and downloaded to the PCD with the 'Project' - 'Build' command.


test-h2.pg4 - SAIA Project Manager
File View Resource Project Online Tools Help

Current Working Directory: c:\pg4\projects\test-h2

Files in project: test-h2.pg4

move-01.sfc [SFC] demo for manual

Ready REMAKE OFFLINE

7.1.2 The individual FBs

The complete package consists of three main FBs with parameters :

• INIT	Initialization	FB with 7 parameters
• EXEC	Execution	FB with 3 parameters
• HOME	Homing position	FB with 9 parameters

Calling FB INIT is always done as follows :

```
CFB        init
     k     1           ; Param. 1: module number     (k 1 - k 16)
           1           ; Param. 2: frequency range   (0 - 3)
           0           ; Param. 3: emergency stop    (0=off/1=on)
           0           ; Param. 4: limit switch      (0=off/1=on)
           1           ; Param. 5: Vmin              (1 - 255)
           100         ; Param. 6: Vmax              (1 - 255)
           20          ; Param. 7: acceleration      (1 - 255)
```

For some typical examples, calling FB 'EXEC' appears as follows :

```
CFB        exec
     k     1           ; Param. 1: module number     (k 1 – k 16)
           LdDestRel   ; Param. 2: function
     r     777         ; Param. 3: value (from source register)

CFB        exec
     k     1           ; Param. 1: module number     (k 1 - k 16 )
           start       ; Param. 2: function
           rNotUsed    ; Param. 3: not used

CFB        exec
     k     1           ; Param. 1: module number     (k 1 - k 16)
           RdPosition  ; Param. 2: function
     r     1000        ; Param. 3: value (in destination register)
```

Three parameters must always be defined, even if only two are required for a function. The third parameter can be defined as 'rNotUsed' or as any register.

<u>Commands (functions) for FB Exec :</u>

```
LdVmin             ; load min. frequency
LdVmax             ; load max. frequency
LdAcc              ; load acceleration

LdDestRel          ; load Destination relative

LdDestAbs          ; load Destination absolute

Start              ; initialise and start motion
Stop               ; stop motion
Continue           ; continue motion after stop
```

```
SetOut2           ; set      Output 2
ResOut2           ; reset    Output 2
SetOut3           ; set      Output 3
ResOut3           ; reset    Output 3

RdPosition        ; read remaining Position
RdIdent           ; read module identification
```

A FB 'SetZero' for the definition of any point as zero position does not exist. The register 'rPosAbs_x' can be used by loading the position as

```
Ld     rPosAbs_x
       0
```

Note: the axis must be stopped to execute this command.

Calling FB 'Home' always appears as follows :

```
CFB        HOME
      k    1      ; Param. 1: module number                (k 1 - k 16)
      r    888    ; Param. 2: ref. position                (R)
           1      ; Param. 3: search direction             (0 = down/1 = up)
           0      ; Param. 4: ref. switch leaving direction (0 = down/1 = up)
           2      ; Param. 5: frequency range              (0 – 3)
           10     ; Param. 6: Vmin                         (1 - 255)
           100    ; Param. 7: Vmax                         (1 - 255)
           20     ; Param. 8: acceleration                 (1 - 255)
           30     ; Param. 9: timeout in sec.              (1 – 65535
                                                           0 = no timeout)
```

General definitions :

```
rDiag               ; error flag if parameter value wrong
fPar_Err
```

Module-specific definitions:

```
fHomeErr_x          ; Error during home procedure
fEndHome_x          ; Low during homing
rPosAbs_x           ; absolute position
EmergTrig_x         ; Emergency stop Triggered
OnDest_x            ; Destination OK
LSxTrig_x           ; Limit Switch 1 or 2 Triggered
Emerg_x             ; Emergency stop
LS1_x               ; Limit Switch 1
REF_x               ; REFerence switch
LS2_x               ; Limit Switch 2
```

"_x" corresponds to module number

The effective addresses of registers and flags are listed in the 'Projectname.MAP' file (for debug purposes).

7.1.3 Parameter definition

Values for 'Vmin', 'Vmax' and 'Acceleration' can be taken from the following tables. For each of these parameters values from 1 to 255 can be selected. Four frequency ranges 0 to 3 are available. Values for setting task parameters should always be taken from a single range.

Values for "Vmin" and "Vmax" (Vmin and Vmax in Hz)

Value	**Range 0**	**Range 1**	**Range 2**	**Range 3**
1	9.5	19	38	76
2	19	38	75	152
3	28	57	114	229
4	38	76	153	305
5	48	95	191	381
6	57	114	229	458
8	76	153	305	610
10	95	191	381	763
15	143	286	572	1144
20	190	381	763	1526
25	238	477	954	1907
30	286	572	1144	2289
40	381	763	1526	3052
50	477	964	1907	3814
60	572	1144	2289	4577
70	667	1335	2670	5340
80	763	1526	3052	6103
90	858	1717	3433	3866
100	954	1907	3815	7629
110	1049	2098	4196	8392
120	1144	2289	4577	9155
130	1240	2480	4959	9918
140	1335	2670	5340	10680
150	1430	2861	5722	11443
160	1526	3052	6103	12206
170	1621	3243	6485	12969
180	1716	3433	6866	13732
190	1812	3624	7248	14495
200	1907	3815	7629	15258
210	2002	4006	8010	16021
220	2098	4196	8392	16784
230	2193	4387	8773	17546
240	2288	4578	9155	18903
250	2384	4769	9536	19072
255	2431	4864	9727	19454

Example for values not shown in the above table :

for ex. 1000Hz : $\text{value} = \frac{1000}{9.5} = 105 \rightarrow \text{range } 0$

Values for acceleration (in Hz/sec)

Value	Range 0	Range 1	Range 2	Range 3
1	152939	305911	611821	1223642
2	76470	152955	305911	611821
3	50980	101970	203940	407881
4	38235	76478	152955	305911
5	30588	61182	122364	244728
6	25489	50985	101970	203940
8	19117	38239	76478	152955
10	15294	30991	61162	122364
15	10196	20394	40788	81776
20	7647	15296	30591	61162
25	6118	12236	24473	48946
30	5098	10197	20394	40788
40	3823	7648	15296	30591
50	3059	6118	12236	24473
60	2549	5099	10197	20394
70	2185	4370	8740	17481
80	1912	3824	7648	15296
90	1699	3399	6798	13596
100	1529	3059	6118	12236
110	1390	2781	5562	1124
120	1274	2549	5099	10197
130	1176	2353	4706	9413
140	1092	2185	4370	8740
150	1020	2039	4079	8158
160	956	1920	3824	7648
170	900	1800	3599	7198
180	850	1700	3399	6798
190	805	1610	3220	6440
200	765	1530	3059	6118
210	728	1457	2913	5826
220	695	1391	2781	5562
230	665	1330	2660	5320
240	637	1275	2549	5099
250	612	1224	2447	4895
255	600	1200	2400	4800

Example : Values are to be defined for Vmin = 100 Hz,
Vmax = 3000 Hz and an acceleration of 6000 Hz/sec.

The range chosen is always the one with the lowest attainable frequencies, for our case "Range 1".

for Vmin → value 5 (95 Hz)
for Vmax → value 150 - 160 = approx. 157
for acceleration → value approx. 51

7.2 Programming in FUPLA with FBoxes

In preparation

7.3 Programming in GRAFTEC with FBoxes

In preparation

8. Error handling and diagnosis

8.1 Definition errors checked by the assembler

The following definition errors in file D2H210_b.MBA are checked during assembly :

- If the number of modules (NbrModules) is lower than 1, no code is assembled and the following warning is displayed in the 'Make' window :

"Remark : No H210 used (NbrModules = 0 in D2H210_B.MBA)"

- If the number of modules (NbrModules) is greater than 16, no code is assembled and the following error message is displayed in the 'Make' window :

"Error : more than 16 Modules H210 defined (NbrModules = 0...16)"

- If an incorrect instruction code is used for FB 'Exec' (e.g. RdIdenti instead of RdIdent), the assembler reports an error :

"Symbol not defined 'H210.RdIdenti'"
(generating the printout 'H210' from $group h210)

- If the definition $group H210 is absent, then for each instruction and each register/flag used in the program the assembler reports :

"Symbol not defined"

8.2 Error handling in run

8.2.1 Wrong parameter

In FB 'Exec' only the command code is checked. Parameter 1 (module no.) and parameter 3 (source/destination register) are not checked, to avoid making execution times longer.

In FBs 'Init' and 'Home' the values of all parameters are checked to verify whether they fall within the permitted range (e.g. frequency range = 0, 1, 2, 3). If a parameter is outside a range, it is set to the minimum value (except acceleration which is set to 255), the error flag 'fPar_Err' is set and diagnostic register 'rDiag' is loaded with the corresponding error code.

Flag 'fPar_Err' is not reset inside the FB. This should take place in XOB 16 or in the 'Init' step.

The error code is composed as follows :

```
rDiag    bit 31 . . . . . . . 24 23 . . . . . . . 16 15 . . . . . . .  8 7 . . . . . . . 0
                \  Reserve  /     \  FB No.  /     \  Par. No.  /   \ Mod. No./
                                  (Init     =  FB 1)
                                  (Exec   =  FB 2)
                                  (Home =  FB 3)
```

Example : If the frequency range parameter 5 in FB Home of Module 2 is incorrect (>3), register 'rDiag' is set to 00 03 05 02 hex.

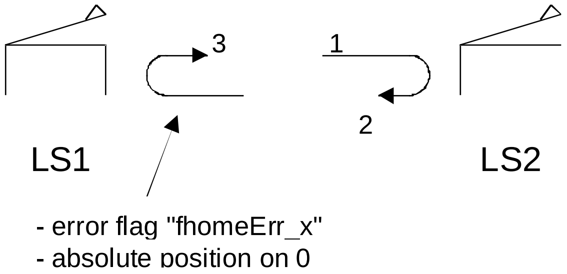
The diagnostic register is overwritten with each incorrect parameter and always contains the last error. It should therefore be evaluated as soon as flag 'fPar_Err' signals an error. The absolute addresses of 'rDiag' and 'fPar_Err' can be viewed in file 'project.MAP'. This can be useful during commissioning with the debugger to locate an error :

- Run until flag 'fPar_Err' = H
- Display register 'rDiag' hex
- Delete flag 'fPar_Err'

8.2.2 Error during homing

If the home position has not been found (e.g. because of a faulty reference switch), the 'fHomeErr_x' error flag is set, motion stops, absolute position is set at 0 and FB 'Home' returns.

Reference switch not existing or wrong connected :



If FB 'Home' returns because the specified timeout has been exceeded, diagnostic register 'rDiag' is additionally loaded with code 9 as the parameter number (timeout is parameter 9).

The flag 'fHomeErr_x' is defined for each module (_x is the module no.) and is reset at the start of each 'Home' FB. This flag should be queried after each call of the 'Home' FB to make sure that the axis is correctly referenced.

Example :

```
CFB    Home         ; Homing axis 3
     K 3
       .
       .
       .

STH    fHomeErr_3   ; Query Home error flag
                    ;   of axis 3
CFB  H Errorhandl   ; Call (user specific) FB,
                    ;   if 'fHomeErr_3' is high

CFB    Exec         ; Motion 1

...
```

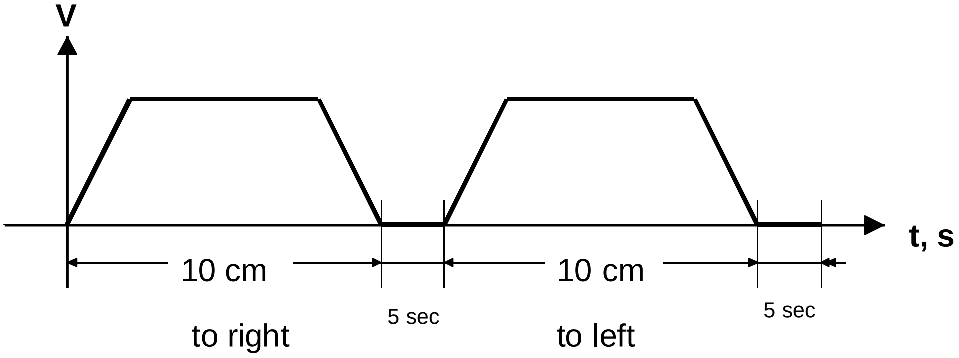
Notes :

9. User examples

9.1 Typical example in GRAFTEC and IL

Based on the example in chapter 6, "Getting started", when PCD input 0 ("Start") is switched on, a sliding carriage driven by a stepper motor will travel from a starting position to the right for 10 cm and, after a 5 second pause, will return to the starting position.

If the "Start" input remains switched on, this sequence is to start again from the beginning after a further 5 second pause.



As a variation, the motion of the carriage is to be stopped with PCD input 1 ("Stop") and made to run on with PCD input 2 ("Continue"), retaining acceleration and braking ramps.

Some data on the working model:

Steps/revolution :	48
Spindle gradient :	1 mm/U
Vmin (Start-Stop) :	40 Hz
Vmax :	1000 Hz
Acceleration :	2000 Hz/sec

The project is entitled "TEST-H2"

The actual user program is written in GRAFTEC and given the name "MOVE-01.SFC".

The parameters for 'Vmin', 'Vmax' and 'Acc' are to set according to the used drive and **cannot** be taken from the shown examples.

The following steps should be executed :

- Copy file "D2H210_b.mba" to project directory "test-h2" and define the base address of the PCD2.H210 module, e.g. 64.

- Define parameters for velocities and acceleration :

  According to section 7.1.3 range 1 is selected.

Vmin	→	value 4
Vmax	→	value approx. 105
Acceleration	→	value approx. 75

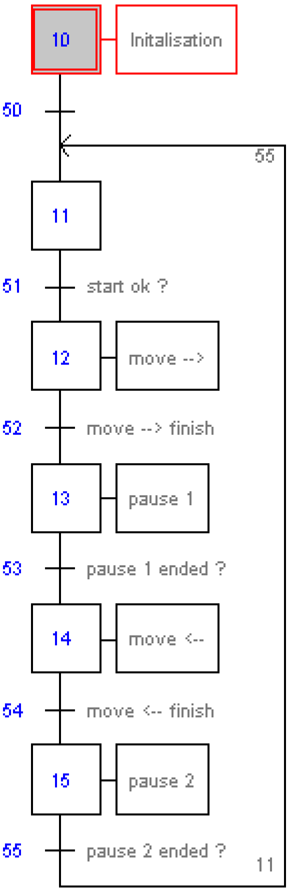
- Define the number of steps :

  100 mm x 48 steps = 4800 steps

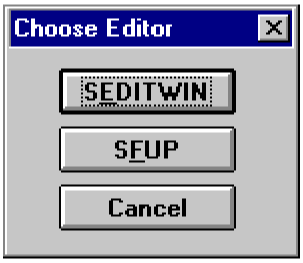
If operating in "relative" mode, 4800 is entered for the first movement (to the right) and -4800 for the return journey.

If operating in "absolute" mode, 4800 is also entered for the first movement, but 0 (zero) is programmed for the return journey.

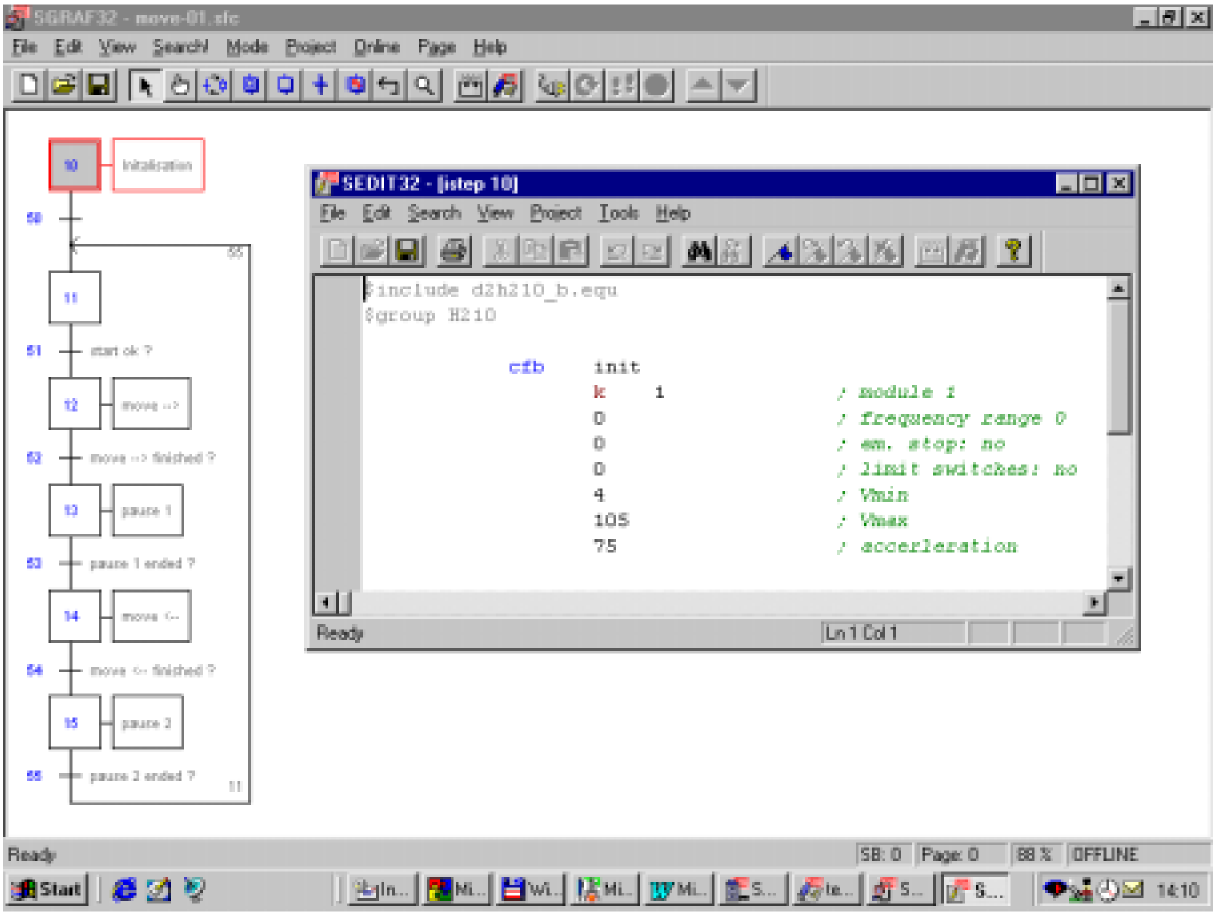
- Write GRAFTEC structure with comments



Code individual steps and transitions with SEDITWIN



(Editing with SFUP is described in section 9.2).



```
$include d2h210_b.equ
$group H210

            cfb     init
                    k     1               ; module 1
                    0                     ; frequency range 0
                    0                     ; em. stop: no
                    0                     ; limit switches: no
                    4                     ; Vmin
                    105                   ; Vmax
                    75                    ; accerleration
```

The code of individual steps and transitions is edited in the same way.

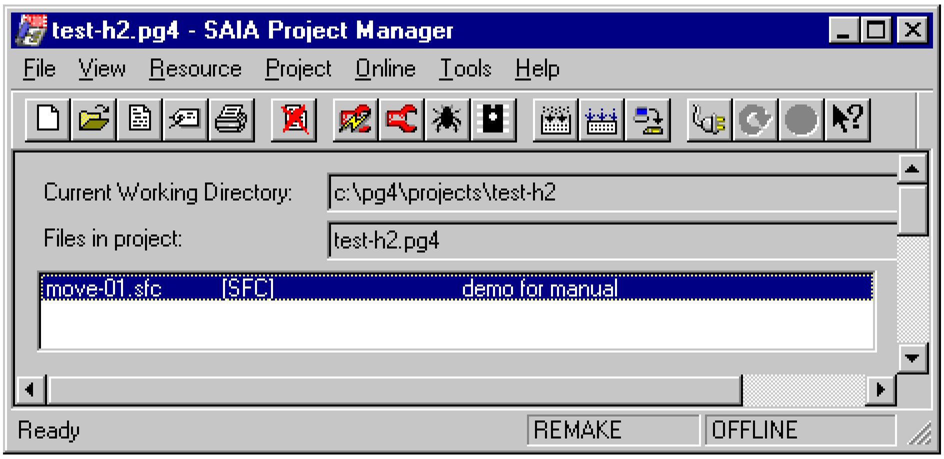
File "move-01.sfc" can, for example, be selected in the notepad editor and printed out. See following pages. Together with the printout of the GRAFTEC structure, the code file printout forms the documentation of the program.

```
 SB    0
;--------------------------------
 IST   10          ;Initalization
       O 50
$include d2h210_b.equ
$group H210

 cfb         init
       k     1     ; module 1
             0     ; frequency range 0
             0     ; em. stop: no
             0     ; limit switches: no
             4     ; Vmin
             105   ; Vmax
             75    ; acceleration
 EST   ;10
;--------------------------------
 ST    11
       I 50
       I 55              ;pause 2 ended ?
       O 51              ;start ok ?
 EST   ;11
;--------------------------------
 ST    12          ;move -->
       I 51              ;start ok ?
       O 52              ;move --> finished ?
 ld    r     1
             4800

 cfb         exec
       k     1     ; module 1
             lddestrel
       r     1

 cfb         exec
       k     1     ; module 1
             start
             rnotused
 EST   ;12
;--------------------------------
 ST    13          ;pause 1
       I 52              ;move --> finished ?
       O 53              ;pause 1 ended ?
 ld    t     1
             50    ; 5 sec
 EST   ;13
;--------------------------------
 ST    14          ;move <--
       I 53              ;pause 1 ended ?
       O 54              ;move <-- finished ?
 ld    r     1
             -4800

 cfb         exec
       k     1     ; module 1
             lddestrel
       r     1

 cfb         exec
       k     1     ; module 1
             start
             rnotused
 EST   ;14
```

```
;-------------------------------
 ST    15          ;pause 2
       I 54              ;move <-- finished ?
       O 55              ;pause 2 ended ?
 ld    t     1
             50    ; 5 sec
 EST   ;15
;-------------------------------
 TR    50
       I 10              ;Initialization
       O 11
 ETR   ;50
;-------------------------------
 TR    51          ;start ok ?
       I 11
       O 12              ;move -->
 sth   i     0
 ETR   ;51
;-------------------------------
 TR    52          ;move --> finished ?
       I 12              ;move -->
       O 13              ;pause 1
 cfb         exec
       k     1     ; module 1
             rdposition
       r     0

 dsp   r     0     ; display module

 sth         ondest_1
 ETR   ;52
;-------------------------------
 TR    53          ;pause 1 ended ?
       I 13              ;pause 1
       O 14              ;move <--
 cfb         exec
       k     1     ; module 1
             rdposition
       r     0

 dsp   r     0     ; display module

 stl   t     1
 ETR   ;53
;-------------------------------
 TR    54          ;move <-- finished ?
       I 14              ;move <--
       O 15              ;pause 2
 cfb         exec
       k     1     ; module 1
             rdposition
       r     0

 dsp   r     0     ; display module

 sth         ondest_1
 ETR   ;54
```

```
;-------------------------------
 TR    55          ;pause 2 ended ?
       I 15              ;pause 2
       O 11
 cfb         exec
       k     1     ; module 1
             rdposition
       r     0

 dsp   r     0     ; display module

 stl   t     1

$endgroup
 ETR   ;55

 ESB   ;0
```

The project manager’s 'Project' -'Build' command is used to process the program and download it into the PCD.



Current Working Directory: c:\pg4\projects\test-h2

Files in project: test-h2.pg4

move-01.sfc [SFC] demo for manual

Ready REMAKE OFFLINE

If a PCD2.F5x0 display module is fitted, direct viewing is possible not only of axis motion, but also counter status.

Other similar examples can be taken from the diskette PCD9.H21E.

Variation with ability to stop and run-on :

If, as indicated in the task description, a motion is to be stopped with PCD input 1 and made to run on again with PCD input 2, transitions 52 (move →) and 54 (move ←) should be programmed as follows :

```
;--------------------------------
 TR    52                 ;move --> finished ?
       I 12                     ;move -->
       O 13                     ;pause 1
 cfb         exec
       k     1            ; module 1
             rdposition
       r     0

 dsp   r     0            ; display module

 sth   i     1            ; stop
 dyn   f     1
 set   f     10           ; stop-store flag
 cfb   h     exec
       k     1            ; module 1
             stop
             rnotused

 sth   i     2            ; continue
 dyn   f     2
 res   f     10           ; stop-store flag
 cfb   h     exec
       k     1            ; module 1
             continue
             rnotused

 sth         ondest_1
 anl   f     10           ; stop-store flag
 ETR   ;52
```

In principle, TR 52 and 54 of this example only wait for the end of the motion (sth ondest_1). It is a peculiarity of GRAFTEC that, after each **unfulfilled** TR, the program returns to the calling COB and, during the next program cycle, processes the unfulfilled TR again **in full**. This is used in the program to refresh the display continuously.

If it is then necessary to interrupt (stop) pulse output during a motion, this must also happen within the TR which is awaiting the end of that motion. If the Stop command is activated, the "ondest_x" flag also becomes high and the TR would be regarded as fulfilled. The Stop command is therefore stored intermediately in a stop store flag and the logical state of this flag is used to decide whether the whole path has really been travelled. This flag is reset by the «Continue» function.

When required, this is a way of interrupting a motion cleanly, i.e. with braking and acceleration ramps and without losing steps.

Application of the reference and limit switches

This example is on the diskette PCD9.H21E.

```
; *************************************
; * Example program for FB's H210 PG4 *
; *************************************
;
; FileName :    H210_Ex4.SRC
; Autor :       Nguyen T.D.  03.01.99
;
; System requirements:
; 1 input module on address 0,
; 1 H210 on the base address defined in the file D2H210_B.MBA
;
; Description:
;
; - This program shows how to work with the HOMING Procedure
;   (Please refer also on the detail description in the manual)
;   If the input I3 is H, the program executes the HOMING
;   Procedure.
;   The limit switches (LS) for the HOMING Procedure must be
;   wired physically for LS simulation if there are no LS on
;   the axis: E1 = I5 = LS1, E2 = I6 = LSRef, E3 = I7 = LS2
;   If there are LS on the axis, the user has to connect LS1,
;   LSRef and LS2 as mentionned above.
; - This program has to be linked together to the examples,
;   either H210_Ex2.SFC or H210_Ex3.SFC for getting other
;   motion program. For starting the motion program, please use
;   the inputs set in the respective program (e.g. I0 = start,
;   I1 = stop, I2 = continue) and refer to the respective
;   description in chapter 9 of the manual.
; - The user must adapt the parameters of the FB INIT in the
;   Init Step of the examples H210_Ex2.SFC or H210_Ex3.SFC
;   (Set Emergency stop = 1, and Limit switch = 1).
; - The working range is limited by means of LS1 and LS2, out
;   of this working range, the flag LS1Trig1 is high and the
;   output A3 of the module is set. The output A3 is also set
;   high if the Emergency Stop ic activated (E0=0). In the
;   pratice, an normal output can be set in the user program
;   for alarm or warning purpose or for other specific purpose
;   in such situations.
; - The Emergency stop input is simulated with E0 = I4.
;
```

```
$include D2H210_B.EQU    ; for PG4 V2.0 the complete path
                         ;  is no more necessary
$group H210

XOB        16            ; startup block

CFB        Exec          ; FB Exec
         K 1             ;  module number
           ResOut3       ;  reset output A3
           rNotUsed      ;  empty register
EXOB

;--------------------------------------------------------------

COB        0             ; cyclic block
           0

LD       R 100           ; Load Offset position for REF point
           0

STH      I 3             ; Start HOMING Procedure
DYN      F 1000
CFB      H HOME          ; FB Home
         K 1             ;  module number
         R 100           ;  offset register
           1             ;  search direction (0=Neg.,1=Pos.)
           0             ;  free travel direction (0=Neg.,1=Pos)
           1             ;  frequency range
           16            ;  Vmin
           80            ;  Vmax
           100           ;  acceleration
           400           ;  time out (Max. time in sec before
                         ;   stop occurs

STH        LSxTrig_1     ; LS1 or LS2 overdrived ?
ORH        EmergTrig_1   ; or Emergency Stop activated
CFB      H Exec          ; FB Exec
         K 1             ;  module number
           SetOut3       ;  command: set output A3
           rNotUsed      ;  empty register
ECOB

$endgroup
;--------------------------------------------------------------
```

Notes

9.2 Typical example in pure FUPLA

In preparation

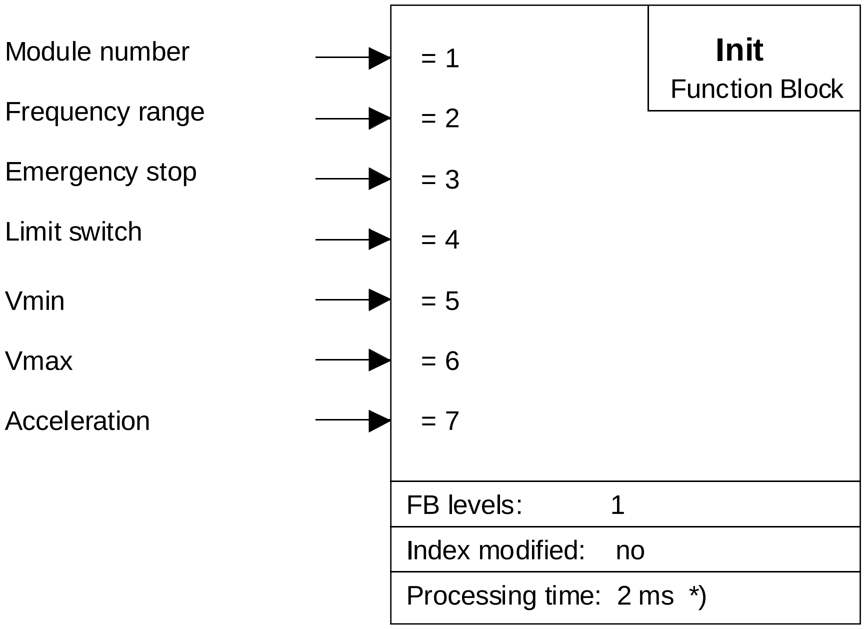
9.3 Typical example in GRAFTEC and FUPLA

In preparation

Appendix A. Summary of all software elements for programming in IL

Function block 'Init'

Init **FB :** Initialization of an H210 module



Function description :

This FB defines the settings of the PCD2.H210 module and reads the base address from file D2H210_B.MBA.

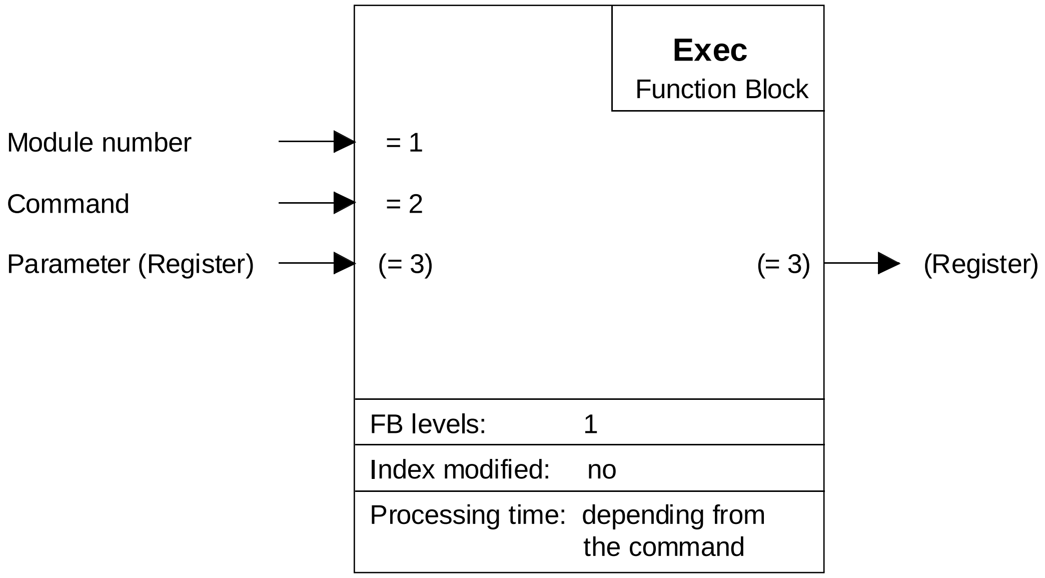
Values for Vmin, Vmax and Acc can be modified at any time and are not considered maximum values.

Parameter 1 must be given as a K constant and all other parameters as integer numbers.

Par.	Designation	Type	Format	Value	Comment
= 1	Module number	K	K n	K 1 – K 16	
= 2	Frequency range		integer	0 – 3	0 = 9.5 – 2431 Hz 1 = 19 – 4864 Hz 2 = 38 – 9727 Hz 3 = 76 – 19454 Hz
= 3	Emergency stop config.		integer	0 / 1	0 = off / 1 = on
= 4	Limit switch config.		integer	0 / 1	0 = off / 1 = on
= 5	Vmin (Start-stop freq.)		integer	1 – Vmax	Entry checked
= 6	Vmax (frequency after acceleration ramp)		integer	Vmin – 255	Entry checked
= 7	Acceleration		integer	255 – 1	Entry checked

Function block 'Exec'

Exec **FB :** Execution of a command for the H210 module



Function description :

This FB is used to send commands to the PCD2.H210 module.

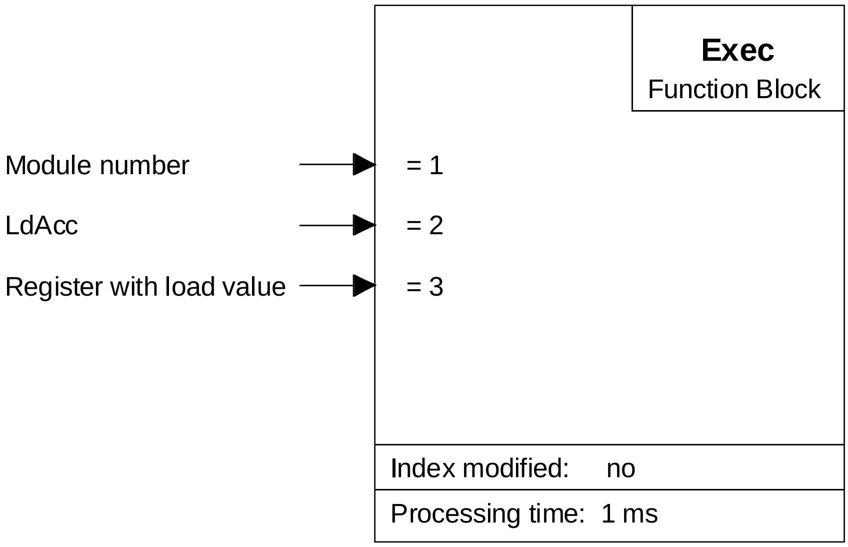
Module number (parameter 1) must be a K constant (K 1…K 16).
The base address is defined in file 'D2H210_B.MBA'.
These FBs support up to 16 x PCD2.H210 modules per PCD system.

Individual commands (parameter 2) are described in the following pages.

The parameter of a command (e.g. acceleration value for command LdAcc) is transferred to a register (parameter 3). If a command does not need a parameter (e.g. Start) any register or 'rNotUsed' can be presented.

Individual instructions for PCD2.H210 (FB parameters)

LdAcc **Command :** Load acceleration



Function description :

This command loads acceleration to the input register.

This value will apply from the next 'Start' command. Acceleration and braking ramps are symmetrical. Acceleration depends on choice of frequency range (see table, section 7.1.3). The following formula can be used to calculate effective acceleration :

$$\text{eff. accel. (frequency range = 3) } [\text{steps/s}^2] = \frac{76.30\ [\text{steps/s}] * 16000\ [1/\text{s}]}{\text{acceleration value}}$$

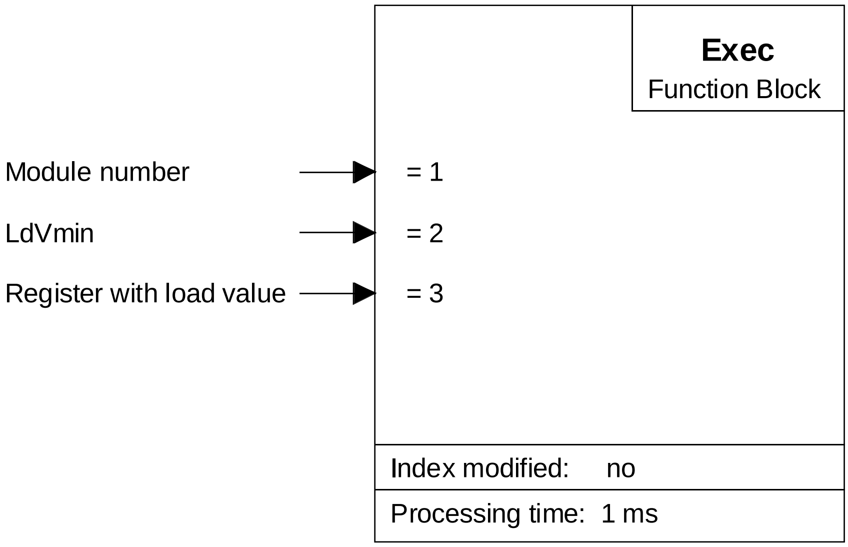
etc. with 38.15 and 19.08 [steps/s] until,

$$\text{eff. accel. (frequency range = 0) } [\text{steps/s}^2] = \frac{9.54\ [\text{steps/sec}] * 16000\ [1/\text{s}]}{\text{acceleration value}}$$

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: LdAcc				
= 3	Acceleration	R	integer	255 - 1	255 = min ; 1 = max Values outside the range are not intercepted.

LdVmin **Command :** Load start-stop frequency



Function description :

This command loads the start-stop frequency (Vmin) to the input register. This value applies from the next 'Start' command.

Frequency grading, i.e. the steps from one frequency stage to the next, depends on the choice of range :

For range 0 (9.5 to 2431 Hz)	→	resolution totals 9.54 Hz
For range 1 (19 to 4864 Hz)	→	resolution totals 19.08 Hz
For range 2 (38 to 9727 Hz)	→	resolution totals 38.15 Hz
For range 3 (76 to 19454 Hz)	→	resolution totals 76.30 Hz.

See also table in section 7.1.3

Calculation of effective start-stop frequency :

For range = 0 : Start-stop frequency = 9.54 Hz * value Vmin

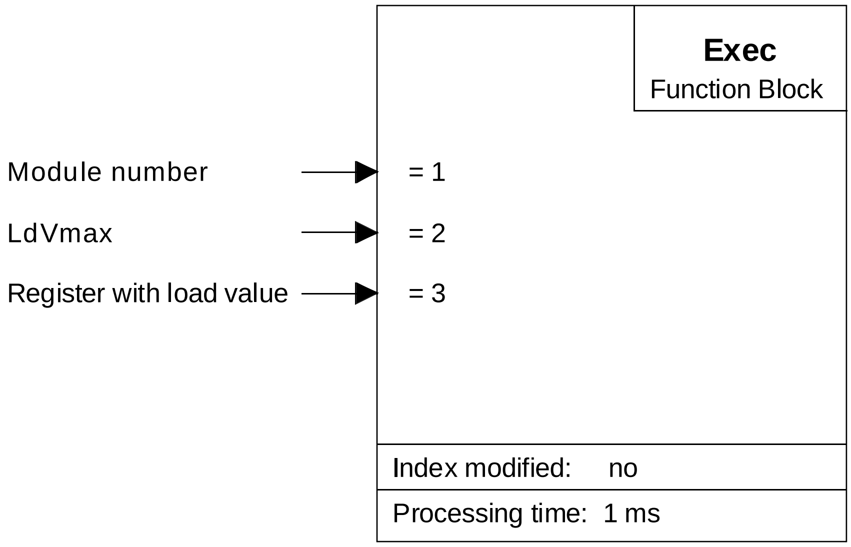
etc. for 19.08 Hz and 38.15 Hz until,

For range = 3 : Start-stop frequency = 76.30 Hz * value Vmin

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: LdVmin				
= 3	Start-stop frequency	R	integer	1 - Vmax	Values outside the range are not intercepted

LdVmax **Command :** Load maximum velocity



Function description :

This command loads the maximum frequency (Vmax) attainable after an acceleration ramp to the input register. This value applies after the next Start command.

Frequency grading, i.e. the steps from one frequency stage to the next, depends on the choice of range :

For range 9.5 to 2431 Hz	→	resolution totals 9.54 Hz
For range 19 to 4864 Hz	→	resolution totals 19.08 Hz
For range 38 to 9727 Hz	→	resolution totals 38.15 Hz
For range 76 to 19454 Hz	→	resolution totals 76.30 Hz.

See also table in section 7.1.3

Calculation of effective frequency :

For range = 0 : Frequency = 9.54 Hz * value Vmax

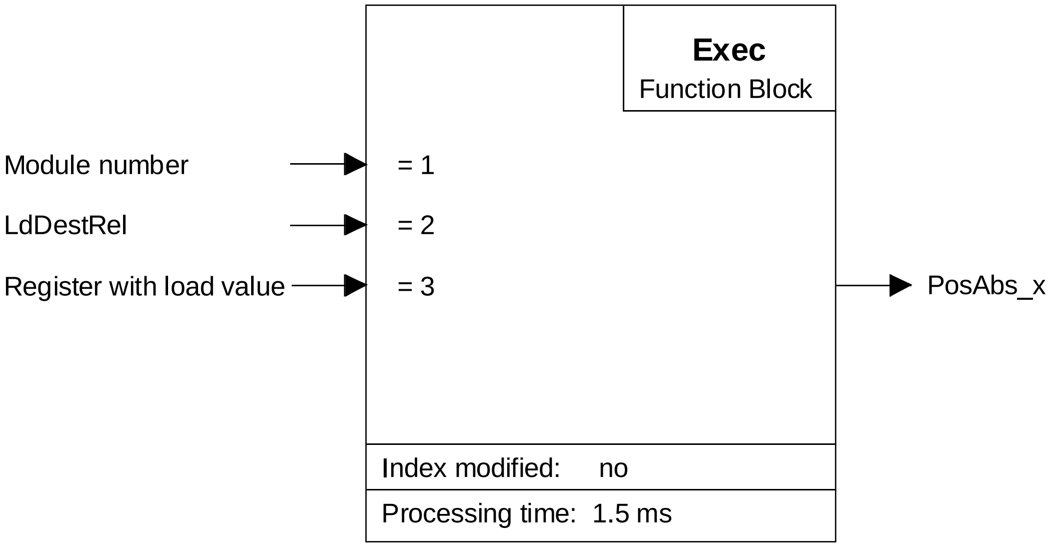
etc. for 19.08 Hz and 38.15 Hz until,

For range = 3 : Frequency = 76.30 Hz * value Vmax

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: LdVmax				
= 3	Maximum frequency	R	integer	Vmin - 255	Values outside the range are not intercepted

LdDestRel **Command :** Load relative travel



Function description :

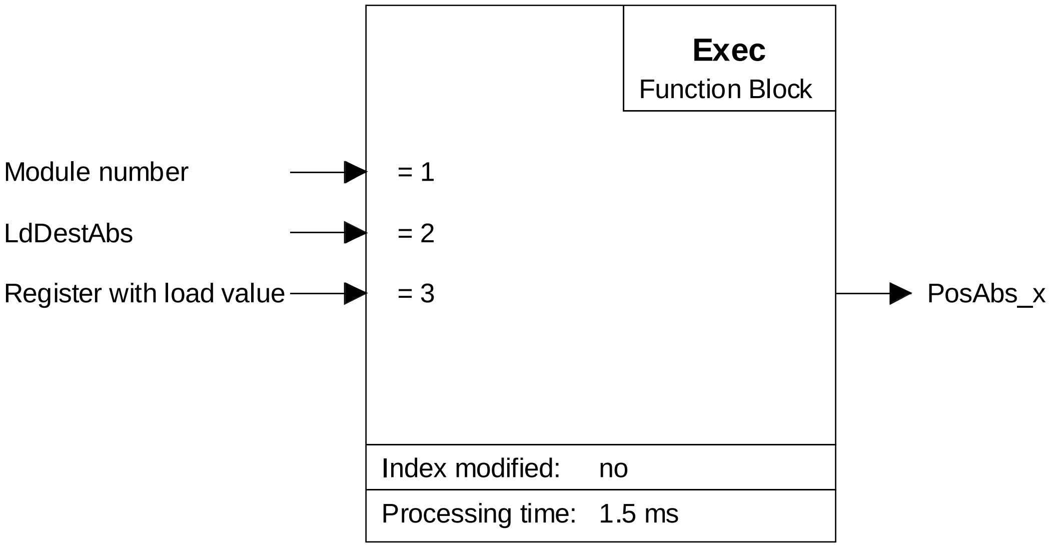
This command loads the number of steps for the next movement to the input register. This value applies from the next Start command. The command always functions in relative mode. The value entered is always an integer (positive or negative) and not bigger than 2^{24}.

This command also calculates the absolute position (register 'PosAbs_x') to be reached after execution of the movement.

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
=-2	Command: LdDestRel				
= 3	Target (relative)	R	integer	24 bit	± 16777215
PosAbs_x	Absolute position	R	integer	32 bit	-2^{31} .. $+2^{31}$ -1

LdDestAbs **Command :** Load absolute destination



Function description :

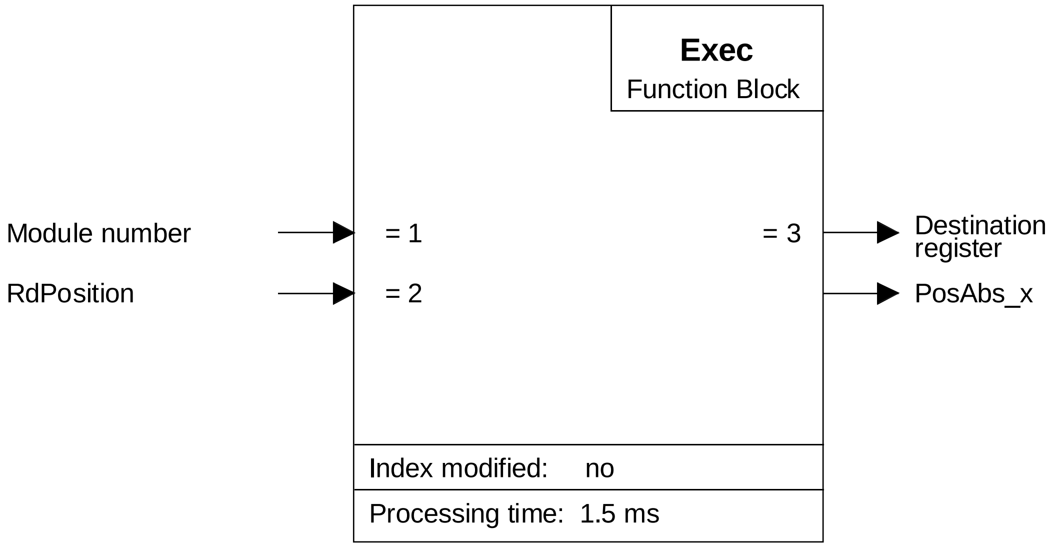
This command loads the destination for the next movement to the input register. This value applies from the next Start command. The command calculates relative movement to final position. The value entered is always a positive integer and not bigger than 2^{24}.

This command also calculates the absolute position (register 'PosAbs_x') to be reached after execution of the movement.

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: LdDestAbs				
= 3	Target (absolute)	R	integer	32 bit	-2^{31} .. $+2^{31}$ -1 but max. 24 bit as relative movement
PosAbs_x	Absolute position	R	integer	32 bit	-2^{31} .. $+2^{31}$ -1

RdPosition **Command :** Read actual (current) position



Function description :

This command reads the module counter. The counter state indicates how many pulses remain to be output before the end of a movement. The result is always positive.

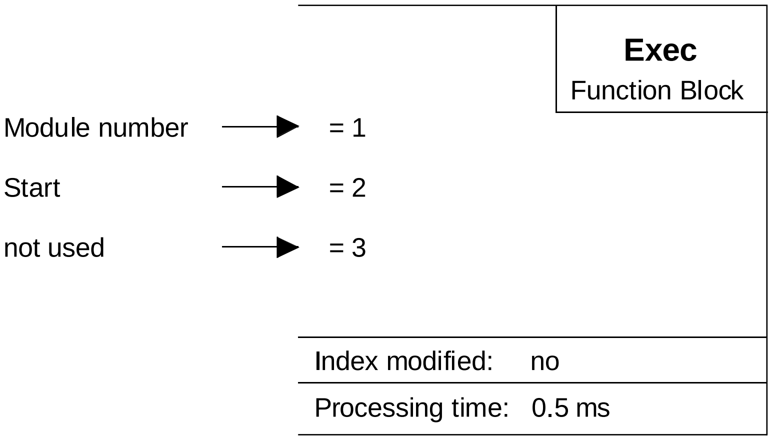
This command also calculates the actual absolute position, which is refreshed in the 'PosAbs_x' register.

Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: RdPosition				
= 3	PCD register with number of remaining steps	R	integer	24 bit	0 .. 16777215
PosAbs_x	Absolute position	R	integer	32 bit	-2^{31} .. $+2^{31}$ -1

Start

Command : Start motion

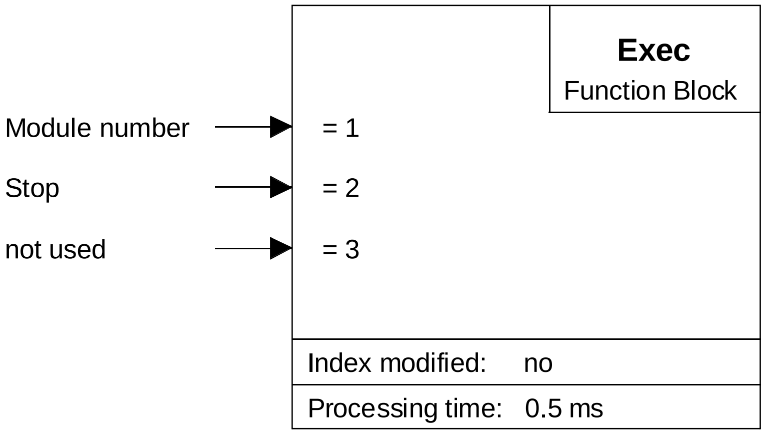


Function description :

This command transmits motion parameters 'Vmin', 'Vmax', 'Acc' and 'Dest' to the working register and starts the movement.

Stop

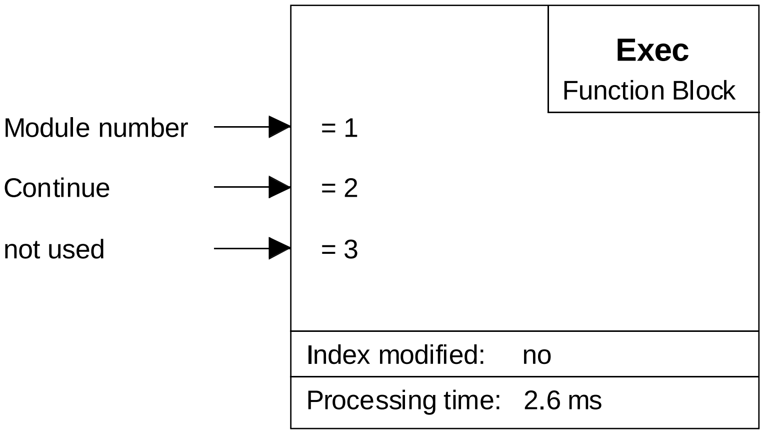
Command : Stop motion



Function description :

This command can be used to stop current motion, applying the normal braking ramp. As soon as the braking ramp has finished and the motor is at a standstill, input 'OnDest_x' is set high. An interrupted movement can be concluded with the ‘Continue’ command. The module is not blocked after a Stop command and, with a new Start command, it can still travel to a new position.

Continue **Command :** Cause stopped motion to continue

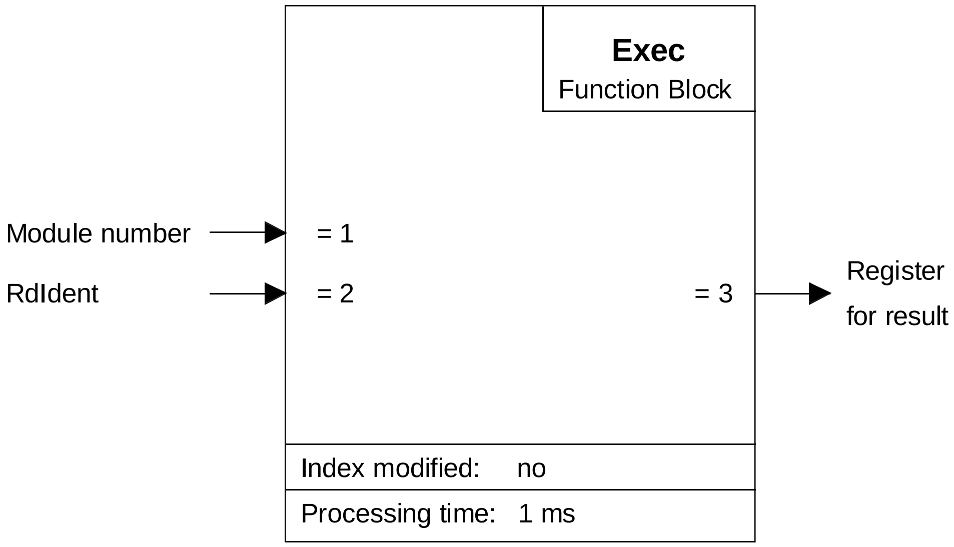


Function description :

This command causes motion which has been interrupted with 'Stop' to run on. Current position is read first, then the remaining path is loaded and motion started.

The 'Continue' command must only be sent when the motor has come to a standstill (Ondest_x = H), as otherwise the destination will not be approached correctly.

RdIdent **Command :** Read module identification



Function description :

This command can be used to check the correct function of the PCD2.H210 module and read the FPGA version. If the module is working properly, the value 32xx will be returned. See table below. If the module is faulty (or incorrectly addressed) the value 0 is read.

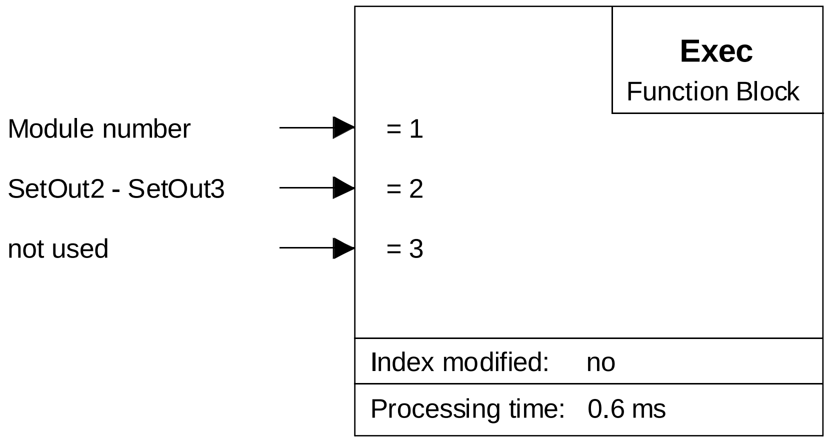
Description of participating input/output elements :

Par.	Designation/Function	Type	Format	Value	Comment
= 1	Module number	K		1 - 16	
= 2	Command: RdIdent				
= 3	Module identifier H210	R	integer	12 bit	0 → faulty

Table of valid identifiers :

Value	FPGA-Version
3200	Version HD0
3201	Version HD1
3202	Version HD2
...	...
3215	Version HDF

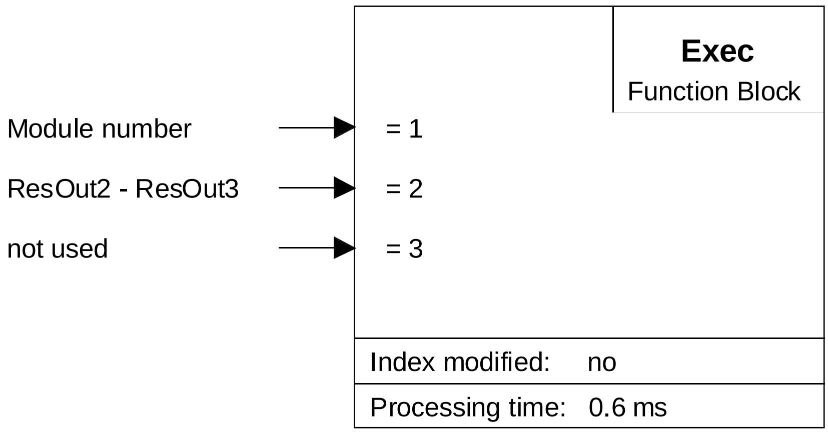
SetOut2 - SetOut3 **Command :** Set output 2 - Set output 3



Function description :

These commands set digital outputs A2 and A3 respectively. Since these outputs are not directly addressed by the PCD1/2 bus, they cannot be read back nor can they be affected by the 'Clear Outputs' or 'Clear All-Elements' commands of the debugger.

ResOut2 – ResOut3 **Command :** Reset output 2 - Reset output 3

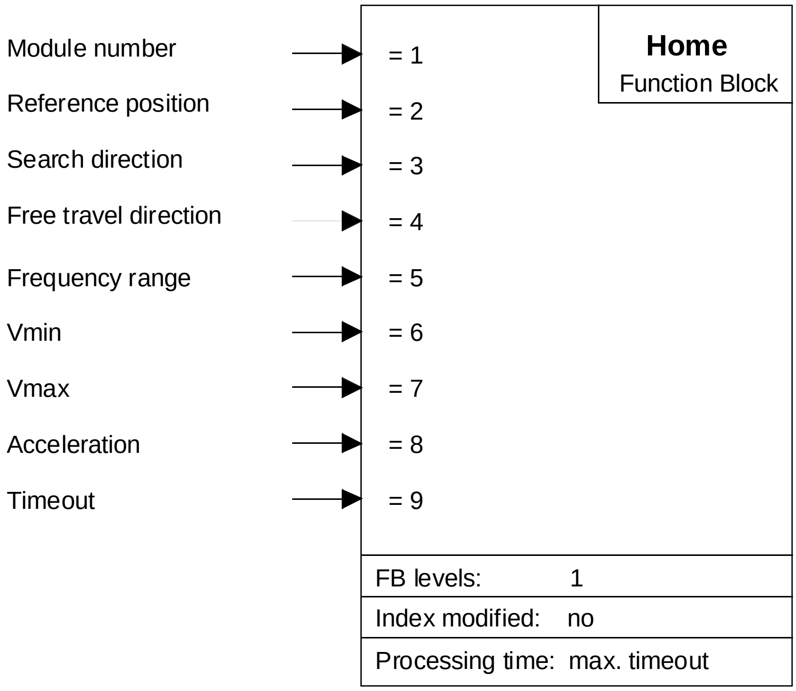


Function description :

These commands reset digital outputs A2 and A3 respectively. Since these outputs are not directly addressed by the PCD1/2 bus, they cannot be read back nor can they be affected by the 'Clear Outputs' or 'Clear All-Elements' commands of the debugger.

The function block 'Home'

Home **FB :** Find reference position



Function description :

This FB defines homing and starts it. The FB is only exited when the reference switch has been found or when the timeout has been reached. The module must be initialized (FB Init) before FB Home can be executed. The flag 'OnDest_x' indicates the end of the homing procedure. See section 5.4 for additional explanation of homing.

Par.	Designation	Type	Format	Value	Comment
= 1	Module number	K	K n	K1 – K 16	
= 2	Home position *)	R	integer	32 Bit	Absolute position
= 3	Search direction		integer	0 / 1	0 = neg. / 1 = pos.
= 4	Free travel direction		integer	0 / 1	0 = neg. / 1 = pos.
= 5	Frequency range		integer	0 – 3	See FB Init
= 6	Vmin for Home		integer	1 – Vmax	Entry checked
= 7	Vmax for Home		integer	Vmin – 255	Entry checked
= 8	Acc. for Home		integer	255 – 1	Entry checked
= 9	Timeout (max. time in seconds before stop occurs)		integer	0 – 65535 [s]	0 = no restriction

*) This parameter allows the definition of an offset for the reference position. (Refer to page 7-6 for the definition of any point as the zero position)

Notes :

Appendix B. Summary of all software elements for programming in FUPLA

In preparation

Notes :

From :

Company :
Department :
Name :
Address :

Tel. :

Date :

To send back to :

SAIA-Burgess Electronics Ltd.
Bahnhofstrasse 18
CH-3280 Murten (Switzerland)
http://www.saia-burgess.com

BA : Electronic Controllers

Manual PCD2.H210

If you have any suggestions concerning the SAIA® PCD, or have found any errors in this manual, brief details would be appreciated.

Your suggestions :