SAIA-Burgess Electronic

SWITCHES • MOTORS • CONTROLLER

Reference Guide

SAIA® PCD Series xx7

The SIMATIC® S7 compatible programmable logic controller

SAIA

English Edition 26/758 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

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

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

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

Issue : 01.02.2000

SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA® PCD

Process Control Devices

Series xx7

Compatible to SIMATIC S7-300/400

Reference Guide

English Edition 26/758 E2 - 02.2000

Update

Manual : **Reference Guide PCD Series xx7 - Edition E2**

Date	Chapter	Page	Description

Contents

Address Identifier and Parameter Ranges

Addr. ID	Parameter Range PCD1. M137	PCD2. M127	PCD2. M157/177 M257	Description
Q	0.0 to 255.7	0.0 to 255.7	0.0 to 255.7	Output bit (in PIQ)
QB	0 to 255	0 to 255	0 to 255	Output byte (in PIQ)
QW	0 to 254	0 to 254	0 to 254	Output word (in PIQ)
QD	0 to 252	0 to 252	0 to 252	Output double word (in PIQ)
DBX	0.0 to 65535.7	0.0 to 65535.7	0.0 to 65535.7	Data bit in data block
DB	1 to 1023	1 to 1023	1 to 1023	Data block
DBB	0 to 65535	0 to 65535	0 to 65535	Data byte in DB
DBW	0 to 65534	0 to 65534	0 to 65534	Data word in DB
DBD	0 to 65532	0 to 65532	0 to 65532	Data double word in DB
DIX	0.0 to 65533.7	0.0 to 65533.7	0.0 to 65533.7	Data bit in instance DB
DI	1 to 1023	1 to 1023	1 to 1023	Instance data block
DIB	0 to 65535	0 to 65535	0 to 65535	Data byte in instance DB
DIW	0 to 65534	0 to 65534	0 to 65534	Data word in instance DB
DID	0 to 65532	0 to 65532	0 to 65532	Data double word in instance DB

Addr. ID	Parameter Range PCD1. M137	PCD2. M127	PCD2. M157/177 M257	Description
I	0.0 to 255.7	0.0 to 255.7	0.0 to 255.7	Input bit (in PII)
IB	0 to 255	0 to 255	0 to 255	Input byte (in PII)
IW	0 to 254	0 to 254	0 to 254	Input word (in PII)
ID	0 to 252	0 to 252	0 to 252	Input double word (in PII)
L	0.0 to 255.7	0.0 to 511.7	0.0 to 511.7	Local data bit
LB	0 to 255	0 to 511	0 to 511	Local data byte
LW	0 to 254	0 to 510	0 to 510	Local data word
LD	0 to 252	0 to 508	0 to 508	Local data double word
M	0.0 to 2047.7	0.0 to 2047.7	0.0 to 2047.7	Bit memory bit
MB	0 to 2047	0 to 2047	0 to 2047	Bit memory byte
MW	0 to 2046	0 to 2046	0 to 2046	Bit memory word
MD	0 to 2044	0 to 2044	0 to 2044	Bit memory double word

*) Also restricted by the size of the working memory

Address Identifier and Parameter Ranges (continued)

Addr. ID	Parameter Range PCD1. M137	PCD2. M127	PCD2. M157/177 M257	Description
PQB	0 to 65535	0 to 65535	0 to 65535	Peripheral output byte (direct I/O access)
PQW	0 to 65534	0 to 65534	0 to 65534	Peripheral output word (direct I/O access)
PQD	0 to 65532	0 to 65532	0 to 65532	Peripheral output double word (direct I/O access)
PIB	0 to 65535	0 to 65535	0 to 65535	Peripheral input byte (direct I/O access)
PIW	0 to 65534	0 to 65534	0 to 65534	Peripheral input word (direct I/O access)
PID	0 to 65532	0 to 65532	0 to 65532	Peripheral input douple word (direct I/O access)
T	0 to 255	0 to 255	0 to 255	Timer
C	0 to 255	0 to 255	0 to 255	Counter

Constant	Range	Description
B (b1, b2) B (b1, b2, b3, b4)		Constant, 2 or 4 bytes
D# Date		IEC date constant
L# Integer		32-bit integer constant
P# Bit pointer		Pointer constant
S5T# Time		S7 time constant *)
T# Time		Time constant
TOD# Time		IEC time constant
C# Count		Counter constant (BCD code)
2#n		Binary constant
W#16# DW#16#		Hexadecimal constant

*) for loading of S7 timers

Notes :

Abbreviations and Mnemonics

Abbrev.	Description	Example
k8	8-bit constant 0 to 255	32
k16	16-bit constant 0 to 65 535	62 531, 0010 0111 0010 1100
k 32	32-bit constant 0 to 4 294 967 295	127 624
i8	8-bit integer -128 to +127	-113
i16	16-bit integer -32768 to +32767	+6523
i32	32-bit integer - 2 147 483 648 to +2 147 483 647	-2 222 222
m	Pointer constant	P#240.3
n	Binary constant	1001 1100
p	Hexadecimal constant	EA12
LABEL	Symbolic jump address (max. 4 characters)	DESTINATION

Abbrev.	Description	Example
a	Byte address	
b	Bit address	
c	Address area	I, Q, M, L, DBX, DIX
d	Address in : MD, DBD, DID or LD	
e	Number in : MW, DBW, DIW or LW	
f	Timer/counter No.	
g	Address area	IB, QB, PIB, MB, LB, DBB, DIB
h	Address area	IW, QW, PIW, MW, LW, DBW, DIW
i	Address area	ID, QD, PID, MD, LD, DBD, DID
q	Block No.	

The above abbreviations and mnemonics are used in the Instruction List.

Registers

ACCU1 to ACCU4 (32 Bits)

The accumulators are registers for processing bytes, words or double words. The address identifiers are loaded into the accumulators, where they are logically gated. The result of the logic operation (RLO) is in ACCU 1 and can be transferred from there to a memory cell.

The accumulators are 32 bits long.

Accumulator designation :

ACCU	Bits
ACCUx (x = 1 to 4)	Bits 0 to 31
ACCUx-L	Bits 0 to 15
ACCUx-H	Bits 16 to 31
ACCUx-LL	Bits 0 to 7
ACCUx-LH	Bits 8 to 15
ACCUx-HL	Bits 16 to 23
ACCUx-HH	Bits 24 to 31

Status Word (16 Bits)

The status word bits are evaluated or set by the instructions.

The status word is 16 bits long.

Bit	Assignment	Description
0	/FC	First check bit
1	RLO	Result of (previous) logic operation
2	STA	Status
3	OR	Or (AND before OR)
4	OS	Stored overflow
5	OV	Overflow
6	CC 0	Condition code 0
7	CC 1	Condition code 1
8	BR	Binary result
9 to 15	Unassigned	---

Registers (continued)

Address Register AR1 and AR2 (32 Bits)

The address registers contain the area-internal or area-crossing pointers for instructions using indirect addressing. The address registers are 32 bits long.

The area-internal and/or area-crossing pointers have the following syntax :

- Area-internal pointer :

  00000000 00000bbb bbbbbbbb bbbbbxxx

- Area-crossing pointer :

  yyyyyyyy 00000bbb bbbbbbbb bbbbbxxx

Legend :	b	Byte address
	x	Bit number
	y	Area identifier (see chapter "Examples of Addressing")

Examples of Addressing

Addressing Examples	Description
Direct Addressing	
L +27	Load 16-bit integer constant "27" into ACCU1
L L#-1	Load 32-bit integer constant "-1" into ACCU1
L 2#1010101010101010	Load binary constant into ACCU1
L DW#16#A0F0BCFD	Load hexadecimal constant into ACCU1
L 'ENDE'	Load ASCII character into ACCU1
L T#500 ms	Load time value into ACCU1
L C#100	Load count value into ACCU1
L B#(100,12)	Load 2-byte constant
L B#(100,12,50,8)	Load 4-byte constant
L P#10.0	Load area-internal pointer into ACCU1
L P#E20.6	Load area-crossing pointer into ACCU1
L -2.5	Load real into ACCU1
L D#1997-03-24	Load date
L TOD 13:20:33.125	Load time of day

Addressing Examples	Description
Direct Addressing	
A I 0.0	ANDing of input bit 0.0
L IB 1	Load input byte 1 into ACCU1
L IW 0	Load input word 0 into ACCU1
L ID 0	Load input double word 0 in ACCU1
Indirect Addressing of Timers/Counters	
SP T [LW 8]	Start timer; the timer number is in local word 8
CU C [LW 10]	Count upwards; the counter number is in local data word 10
Area-Internal Memory-Indirect Addressing	
A I [LD 12] | L P#22.2 Example : T LD 12 A I [LD 12]	AND operation : The address of the input is in local data double word 12 as pointer
A I [DBD 1]	AND operation : The address of the input is in data double word 1 of the open DB as pointer
A I [DID 1]	AND operation : The address of the output is in data double word 12 of the open instance DB as pointer
A I [MD 12]	AND operation : The address of the output is in memory double word 12 as pointer

Examples of Addressing (continued)

Addressing Examples	Description
Area-Internal Register-Indirect Addressing	
A I [AR1,P#12.2]	AND operation : The address of the input is calculated from the "pointer value in AR1 + P#12.2"
Area-Crossing Register-Indirect Addressing	
For area-crossing register-indirect addressing, the address must also contain an area identifier. The address is in the address register. The area identifiers are as follows :	

Area identifier	Coding binary	Coding hex.	Area
P	1000 0**000**	80	I/O area
I	1000 0**001**	81	Input area
Q	1000 0**010**	82	Output area
M	1000 0**011**	83	Bit memory area
DB	1000 0**100**	84	Data area
DI	1000 0**101**	85	Instance data area
L	1000 0**110**	86	Local data area
VL	1000 0**111**	87	Predecessor local data area (access to local data of invoking block)

Addressing Examples	Description
Area-Internal Register-Indirect Addressing (continued)	
L B [AR1,P#8.0]	Load byte into ACCU1 : The address is calculated from the "pointer value in AR1 + P#8.0"
U [AR1,P#32.3]	AND operation : The address of the operand is calculated from the "pointer value in AR1 + P#32.3"
Addressing Via Parameters	
A Parameter	The operand is addressed via parameters

Examples of how to calculate the pointer

- Example for sum of bit addresses ≤ 7 :
  LAR1 P#8.2
  A I [AR1,P#10.2]
  Result : Input 18.4 is addressed
  (by adding the byte and the bit addresses)

- Example for sum of bit addresses > 7 :
  L P#10.5
  LAR1
  A I [AR1,P#10.7]
  Result : Input 21.4 is addressed
  (by adding the byte and bit addresses with carry)

Bit Logic Instructions

A/AN

Instr.	Address ID	Description
A/AN		AND/AND NOT
	I/Q a.b	Input/output
	M a.b	Bit memory
	L a.b	Local data bit
	DBX a.b	Data bit
	DIX a.b	Instance data bit
	c [d]	Memory-indirect, area-internal
	c [AR1,m]	Register-indirect, area-internal (AR1)
	c [AR2,m]	Register-indirect, area-internal (AR2)
	[AR1,m]	Area-crossing (AR1)
	[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	Yes	-	Yes	Yes
Instruction affects :	-	-	-	-	-	Yes	Yes	Yes	1

All logic instructions generate a result (new RLO). The first instruction in a logic string generates the new RLO from the signal state scanned. The subsequent logic instructions generate the new RLO from the signal state scanned and the old RLO. The logic string ends with an instruction which limits the RLO (e.g. a memory instruction).

Bit Logic Instructions (continued)

O/ON

Instr.	Address ID	Description
O/ON		OR/OR NOT
	I/Q a.b	Input/output
	M a.b	Bit memory
	L a.b	Local data bit
	DBX a.b	Data bit
	DIX a.b	Instance data bit
	c [d]	Memory-indirect, area-internal
	c [AR1,m]	Register-indirect, area-internal (AR1)
	c [AR2,m]	Register-indirect, area-internal (AR2)
	[AR1,m]	Area-crossing (AR1)
	[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	Yes
Instruction affects :	-	-	-	-	-	0	Yes	Yes	1

Bit Logic Instructions (continued)

X/XN

Instr.	Address ID	Description
X/XN		Exclusive OR/
		Exclusive OR NOT
	I/Q a.b	Input/output
	M a.b	Bit memory
	L a.b	Local data bit
	DBX a.b	Data bit
	DIX a.b	Instance data bit
	c [d]	Memory-indirect, area-internal
	c [AR1,m]	Register-indirect, area-internal (AR1)
	c [AR2,m]	Register-indirect, area-internal (AR2)
	[AR1,m]	Area-crossing (AR1)
	[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	Yes
Instruction affects :	-	-	-	-	-	0	Yes	Yes	1

Bit Logic Instructions with Parenthetical Expressions

A(, AN(, O(, ON(, X(, XN(,

Instr.	Address ID	Description
A(		AND left parenthesis
AN(		AND NOT left parenthesis
O(		OR left parenthesis
ON(		OR NOT left parenthesis
X(		Exclusive OR left parenthesis
XN(		Exclusice OR NOT left parenthesis

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	Yes	-	Yes	Yes
Instruction affects :	-	-	-	-	-	0	1	-	0

Saving the RLO and OR bits and the relevant function identifier (A, AN, ...) to the nesting stack. Seven nesting levels are possible per block. After the right parenthesis, the logic operation indicated by the function identifier is performed on the saved RLO and the current RLO, the current OR is overwritten with the saved OR.

Bit Logic Instructions with Parenthetical Expressions (continued)

)

Instr.	Address ID	Description
)		Right parenthesis, removing an entry from the nesting stack.

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	Yes	1	Yes	1

Saving the RLO and OR bits and the relevant function identifier (A, AN, ...) to the nesting stack. Seven nesting levels are possible per block. After the right parenthesis, the logic operation indicated by the function identifier is performed on the saved RLO and the current RLO, the current OR is overwritten with the saved OR.

ORing of AND Instructions

O

Instr.	Address ID	Description
O		ORing of AND operations according to the rule : AND before OR.

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	Yes
Instruction affects :	-	-	-	-	-	Yes	1	-	Yes

The ORing of AND instructions is implemented according to the rule : AND before OR.

Logic Instruction with Timers and Counters

A/AN

Instr.	Address ID		Description
A/AN			AND/AND NOT
	T	f	Timer
	T	[e]	Timer, memory-indirect addressing
	C	f	Counter
	C	[e]	Counter, memory-indirect addressing
	Timer parameter		Timer/counter
	Counter parameter		(addressing via parameter)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	Yes	-	Yes	Yes
Instruction affects :	-	-	-	-	-	Yes	Yes	Yes	1

Examining the status of the addressed timer/counter and gating the result with the RLO according to the appropriate logic function.

Logic Instructions with Timers and Counters (continued)

O/ON

Instr.	Address ID		Description
O/ON			OR/OR NOT
	T	f	Timer
	T	[e]	Timer, memory-indirect addressing
	C	f	Counter
	C	[e]	Counter, memory-indirect addressing
	Timer parameter		Timer/counter
	Counter parameter		(addressing via parameter)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	Yes
Instruction affects :	-	-	-	-	-	0	Yes	Yes	1

Examining the status of the addressed timer/counter and gating the result with the RLO according to the appropriate logic function.

Logic Instructions with Timers and Counters (continued)

X/XN

Instr.	Address ID		Description
X/XN			EXCLUSIVE OR/
			EXCLUSIVE OR NOT
	T	f	Timer
	T	[e]	Timer, memory-indirect addressing
	C	f	Counter
	C	[e]	Counter, memory-indirect addressing
	Timer parameter		EXCLUSIVE OR timer/counter
	Counter parameter		(addressing via parameter)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	Yes
Instruction affects :	-	-	-	-	-	0	Yes	Yes	1

Examining the status of the addressed timer/counter and gating the result with the RLO according to the appropriate logic function.

Word Logic Instructions with the Contents of Accumulator 1

AW, OW, XOW, AD, OD, XOD

Instr.	Address ID	Description
AW		AND ACCU2-L
AW	W#16#p	AND 16-bit constant
OW		OR ACCU2-L
OW	W#16#p	OR 16-bit constant
XOW		EXCLUSIVE OR ACCU2-L
XOW	W#16#p	EXCLUSIVE OR 16-bit constant
AD		AND ACCU2
AD	DW#16#p	AND 32-bit constant
OD		OR ACCU2
OD	DW#16#p	OR 32-bit constant
XOD		EXCLUSIVE OR ACCU2
XOD	DW#16#p	EXCLUSIVE OR 32-bit constant

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	0	0	-	-	-	-	-

Gating the contents of ACCU1 and/or ACCU1-L with a word or double word according to the appropriate function. The word or double word is either specified in the instruction as an address or is in ACCU2. The result is in ACCU1 and/or ACCU1-L.

Evaluating Conditions Using AND, OR and EXCLUSIVE OR

A/AN, O/ON, X/XN

Instr.	Address ID	Description
A/AN		AND/AND NOT
O/ON		OR/OR NOT
X/XN		EXCLUSIVE OR/EXCLUSIVE OR NOT
	== 0	Result = 0 (CC1 = 0 and CC0 = 0)
	> 0	Result > 0 (CC1 = 1 and CC0 = 0)
	< 0	Result < 0 (CC1 = 0 and CC0 = 1)
	<> 0	Result ≠ 0 ((CC1 = 0 and CC0 = 1) or (CC1 = 1 and CC0 = 0))
	<= 0	Result <= 0 ((CC1 = 0 and CC0 = 1) or (CC1 = 0 and CC0 = 0))
	>= 0	Result >= 0 ((CC1 = 1 and CC0 = 0) or (CC1 = 0 and CC0 = 0))

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	Yes	Yes	-	-	Yes	-	Yes	Yes
Instruction affects :	-	-	-	-	-	Yes	Yes	Yes	1

All logic instructions generate a result (new RLO). The first instruction in a logic string generates the new RLO from the signal state scanned. The subsequent logic instructions generate the new RLO from the signal state scanned and the old RLO. The logic string ends with an instruction which limits the RLO (e.g. a memory instruction).

Evaluating Conditions Using AND, OR and EXCLUSIVE OR (continued)

A/AN, O/ON, X/XN

Instr.	Address ID	Description
A/AN		AND/AND NOT
O/ON		OR/OR NOT
X/XN		EXCLUSIVE OR/EXCLUSIVE OR NOT
	UO	Unordered math instruction (CC1 = 1 and CC0 = 1)
	OS	AND OS = 1
	BR	AND BR = 1
	OV	AND OV = 1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes	Yes
Instruction affects :	-	-	-	-	-	Yes	Yes	Yes	1

Edge-Triggered Instructions

FP/FN

Instr.	Address ID	Description
FP/FN	I/Q a.b M a.b L a.b *) DBX a.b DIX a.b c [d] c [AR1,m] c [AR2,m] [AR1,m] [AR2,m] Parameter	The positive/negative edge is indicated by RLO = 1. The bit addressed in the instruction is the auxiliary edge bit memory.

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	Yes	Yes	1

The current RLO is compared with the status of the instruction or "edge bit memory". FP detects a change from "0" to "l"; FN detects a change from "l" to "0".

*) Unnecessary if the bit being monitored is in the process image (local data of a block are only valid while the block is running).

Setting/Resetting Bit Addresses

S, R,

Instr.	Address ID	Description
S		Set addressed bit to "1"
R		Set addressed bit to "0"
	I/Q a.b	Input/output
	M a.b	Bit memory
	L a.b	Local data bit
	DBX a.b	Data bit
	DIX a.b	Instance data bit
	c [d]	Memory-indirect, area-internal
	c [AR1,m]	Register-indirect, area-internal (AR1)
	c [AR2,m]	Register-indirect, area-internal (AR2)
	[AR1,m]	Area-crossing (AR1)
	[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	Yes	-	0

Assigning the value "l" or "0" to the addressed instruction when RLO = 1. The instructions can be dependent on the MCR.

Setting/Resetting Bit Addresses (continued)

=

Instr.	Address ID	Description
=		Assign RLO
	I/Q a.b	To input/output
	M a.b	To bit memory
	L a.b	To local data bit
	DBX a.b	To data bit
	DIX a.b	To instance data bit
	c [d]	Memory-indirect, area-internal
	c [AR1,m]	Register-indirect, area-internal (AR1)
	c [AR2,m]	Register-indirect, area-internal (AR2)
	[AR1,m]	Area-crossing (AR1)
	[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	Yes	-	0

The RLO is written to the address of the instruction. The instructions can be dependent on the MCR.

Instructions Directly Affecting the RLO

CLR, SET,

Instr.	Address ID	description
CLR		Set RLO to "0"

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	0	0	0	0

Instr.	Address ID	Description
SET		Set RLO to "1"

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	0	1	1	0

The above instructions have a direct effect on the RLO.

Instructions Directly Affecting the RLO (continued)

NOT, SAVE

Instr.	Address ID	Description
NOT		Negate RLO

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	Yes	-	Yes	-
Instruction affects :	-	-	-	-	-	-	1	Yes	-

Instr.	Address ID	Description
SAVE		Save RLO to the BR bit

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	Yes	-	-	-	-	-	-	-	-

The above instructions have a direct effect on the RLO.

Timer Instructions

SP, SE, SD, SS,

Instr.	Address ID	Description
SP	T f T [e]	Start timer as pulse on edge change from "0" to "1".
	Timer para.	
SE	T f T [e]	Start timer as extended pulse on edge change from "0" to "1".
	Timer para.	
SD	T f T [e]	Start timer as ON delay on edge change from "0" to "1".
	Timer para.	
SS	T f T [e]	Start timer as retentive ON delay on edge change from "0" to "1".
	Timer para.	

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	-	-	0

Starting or resetting a timer. The time value must be in ACCU1-L. The instructions are triggered by an edge transition in the RLO; that is, when the status of the RLO has changed between two calls.

Timer Instructions (continued)

SF, FR, R

Instr.	Address ID	Description
SF	T f T [e] Timer para.	Start timer as OFF delay on edge change from "1" to "0".
FR	T f T [e] Timer para.	Enable timer for restarting on edge change from "0" to "1". (reset edge bit memory for starting timer).
R	T f T [e] Timer para.	Reset timer

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	-	-	0

Starting or resetting a timer. The time value must be in ACCU1-L. The instructions are triggered by an edge transition in the RLO; that is, when the status of the RLO has changed between two calls.

Counter Instructions

S, R, CU, CD, FR

Instr.	Address ID	Description
S	C f C [e] Counter para.	Presetting of counter on edge change from "0" to "1".
R	C f C [e] Counter para.	Reset counter to "0" when RLO = "1"
CU	C f C [e] Counter para.	Increment counter by 1 on edge change from "0" to "1".
CD	C f C [e] Counter para.	Decrement counter by 1 on edge change from "0" to "1".
FR	C f C [e] Counter para.	Enable counter on edge change from "0" to "1". (reset edge bit memory for up and down counting and setting the counter)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	-	-	0

The count value must be in ACCU1-L in the form of a BCD number (0 - 999).

Load Instructions

L

Instr.	Address ID	Description
L		Load ...
	IB a	Input byte
	QB a	Output byte
	PIB a	Peripheral input byte
	MB a	Bit memory byte
	LB a	Local data byte
	DBB a	Data byte
	DIB a	Instance data byte ... into ACCU1
	g [d]	Memory-indirect, area-internal
	g [AR1,m]	Register-indirect, area-internal (AR1)
	g [AR2,m]	Register-indirect, area-internal (AR2)
	B[AR1,m]	Area-crossing (AR1)
	B[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Loading address identifiers into ACCU1. The contents of ACCU1 are first saved to ACCU2.
The status word is not affected.

Load Instructions (continued)

L

Instr.	Address ID	Description
L		Load ...
	IW a	Input word
	QW a	Output word
	PIW a	Peripheral input word
	MW a	Bit memory word
	LW a	Local data word
	DBW a	Data word
	DIW a	Instance data word ... into ACCU1-L
	h [d]	Memory-indirect, area-internal
	h [AR1,m]	Register-indirect, area-internal (AR1)
	h [AR2,m]	Register-indirect, area-internal (AR2)
	W[AR1,m]	Area-crossing (AR1)
	W[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Load Instructions (continued)

L

Instr.	Address ID	Description
L		Load ...
	ID a	Input double word
	QD a	Output double word
	PID a	Peripheral input double word
	MD a	Bit memory double word
	LD a	Local data double word
	DBD a	Data double word
	DID a	Instance data double word ... in ACCU1
	i [d]	Memory-indirect, area-internal
	i [AR1,m]	Register-indirect, area-internal (AR1)
	i [AR2,m]	Register-indirect, area-internal (AR2)
	D[AR1,m]	Area-crossing (AR1)
	D[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Load Instructions (continued)

L

Instr.	Address ID	Description
L		Load ...
	k8	8-bit constant into ACCU1-LL
	k16	16-bit constant into ACCU1-L
	k32	32-bit constant into ACCU1
	Parameter	Load constant into ACCU1 (addressed via parameter)
L	2#n	Load 16-bit binary constant into ACCU1-L
		Load 32-bit binary constant into ACCU1
	B#16#p	Load 8-bit hexadecimal constant into ACCU1-L
L	W#16#p	Load 16-bit hexadecimal constant into ACCU1-L
	DW#16#p	Load 32-bit hexadecimal constant into ACCU1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Load Instructions (continued)

L

Instr.	Address ID	Description
L	'x'	Load 1 character
	'xx'	Load 2 characters
	'xxx'	Load 3 characters
	'xxxx'	Load 4 characters
L	D# time value	Load IEC date
L	S5T# time value	Load S7 time constant (16 bits)
L	TOD# time value	Load IEC time constant
L	T# time value	Load 16-bit time constant
		Load 32-bit time constant
L	C# count value	Load counter constant (BCD code)
L	B# (b1, b2)	Load constant as byte (b1, b2)
	B# (b1, b2, b3, b4)	Load constant as 4 bytes (b1, b2, b3, b4)
L	P# bit pointer	Load bit pointer
L	L# integer	Load 32-bit integer constant
L	Real number	Load floating-point number

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Load Instructions for Timers and Counters

L, LC

Instr.	Address ID	Description
L	T f T (e)	Load time value
	Timer para.	Load time value (addressed via parameter)
L	C f C (e)	Load count value
	Counter para.	Load count value (addressed via parameter)
LC	T f T (e)	Load time value in BCD
	Timer para.	Load time value in BCD (addressed via parameter)
LC	C f C (e)	Load count value in BCD
	Counter para.	Load count value in BCD (addressed via parameter)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Loading a time value or count value into ACCU1. The contents of ACCU1 are first saved to ACCU2. The bits of the status word are not affected.

Transfer Instructions

T

Instr.	Address ID	Description
T		Transfer contents of ACCU1-LL to ...
	IB a	Input byte
	QB a	Output byte
	PQB a	Peripheral output byte
	MB a	Bit memory byte
	LB a	Local data byte
	DBB a	Data byte
	DIB a	Instance data byte
	g [d]	Memory-indirect, area-internal
	g [AR1,m]	Register-indirect, area-internal (AR1)
	g [AR2,m]	Register-indirect, area-internal (AR2)
	B[AR1,m]	Area-crossing (AR1)
	B[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Transferring the contents of ACCU1 to the addressed operand. Note that some instructions are affected by the MCR. The status word is not affected.

Transfer Instructions (continued)

T

Instr.	Address ID	Description
T		Transfer contents of ACCU1-L to ...
	IW a	Input word
	QW a	Output word
	PQW a	Peripheral output word
	MW a	Bit memory word
	LW a	Local data word
	DBW a	Data word
	DIW a	Instance data word
	h [d]	Memory-indirect, area-internal
	h [AR1,m]	Register-indirect, area-internal (AR1)
	h [AR2,m]	Register-indirect, area-internal (AR2)
	W[AR1,m]	Area-crossing (AR1)
	W[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Transfer Instructions (continued)

T

Instr.	Address ID	Description
T		Transfer contents of ACCU1 to ...
	ID a	Input double word
	QD a	Output double word
	PQD a	Peripheral output double word
	MD a	Bit memory double word
	LD a	Local data double word
	DBD a	Data double word
	DID a	Instance data double word
	i [d]	Memory-indirect, area-internal
	i [AR1,m]	Register-indirect, area-internal (AR1)
	i [AR2,m]	Register-indirect, area-internal (AR2)
	D[AR1,m]	Area-crossing (AR1)
	D[AR2,m]	Area-crossing (AR2)
	Parameter	Via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Load and Transfer Instructions for Address Registers

LAR1, LAR2

Instr.	Address ID	description
LAR1		Load contents from ...
	-	ACCU1
	AR2	Address register 2
	DBD a	Data double word
	DID a	Instance data double word
	m	32-bit constant as pointer
	LD a	Local data double word
	MD a	Bit memory double word
		... into AR1
LAR2		Load contents from ...
	-	ACCU1
	DBD a	Data double word
	DID a	Instance data double word
	m	32-bit constant as pointer
	LD a	Local data double word
	MD a	Bit memory double word
		... into AR2

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Loading a double word from a memory area or register into address register 1 (AR1) or address register 2 (AR2). The status word is not affected.

Load and Transfer Instructions for Address Registers (continued)

TAR1, TAR2, CAR

Instr.	Address ID		Description
TAR1			Transfer contents from AR1 in ...
	-		ACCU1
	AR2		Address register 2
	DBD	a	Data double word
	DID	a	Instance data double word
	m		32-bit constant as pointer
	LD	a	Local data double word
	MD	a	Bit memory double word
TAR2			Transfer contents from AR2 in ...
	-		ACCU1
	DBD	a	Data double word
	DID	a	Instance data double word
	m		32-bit constant as pointer
	LD	a	Local data double word
	MD	a	Bit memory double word
CAR			Exchange the contents of AR1 and AR2

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Transferring a double word from address register 1 (AR1) or address register 2 (AR2) to a memory area or register. The contents of ACCU1 are first saved to ACCU2. The status word is not affected.

Load and Transfer Instructions for the Status Word

L STW, T STW

Instr.	Address ID	description
L	STW	Load status word into ACCU1

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Instruction affects :	-	-	-	-	-	-	-	-	-

T	STW	Transfer ACCU1 (bits 0 to 8) to the status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Load Instructions for DB Number and DB Length

L

Instr.	Address ID	Description
L	DBNO	Load number of data block
L	DINO	Load number of instance data block
L	DBLG	Load length of data block into byte
L	DILG	Load length of instance data block into byte

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Loading the number/length of a data block into ACCU1. The old contents of ACCU1 are saved to ACCU2. The status word is not affected.

Notes :

Integer Math (16 Bits)

+I, -I, ∗I, /I

Instr.	Address ID	Description
+I		Add 2 integers (16 bits) (ACCU1-L) = (ACCU2-L) + (ACCU1-L)
-I		Subtract 1 integer from another (16 bits) (ACCU1-L) = (ACCU2-L) - (ACCU1-L)
∗I		Multiply 1 integer by another (16 bits) (ACCU1) = (ACCU2-L) ∗ (ACCU1-L)
/I		Divide 1 integer by another (16 bits) (ACCU1-L) = (ACCU2-L) : (ACCU1-L) The remainder is in ACCU1-H

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

Math instructions on two 16-bit words. The result is written to ACCU1 and/or ACCU1-L.
ACCU3 and ACCU4 are then transferred to ACCU2 and ACCU3.

Integer Math (32 Bits)

+D, -D, ∗D, /D, MOD

Instr.	Address ID	Description
+D		Add 2 integers (32 bits) (ACCU1) = (ACCU2) + (ACCU1)
-D		Subtract 1 integer from another (32 bits) (ACCU1) = (ACCU2) - (ACCU1)
∗D		Multiply 1 integer by another (32 bits) (ACCU1) = (ACCU2) ∗ (ACCU1)
/D		Divide 1 integer by another (32 bits) (ACCU1) = (ACCU2) : (ACCU1)
MOD		Divide 1 integer by another (32 bits) and load the remainder into ACCU1 : (ACCU1) = remainder of [(ACCU2) : (ACCU1)]

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

Math instructions on two 32-bit words. The result is written to ACCU1. ACCU3 and ACCU4 are then transferred to ACCU2 and ACCU3.

Floating-Point Math (32 Bits)

+R, -R, ∗R, /R

Instr.	Address ID	Description
+R		Add 2 real numbers (32 bits) (ACCU1) = (ACCU2) + (ACCU1)
-R		Subtract 1 real number from another (32 bits) (ACCU1) = (ACCU2) - (ACCU1)
∗R		Multiply 1 real number by another (32 bits) (ACCU1) = (ACCU2) ∗ (ACCU1)
/R		Divide 1 real number by another (32 bits) (ACCU1) = (ACCU2) : (ACCU1)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

The result of the math instruction is in ACCU1.

Floating-Point Math (32 Bits) (continued)

NEGR, ABS

Instr.	Address ID	Description
NEGR		Negate the real number in ACCU1
ABS		Form the absolute value of the real number in ACCU1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

The result of the math instruction is in ACCU1.

Square Root and Square Instructions (32 Bits)

SQRT, SQR

Instr.	Address ID	Description
SQRT		Calculate the square root of a real number in ACCU1
SQR		Form the square of the real number in ACCU1

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

The result of the instruction is in ACCU1. The SQRT instruction can be interrupted.

Logarithmic Function (32 Bits)

LN, EXP

Instr.	Address ID	Description
LN		Form the natural logarithm of a real number in ACCU1
EXP		Calculate the exponential value of a real number in ACCU1 to the base e (= 2,71828)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

The result of the logarithmic function is in ACCU1. The instructions can be interrupted.

Trigonometrical Functions (32 Bits)

SIN, ASIN, COS, ACOS, TAN, ATAN

Instr.	Address ID	Description
SIN		Calculate the sine of a real number
ASIN		Calculate the arcsine of a real number
COS		Calculate the cosine of a real number
ACOS		Calculate the arccosine of a real number
TAN		Calculate the tangent of a real number
ATAN		Calculate the arctangent of a real number

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

The result of the instruction is in ACCU1. The instructions can be interrupted.

Adding Constants

+

Instr.	Address ID	Description
+	i8	Add an 8-bit integer constant
+	i16	Add a 16-bit integer constant
+	i32	Add a 32-bit integer constant

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Adding integer constants and storing the result in ACCU1. The status word is not affected.

Adding Using Address Registers

+AR1, +AR2

Instr.	Address ID	Description
+AR1		Add the contents of ACCU1-L to those of AR1
+AR1	m (0 to 4095)	Add a pointer constant to the contents of AR1
+AR2		Add the contents of ACCU1-L to those of AR2
+AR2	m (0 to 4095)	Add a pointer constant to the contents of AR2

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Adding a 16-bit integer to the contents of the address register. The value is either specified as an address in the instruction or is in ACCU1-L. The status word is not affected.

Comparison Instructions (16-Bit Integers)

==I, <>I, <I, <=I, >I, >=I

Instr.	Address ID	Description
==I		ACCU2-L = ACCU1-L
<>I		ACCU2-L ≠ ACCU1-L
<I		ACCU2-L < ACCU1-L
<=I		ACCU2-L <= ACCU1-L
>I		ACCU2-L > ACCU1-L
>=I		ACCU2-L >= ACCU1-L

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	0	-	0	Yes	Yes	1

Comparing the 16-bit integers in ACCU1-L and ACCU2-L. RLO = 1 if the condition is satisfied.

Comparison Instructions (32-Bit Integers)

==D, <>D, <D, <=D, >D, >=D

Instr.	Address ID	Description
==D		ACCU2 = ACCU1
<>D		ACCU2 ≠ ACCU1
<D		ACCU2 < ACCU1
<=D		ACCU2 <= ACCU1
>D		ACCU2 > ACCU1
>=D		ACCU2 >= ACCU1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	0	-	0	Yes	Yes	1

Comparing the 32-bit integers in ACCU1 and ACCU2. RLO = 1 if the condition is satisfied.

Comparison Instructions (32-Bit Real Numbers)

==R, <>R, <R, <=R, >R, >=R

Instr.	Address ID	Description
==R		ACCU2 = ACCU1
<>R		ACCU2 ≠ ACCU1
<R		ACCU2 < ACCU1
<=R		ACCU2 <= ACCU1
>R		ACCU2 > ACCU1
>=R		ACCU2 >= ACCU1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	0	Yes	Yes	1

Comparing the 32-bit real numbers in ACCU1 and ACCU2. RLO = 1 if the condition is satisfied.

Shift Instructions

SLW, SLD, SRW, SRD

Instr.	Address ID	Description
SLW *)		Shift the contents of ACCU1-L to the left. Positions that become free are provided with zeros.
SLW	0 ... 15	
SLD		Shift the contents of ACCU1 to the left. Positions that become free are provided with zeros.
SLD	0 ... 32	
SRW *)		Shift the contents of ACCU1-L to the right. Positions that become free are provided with zeros.
SRW	0 ... 15	
SRD		Shift the contents of ACCU1 to the right. Positions that become free are provided with zeros.
SRD	0 ... 32	

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	0	0	-	-	-	-	-

*) No. of places shifted : 0 to 16

Shifting the contents of ACCU1 and ACCU1-L to the left or right by the specified number of places.
If no address identifier is specified, the contents of ACCU2-LL are used as the number of places.
The last bit shifted is loaded into condition code bit CC1.

Shift Instructions (continued)

SSI, SSD

Instr.	Address ID	Description
SSI *)		Shift the contents of ACCU1-L with sign to the right. Positions that become free are provided with the sign (bit 15).
SSI	0 ... 15	
SSD		Shift the contents of ACCU1 with sign to the right. Positions that become free are provided with the sign (bit 31).
SSD	0 ... 32	

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	0	0	-	-	-	-	-

*) No. of places shifted : 0 to 16

Shifting the contents of ACCU1 and ACCU1-L to the left or right by the specified number of places.
If no address identifier is specified, the contents of ACCU2-LL are used as the number of places.
The last bit shifted is loaded into condition code bit CC1.

Rotate Instructions

RLD, RRD, RLDA, RRDA

Instr.	Address ID	Description
RLD		Rotate the contents of ACCU1 to the left
RLD	0 ... 32	
RRD		Rotate the contents of ACCU1 to the right
RRD	0 ... 32	

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	-	-	-	-	-

Instr.	Address ID	Description
RLDA		Rotate the contents of ACCU1 one bit position to the left through condition code bit CC1
RRDA		Rotate the contents of ACCU1 one bit position to the right through condition code bit CC1

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	0	0	-	-	-	-	-

Rotate the contents of ACCU1 to the left or right by the specified number of places. If no address identifier is specified, the contents of ACCU2-LL are used as the number of places. The last bit shifted is loaded into condition code bit CC1.

Notes :

Accumulator Transfer Instructions, Incrementing and Decrementing

CAW, CAD, TAK, ENT, LEAVE, PUSH, POP, INC, DEC

Instr.	Address ID	Description
CAW		Reverse the order of the bytes in ACCU1-L
CAD		Reverse the order of the bytes in ACCU1
TAK		Swap the contents of ACCU1 and ACCU2
ENT		The contents of ACCU2 and ACCU3 are transferred to ACCU3 and ACCU4
LEAVE		The contents of ACCU3 and ACCU4 are transferred to ACCU2 and ACCU3
PUSH		The contents of ACCU1, ACCU2 and ACCU3 are transferred to ACCU2, ACCU3 and ACCU4
POP		The contents of ACCU2, ACCU3 and ACCU4 are transferred to ACCU1, ACCU2 and ACCU3
INC	k8	Increment ACCU1-LL
DEC	k8	Decrement ACCU1-LL

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

The status word is not affected.

Program Display and Null Operation Instructions

BLD, NOP

Instr.	Address ID	Description
BLD	k8	Program display instruction : Is treated by the CPU as a null operation instruction
NOP	0 1	Null operation instruction :

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

The status word is not affected.

Data Type Conversion Instructions

BTI, BTD, DTR, ITD

Instr.	Address ID	Description
BTI		Convert contents of ACCU1-L from BCD (0 to +/- 999) to integer (16 bits) (**B**CD **T**o **I**nt)
BTD		Convert contents of ACCU1 from BCD (0 to +/- 9 999 999) to double integer (32 bits) (**B**CD **T**o **D**oubleint)
DTR		Convert contents of ACCU1 from double integer (32 bits) to real number (32 bits) (**D**oubleint **T**o **R**eal)
ITD		Convert contents of ACCU1 from integer (16 bits) to double integer (32 bits) (**I**nt **T**o **D**oubleint)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

The results of the conversion are in ACCU1.

Data Type Conversion Instructions (continued)

ITB, DTB

Instr.	Address ID	Description
ITB		Convert contents of ACCU1-L from integer (16 bits) to BCD (0 to +/- 999) (**I**nt **T**o **B**CD)
DTB		Convert contents of ACCU1 from double integer (32 bits) to BCD (0 to +/- 9 999 999) (**D**oubleint **T**o **B**CD)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	Yes	Yes	-	-	-	-

The results of the conversion are in ACCU1.

Data Type Conversion Instructions (continued)

RND, RND-, RND+, TRUNC

Instr.	Address ID	Description
RND		Convert a real number into a 32-bit integer.
RND-		Convert a real number into a 32-bit integer. The number is rounded down to the next whole number.
RND+		Convert a real number into a 32-bit integer. The number is rounded up to the next whole number.
TRUNC		Convert a real number into a 32-bit integer. The places after the decimal point are truncated.

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	Yes	Yes	-	-	-	-

The real number to be converted is in ACCU1.

Forming the Ones and Twos Complements

INVI, INVD, NEGI, NEGD

Instr.	Address ID	Description
INVI		Form the ones complement of ACCU1-L
INVD		Form the ones complement of ACCU1.

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

NEGI		Form the twos complement of ACCU1-L (integer)
NEGD		Form the twos complement of ACCU1 (double integer)

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	Yes	Yes	Yes	Yes	-	-	-	-

Block Call Instructions

CALL, UC, CC

Instr.	Address ID	Description
CALL	FB q, DB q	Unconditional call of an FB, with parameter transfer.
CALL	SFB q, DB q	Unconditional call of an SFB, with parameter transfer.
CALL	FC q	Unconditional call of a function, with parameter transfer.
CALL	SFC q	Unconditional call of an SFC, with parameter transfer.
UC	FB q FC q SFC q FB [e] FC [e] Parameter	Unconditional call of blocks, without parameter transfer. Memory-indirect FB call Memory-indirect FC call FB/FC call via parameter
CC	FB q FC q FB [e] FC [e] Parameter	Conditional call of blocks, without parameter transfer. Memory-indirect FB call Memory-indirect FC call FB/FC call via parameter

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	0	0	1	-	0

The information on the status word only relates to the block call itself and not to the commands called in this block.

Status word for CC : depending of RLO, set RLO to 1

Block Call Instructions (continued)

OPN

Instr.	Address ID	Description
OPN		Open :
	DB q	Data block
	DI q	Instance data block
	DB [e]	Data block, memory-indirect
	DI [e]	Instance data block, memory-indirect
	Parameter	Data block using parameters

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Block End Instructions

BE, BEU, BEC

Instr.	Address ID	Description
BE		End block
BEU		End block unconditionally

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	0	0	1	-	0

BEC		End block conditionally if RLO = "1"

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	Yes	0	1	1	0

Exchanging Shared Data Block and Instance Data Block

CDB

Instr.	Address ID	Description
CDB		Exchange shared data block and instance data block

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Exchanging the two current data blocks. The current shared data block becomes the current instance data block, and vice versa. The status word is not affected.

Jump Instructions

JU, JC, JCN, JCB, JNB

Instr.	Address ID	Description
JU	LABEL	Jump unconditionally

Status word									
Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

JC	LABEL	Jump if RLO = "1"
JCN	LABEL	Jump if RLO = "0"

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	1	1	0

JCB	LABEL	Jump if RLO = "1" Save the RLO in the BR bit
JNB	LABEL	Jump if RLO = "0" Save the RLO in the BR bit

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	Yes	-	-	-	-	0	1	1	0

Jumping as a function of conditions.

Jump Instructions (continued)

JBI, JNBI, JO, JOS

Instr.	Address ID	Description
JBI	LABEL	Jump if BR = "1"
JNBI	LABEL	Jump if BR = "0"

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	Yes	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	0	1	-	0

JO	LABEL	Jump on stored overflow (OV = "1")

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	Yes	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

JOS	LABEL	Jump on stored overflow (OS = "1")

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	Yes	-	-	-	-
Instruction affects :	-	-	-	-	0	-	-	-	-

Jump Instructions (continued)

JUO, JZ, JP, JM, JN, JMZ, JPZ

Instr.	Address ID	Description
JUO	LABEL	Jump if "unordered math instruction" (CC1 = 1 and CC0 = 1)
JZ	LABEL	Jump if result = 0 (CC1 = 0 and CC0 = 0)
JP	LABEL	Jump if result > 0 (CC1 = 1 and CC0 = 0)
JM	LABEL	Jump if result < 0 (CC1 = 0 and CC0 = 1)
JN	LABEL	Jump if result ≠ 0 (CC1 = 1 and CC0 = 0) or (CC1 = 0 and CC0 = 1)
JMZ	LABEL	Jump if result ≤ 0 (CC1 = 0 and CC0 = 1) or (CC1 = 0 and CC0 = 0)
JPZ	LABEL	Jump if result ≥ 0 (CC1 = 1 and CC0 = 0) or (CC1 = 0 and CC0 = 0)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	Yes	Yes	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Jump Instructions (continued)

JL, LOOP

Instr.	Address ID	Description
JL	LABEL	Jump distributor This instruction is followed by a list of jump instructions. The address identifier is a jump label to subsequent instructions in this list. ACCU1-LL contains the number of the jump instruction to be executed (max. 254). The number of the first jump instruction is 0.
LOOP	LABEL	Decrement ACCU1-L and jump if ACCU1-L ≠ 0 (loop programming)

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

Instructions for the Master Control Relay (MCR)

MCR(,)MCR, MCRA, MCRD

Instr.	Address ID	Description
MCR(		Open an MCR zone. Save the RLO to the MCR stack.

Status word

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	Yes	-
Instruction affects :	-	-	-	-	-	0	1	-	0

)MCR		Close an MCR zone. Pop an entry off the MCR stack.

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	0	1	-	0

MCRA		Activate the MCR
MCRD		Deactivate the MCR

Status bit	BR	CC1	CC0	OV	OS	OR	STA	RLO	/FC
Instruction evaluates :	-	-	-	-	-	-	-	-	-
Instruction affects :	-	-	-	-	-	-	-	-	-

MCR = 1 → MCR is deactivated
MCR = 0 → MCR is activated; "T" and "=" instructions write zeros to the corresponding address identifiers if RLO = "0"; "S" and "R" instructions leave the memory contents unchanged.

Notes :

Blocks and Functions of the CPUs

Organization Blocks	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257	Start Events (Hexadecimal Values)
Free cycle :				
OB 1	x	x	x	1101, 1102, 1103
Time-of-day interrupts :				
OB 10	x	x	x	1111
OB 11	x	x	x	1112
OB 12		x	x	1113
OB 13		x	x	1114
OB 14			x	1115
OB 15			x	1116
OB 16			x	1117
OB 17			x	1118
Time-delay interrupts :				
OB 20	x	x	x	1121
OB 21		x	x	1122
OB 22		x	x	1123
OB 23		x	x	1124

A user program for the PCD1/2.Mxx7 is made up of blocks containing the statements, parameters and data for the relevant CPU. The number of blocks you can create or which are provided by the operating system is different for each of the PCD1/2.Mxx7. You will find a detailed description of the OBs and their use in the *STEP 7 Programming Manual*.

Blocks and Functions of the CPUs (continued)

Organization Blocks	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257	Start Events (Hexadecimal Values)
Timed interrupts :				
OB 30			x	1131
OB 31			x	1132
OB 32		x	x	1133
OB 33		x	x	1134
OB 34		x	x	1135
OB 35	x	x	x	1136
OB 36			x	1137
OB 37			x	1138
OB 38			x	1139
Hardware interrupts :				
OB 40	x	x	x	1141, 1142, 1143, 1144
OB 41	x	x	x	1141, 1142, 1143, 1144
OB 42		x	x	1141, 1142, 1143, 1144
OB 43		x	x	1141, 1142, 1143, 1144
OB 44			x	1141, 1142, 1143, 1144
OB 45			x	1141, 1142, 1143, 1144
OB 46			x	1141, 1142, 1143, 1144
OB 47			x	1141, 1142, 1143, 1144

Blocks and Functions of the CPUs (continued)

Organization Blocks	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257	Start Events (Hexadecimal Values)
Asynchronous error interrupts :				
OB 80	x	x	x	3501, 3502, 3505, 3506, 3507
OB 81	x	x	x	3821, 3822, 3823, 3831, 3832, 3833 3921, 3922, 3923, 3931, 3932, 3933
OB 82	x	x	x	3842, 3942
OB 83	x	x	x	3861, 3863, 3864, 3961
OB 84	x	x	x	3881, 3981
OB 85	x	x	x	35A1, 35A3, 39B1, 39B2
OB 86	x	x	x	38C1, 38C2, 39C1
OB 87	x	x	x	35D2, 35D3, 35D4, 35D5, 35E1, 35E2, 35E3, 35E4, 35E5, 35E6
Complete restart :				
OB 100	x	x	x	1381, 1382
Synchronous error interrupts :				
OB 121	x	x	x	2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 253A, 253C, 253D,253E, 253F
OB 122	x	x	x	2942, 2943, 2944, 2945

Function Blocks, Functions and Data Blocks

Functions Blocks	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257
Quantity	512	512	512
Permissible numbers	0 to 511	0 to 511	0 to 511
Maximum size of a function block (code required for execution)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)

Data Blocks	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257
Quantity	1023	1023	1023
Permissible numbers	1 to 1023	1 to 1023	1 to 1023
Maximum size of a data block (number of data bytes)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)

Functions	PCD1.M137	PCD2.M127	PCD2.M157 / M177 / M257
Quantity	1024	1024	1024
Permissible numbers	0 to 1023	0 to 1023	0 to 1023
Maximum size of a function (code required for execution)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)	64 Kbytes - 2 bytes *)

*) Block size restricted by size of working memory

The above tables list the quantities, numbers and maximum sizes of the function blocks, functions and data blocks you can create for the various CPUs of the SAIA PCD Series xx7.

System Functions

SFC No.	SFC Name	Function
0	SET_CLK	Set clock
1	READ_CLK	Read clock
2	SET_RTM	Set run-time meter
3	CTRL_RTM	Start and stop run-time meter
4	READ_RTM	Read run-time meter
6	RD_SINFO	Read start information of current OB
13	DP_NRMDG	Read slave diagnostic data First call Intermediate call, REQ = 0 Last call (6 - 240 bytes)
14	DPRD_DAT	Read consistent user data (8 bytes)
15	DPWR_DAT	Write consistent user data (8 bytes)

SFC No.	SFC Name	Function
20	BLKMOV	Copy variable (n = number of bytes to be copied)
21	FILL	Set array default variables (n = length of target variables in bytes)
22	CREAT_DB	Create data block
23	DEL_DB	Delete data block
24	TEST_DB	Test data block
25	COMPRESS	Compress user memory First call (trigger) Intermediate call (active) Last call (finished)
26	UPDAT_PI	Update process image input table (run-time entry for 1 DI 32 in the central rack)
27	UPDAT_PO	Update process image output table (run-time entry for 1 DO 32 in the central rack)

System Functions (continued)

SFC No.	SFC Name	Function
28	SET_TINT	Set time-of-day interrupt
29	CAN_TINT	Cancel time-of-day interrupt
30	ACT_TINT	Activate time-of-day interrupt
31	QRY_TINT	Query time-of-day interrupt
32	SRT_DINT	Start time-delay interrupt
33	CAN_DINT	Cancel time-delay interrupt
34	QRY_DINT	Query time-delay interrupt
36	MSK_FLT	Mask synchronous faults
37	DMSK_FLT	Demask synchronous faults
38	READ_ERR	Read error register
39	DIS_IRT	Discard new events Block all events (MODE = 0) Block all events of a priority class (MODE = 1) Block one event (MODE = 2)
40	EN_IRT	Stop discarding events Enable all events (MODE = 0) Enable all events in a priority class (MODE = 1) Enable an event (MODE = 2)

SFC No.	SFC Name	Function
41	DIS_AIRT	Delay interrupt events the first time delay is activated if the delay is already activated.
42	EN_AIRT	Stop delaying interrupt events when canceling the last delay if other delays are present
43	RE_TRIGR	Retrigger watchdog monitoring
44	REPL_VAL	Transfer substitute value to ACCU1
46	STP	Force CPU into STOP mode
52	WR_USMSG	Write user entry in diagnostic buffer
60	GD_SND	Send GD package
61	GD_RCV	Receive GD package
64	TIME-TCK	Display millisecond timer

System Functions for SAIA® PCD - Series xx7

SFC No.	SFC Name	Function
220	LON_INIT	Initialize the LON-interface
221	NV_SEND	Send NV on the LON network
223	MSG_SEND	Send message on the LON network
227	PCD104_RD	Read Data from the Dual-Port-Ram
228	PCD104_WR	Write Data on the Dual-Port-Ram
229	PCD104_ST	Status of the Dual-Port-Ram
230	RD_COMP	Compilation status of a block
239	WDOG	Refresh time out of the Watch Dog

SFC No.	SFC Name	Function
240	COM_RCV	Read received data on a serial port
241	COM_SEND	Send data on a serial port
242	COM_STAT	Status of a serial port
243	COM_INIT	Initialize a serial port without protocol
244	COM_SIG	Access to the modem signals
245	B_INIT	Initialize a serial port with protocol (with protocol for SFB 12-14)
248	_INTDIR_	Bidirectional faster counter
250	INP_INT	Enable/disable interrupt inputs
251	INITCNTR	Configure and start integrated counter
252	READCNTR	Status of the counter
253	READ_SSI	Read data of the SSI interface
254	GRAY2BIN	Convert Gray-Code to Binary-Code

System Function Blocks

SFB No.	SFB Name	Function
12	BSEND	Send data block by block JOB activated (1 - 440 bytes) JOB activated (> 440 bytes) JOB checked JOB finished (DONE = 1)
13	BRCV	Receive data block by block JOB activated JOB checked JOB finished

SFB No.	SFB Name	Function
14	GET	Read data from remote CPU (one area specified) JOB activated JOB checked JOB finished (NDR = 1; 1 - 450 bytes)

System Function Blocks for SAIA® PCD - Series xx7

SFB No.	SFB Name	Function
240	FLASH	Read - Write DB's on Flash-EEprom

SFB No.	SFB Name	Function

Alphabetical Index of Instructions

Alphabetical Index of Instructions (continued)

Notes :

From :

Company :
Department :
Name :
Address :

Tel. :

Date :

To

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

BA: Electronic Controllers

Reference Guide PCD Series xx7

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

Your suggestions :