



**SESAM 4**

RSC - Kassen  
Protokoll

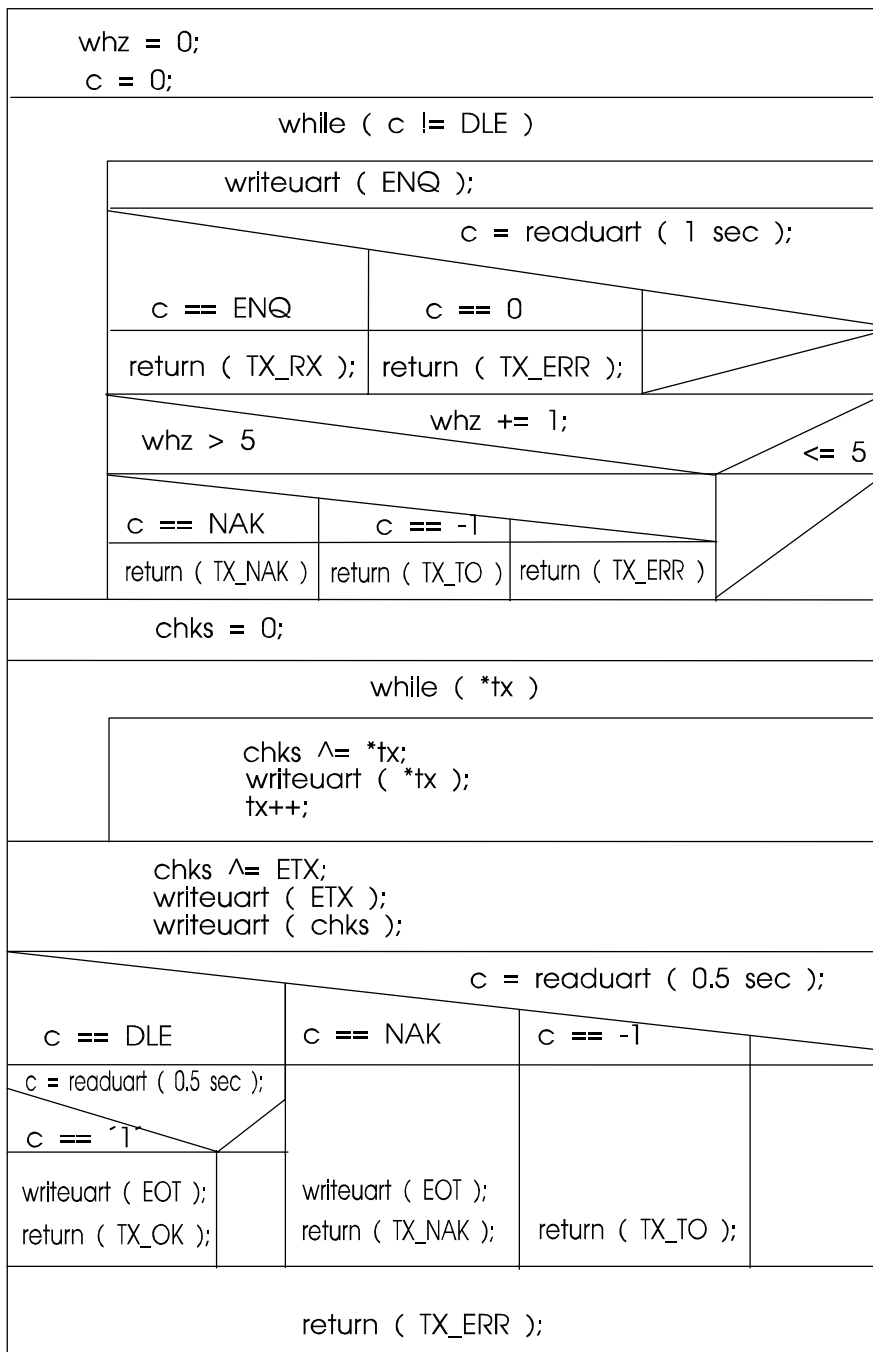
<b>Contents</b>	<b>Page</b>
<b>RSC - PROTOCOL</b>	<b>3</b>
<b>LSV2 Protocol</b>	<b>3</b>
LSV2 Send: int tx_lsv2(char *tx)	4
LSV2 Receive: int rx_lsv2(char *tx)	5
<b>Messages:</b>	<b>6</b>
Request table: Schank <=> Kassa/PC	6
Tablemode:	6
close table: Schank <=> Kassa/PC	6
Bonrequest: Kassa/PC => Schank (used only in polling mode)	7
Response:	7
Bon-send: Kassa/PC <=> Schank	7
Example:	9
<b>Request accounts:</b>	<b>11</b>
Request: total sections	12
Response:	12
Request: totals all waiter	13
Response:	13
Request: Waiter totals by section	14
Response:	14
<b>cancel totals:</b>	<b>15</b>
<b>Request: Credit-memory</b>	<b>16</b>
Response:	16

**RSC - Protocol****LSV2 Protocol**

Sender		Receiver	
ENQ	(0x05)	=>	
		<=	DLE '0'      Timeout 1 sec. (0x010 0x030)
STX	(0x02)	=>	Timeout 0,5 sec. / Character
Daten....		=>	
ETX	(0x03)	=>	
LRC		=>	LRC = EXOR (Data+ETX) excluding STX
		<=	Timeout 0,5 sec DLE '1'      (0x010 0x031)
EOT	(0x04)	=>	
Timeouts for 9600 Bd			

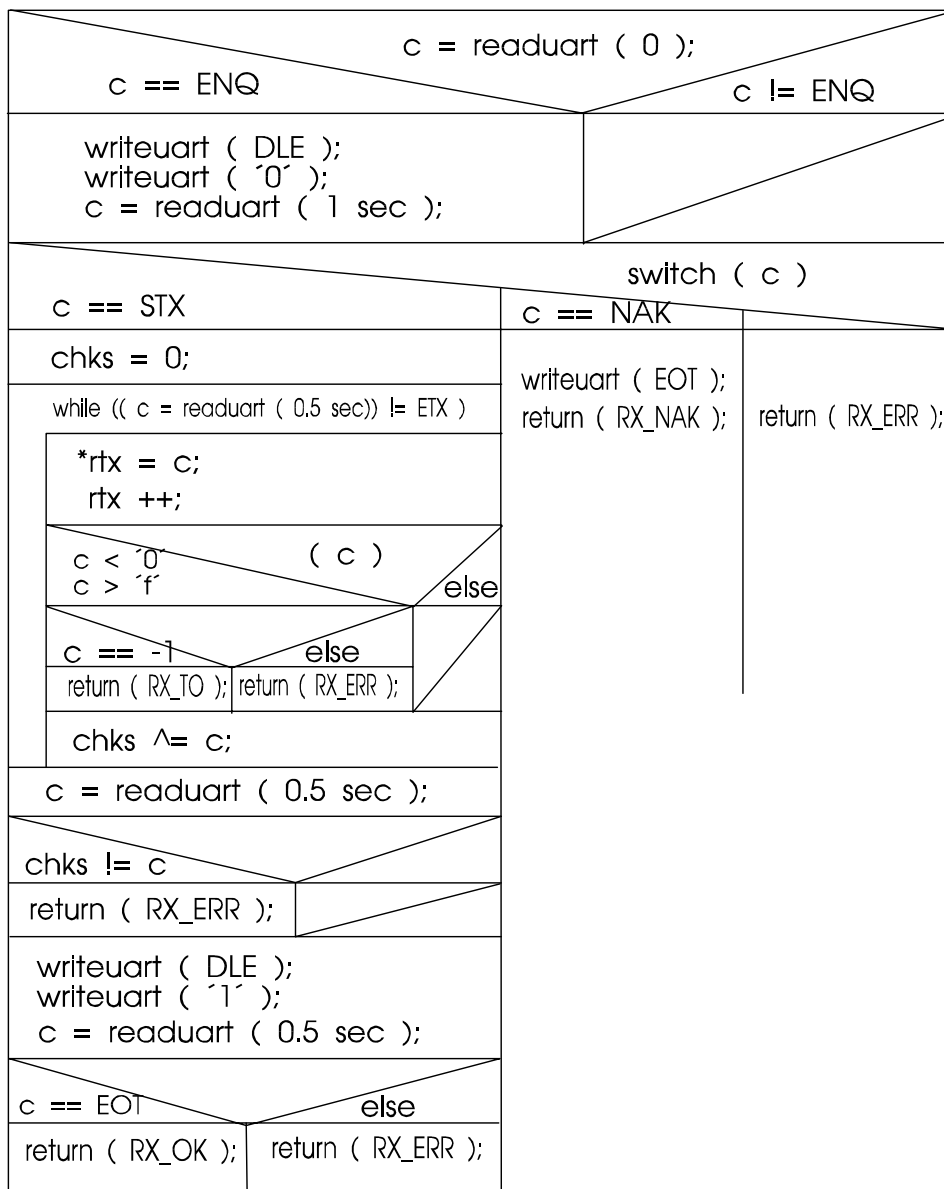
## LSV2 Send: int tx lsv2(char \*tx)

Returnvalue:   #define TX\_OK    0    /\* OK                               \*/  
                  #define TX\_TO   1    /\* Timeout                       \*/  
                  #define TX\_RX   2    /\* Data received               \*/  
                  #define TX\_NAK   3    /\* Data not accept           \*/  
                  #define TX\_ERR   4    /\* Transmission error       \*/



## LSV2 Receive: int rx\_lsv2(char \*tx)

Returnvalue:   #define RX\_OK    0     /\* OK                               \*/  
                   #define RX\_TO    1     /\* Timeout                        \*/  
                   #define TX\_NAK   3     /\* Data not accept           \*/  
                   #define TX\_ERR   4     /\* Transmission error       \*/



## Messages:

### Request table: Schank <=> Kassa/PC

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	1 3	IDENT = TIFRAGE
5 - 6	0 X	open mode
7 - 9	X X X	Waiter
10 - 14	X X X X X	Table

**open:**

TTEST 0 Test table  
TMAKE 1 Open table.  
TMAKE2 2 Open table and lock. (PC ==> Schank )  
temporary not accessable

### Response:

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	2 1	IDENT = TIANZWORT
5 - 6	X X	tablemode
7 - 9	X X X	Waiter who occupied table
10 - 14	X X X X X	table
15 - 22	X X X X X X,X X	table summ (in cents)

### Tablemode:

This response modes will be shown on the Display

```

TIA_OK      0    "Table Su:%8u.%02u"
TIA_TIDIS   1                    ( internal use only )
TIA_EXDIS   2    "Table ext. disable" ( internal use only )
TIA_NOTABLE 3    "Tablenumber ?"
TIA_LEER    4    "Table empty !"
TIA_BESETZT 5    "Table occupied by %3u"
TIA_NOMEM   6    "Table memory full"
TIA_NEU     7    "new table opened"
TIA_ZUKLEIN 8    "Table nr too small"
TIA_ZUGROSS 9    "Table nr too large"
TIA_ERROR   10   "Error TEST-GAST" ( could not send to PC )
TIA_TIMEOUT 11   "PC/Kassa TIMEOUT" ( PC didn't respond )
TIA_TINUM   12   "Tablenumber: %5u"
TIA_TIGESP  13   "Table blocked"
TIA_NERLAUB 14   "Table not valid"
TIA_BLOCK   15   "Table locked"

```

### close table: Schank <=> Kassa/PC

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 9	IDENT = STMPPAR
5 - 6	1 2	SID = STISCLR
7 - 8	0 X	CLRDIR
9 - 12	X X X X	4 Bytes not used
13 - 17	X X X X X	Tablenumber

**CLRDIR:**

0 delete content of table  
1 close and remove table  
2 lock table  
3 unlock table

**Bonrequest:** Kassa/PC => Schank (used only in polling mode)

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	1 8	IDENT = request

**Response:**

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	1 9	IDENT = no data

**Bon-send:** Kassa/PC <=> Schank

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 7	IDENT = AUFZPAR
5 - 12	X X X X X X X X	Flags
13 - 15	X X X	Waiter (1-127)
16 - 18	X X X	Section (1-20)
19 - 23	X X X X X	Table (0-65535)
24 - 28	X X X X X	Quantity (1-32767)
29 - 33	X X X X X	PLU-number (1-1000)
34 - 41	X X X X X X,X X	Sum Price*Quantity (in cents) Max: 16777215 (0x00ffffff)

**Flags:**

Flags are 32 bit values and will be send as ASCII-Hex.

i.e. ( printf („%08X“,flag);

Bit 0 - 2 Record Type

LEER_RC	0x00000000	
BEIL_RC	0x00000001	Change of attribute (i.e. change Rice to Chips)
SPA_RC	0x00000002	Add value to section
PLU_RC	0x00000003	order PLU
EXT_RC	0x00000004	External PLU (from Coffemachine, Spirit-dispenser...)
GET_RC	0x00000005	Drink PLU ( dispensed by the Schank)

Bit 3 - 4 Pricelevel

PR1_RC	0x00000000
PR2_RC	0x00000008
PR3_RC	0x00000010
PR4_RC	0x00000018

Bit 5 HEAD\_RC Header print receipt

Bit 6 END\_RC End print

Bit 7 NACH\_RC follower data (drink taken without waiterkey should not happen)

Bit 8 STOR\_RC Storno

Bit 9 STOE\_RC Storno Extra

Bit 10 not used

Bit 11 HAUS\_RC PLU-taken with housekey

Bit 12 DRUK\_RC print this record

Bit 13 not used

Bit 14 BEIN\_RC next record is change of attribute ( used by the printer )

Bit 15 SPAR\_RC add into section memory

Bit 16 CRED\_RC add into credit memory

Bit 17 GAST\_RC add into guestcheck ( table memory )

Bit 18 KASS\_RC sent to PC ( internal use only )

Bit 19 not used

Bit 20 LAG\_RC reduce stock

Bit 21 ERRO\_RC follower message because of net error

Bit 22 - 31 printer 1-10 (by bit you can print one message up to 10 printer at the same time)



## Example:

send to the PC: Schank --> Kassa

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 7	IDENT = AUFZPAR
5 - 12	0 0 D 4 9 0 6 D	Flags: GET_RC A drink is dispensed PR2_RC Pricelevel 2 HEAD_RC,END_RC single receipt DRUK_RC print enable SPAR_RC add into section and detailed memory. KASS_RC send to the PC LAG_RC reduce stock 0x00C00000 print at 1. u. 2. receipt printer
13 - 15	0 0 1	waiter 1
16 - 18	0 0 3	section 3
19 - 23	0 0 0 2 5	table 25
24 - 28	0 0 0 0 1	quantity 1
29 - 33	0 0 0 1 5	PLU-Number 15
34 - 41	0 0 0 0 2 6 0 0	Total price*quantity (in cents)

## add Credit : KASSA --> Schank ( standard )

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 7	IDENT = AUFZPAR
5 - 12	0 0 0 1 0 0 0 3	Flags: <b>PLU_RC</b> <b>CRED_RC</b> add into creditmemory
13 - 15	0 0 1	waiter 1
16 - 18	0 0 0	section 0 (not used)
19 - 23	0 0 0 0 0	table 0 (not used)
24 - 28	0 0 0 1 0	quantity 10
29 - 33	0 0 1 3 5	PLU-Number 135
34 - 41	0 0 0 0 0 0 0 0	Sum (not used)

## add Credit : KASSA --> Schank ( advanced )

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 7	IDENT = AUFZPAR
5 - 12	0 0 5 1 9 0 6 3	Flags: <b>PLU_RC</b> <b>HEAD_RC,END_RC</b> print single receipt <b>DRUK_RC</b> enable printing <b>SPAR_RC</b> add summ into section ( in this case the prices in the Schank must have valid values ) <b>CRED_RC</b> add into credit memory <b>LAG_RC</b> reduce stock <b>0x00400000</b> 1. receipt printer
13 - 15	0 0 1	waiter 1
16 - 18	0 0 3	section 3
19 - 23	0 0 0 2 5	table 25 (used by the printer)
24 - 28	0 0 0 1 0	quantity 10
29 - 33	0 0 1 3 5	PLU-Number 135
34 - 41	0 0 0 7 0 0 0 0	Summ Price*quantity (in cents)

**Request accounts:**

NODE = 00 always all computers

IDENT = PROGPARG ( 5 )

SID = PRTSPATA ( 42 ) sections day all waiters  
PRTSPAMA ( 43 ) sections month all waiters  
PRTAKTAB ( 44 ) totals for all waiters day  
PRTAKMAB ( 45 ) totals all waiters month  
PRTKETAB ( 46 ) waiter sections day  
PRTKEMAB ( 47 ) waiter sections month

PRTKETDA ( 48 ) waiter detail (old )  
PRTKEMDA ( 49 ) waiter details including storno  
PRTDETTA ( 50 ) total details day  
PRTDETMO ( 51 ) total details month

PRTAKTST ( 62 ) get waiters with values

INDEX = waiter (1-128)

## Request: total sections

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPARG
5 - 6	4 2 ( 4 3 )	SID = total day (month)

## Response:

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPARG
5 - 6	4 2 ( 4 3 )	SID = total day (month)
7 - 10	X X X X	INDEX = not used
11 - 18	X X X X X X,X X	Section 1 Price 1
19 - 26	X X X X X X,X X	Section 1 Price 2
27 - 34	X X X X X X,X X	Section 1 Price 3
35 - 42	X X X X X X,X X	Section 1 Price 4
43 - 50	X X X X X X,X X	Section 2 Price 1
51 - 58	X X X X X X,X X	Section 2 Price 2
59 - 66	X X X X X X,X X	Section 2 Price 3
67 - 74	X X X X X X,X X	Section 2 Price 4
75 -	.	.
- 618	.	.
619 - 626	X X X X X X,X X	Section 20 Price 1
627 - 634	X X X X X X,X X	Section 20 Price 2
635 - 642	X X X X X X,X X	Section 20 Price 3
643 - 650	X X X X X X,X X	Section 20 Price 4
651 - 658	X X X X X X,X X	Total Storno Price 1
659 - 666	X X X X X X,X X	Total Storno Price 2
667 - 674	X X X X X X,X X	Total Storno Price 3
675 - 682	X X X X X X,X X	Total Storno Price 4
683 - 690	X X X X X X,X X	Total Scheck Price 1 ( not used )
691 - 698	X X X X X X,X X	Total Scheck Price 2 ( not used )
699 - 706	X X X X X X,X X	Total Scheck Price 3 ( not used )
707 - 714	X X X X X X,X X	Total Scheck Price 4 ( not used )
715 - 722	X X X X X X,X X	Total Kreditk Price 1 ( not used )
723 - 730	X X X X X X,X X	Total Kreditk Price 2 ( not used )
731 - 738	X X X X X X,X X	Total Kreditk Price 3 ( not used )
739 - 746	X X X X X X,X X	Total Kreditk Price 4 ( not used )
747 - 754	X X X X X X,X X	Total Zubuch Price 1 ( not used )
755 - 762	X X X X X X,X X	Total Zubuch Price 2 ( not used )
763 - 770	X X X X X X,X X	Total Zubuch Price 3 ( not used )
771 - 778	X X X X X X,X X	Total Zubuch Price 4 ( not used )
779 - 786	X X X X X X,X X	Total Abbuch Price 1 ( not used )
787 - 794	X X X X X X,X X	Total Abbuch Price 2 ( not used )
795 - 802	X X X X X X,X X	Total Abbuch Price 3 ( not used )
803 - 818	X X X X X X,X X	Total Abbuch Price 4 ( not used )
819 - 822	X X X X	Year <b>Date last cancel totals</b>
823 - 824	X X	Month 1-12
825 - 826	X X	Day 1-31
827 - 828	X X	Hour 1-23
829 - 830	X X	Minute 1-59
831 - 834	X X X X	Year <b>Current Date</b>
835 - 836	X X	Month 1-12
837 - 838	X X	Day 1-31
839 - 840	X X	Hour 1-23
841 - 842	X X	Minute 1-59

**Request:** totals all waiter

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	4 4 ( 4 5 )	SID = All waiter day (month)

**Response:**

Only waiters with values will be sent.

**Attention:** for this message you must use longer timeouts

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	4 4 ( 4 5 )	SID = All waiter day (month)
7 - 10	X X X X	INDEX = not used
11 - 13	X X X	Waiter Number (1-128)
14 - 21	X X X X X X.X X	Waiter Total
22 - 29	X X X X X X.X X	Waiter Provision
30 - 32	X X X	Waiter Number (1-128)
33 - 40	X X X X X X.X X	Waiter Total
41 - 48	X X X X X X.X X	Waiter Provision
43 -		.
-		.
-		.

## Request: Waiter totals by section

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	4 6 ( 4 7 )	SID = Waiter Section Day (month)
7 - 10	X X X X	INDEX = Waiter

## Response:

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	4 6 ( 4 7 )	SID = Waiter by section Day (Month)
7 - 10	X X X X	INDEX = Waiter
11 - 18	X X X X X X,X X	Section 1
19 - 26	X X X X X X,X X	Section 2
27 -	.	.
-	.	.
-	.	.
163 - 170	X X X X X X,X X	Section 20
171 - 178	X X X X X X,X X	Total Storno
179 - 186	X X X X X X,X X	Total Scheck (not used )
187 - 194	X X X X X X,X X	Total Kreditk (not used )
195 - 202	X X X X X X,X X	Total Zubuch (not used )
203 - 210	X X X X X X,X X	Total Abbuch (not used )
211 - 218	X X X X X X,X X	Total Abbuch (not used )
219 - 226	X X X X X X,X X	Reserve
227 - 230	X X X X	Year <b>Date last cancel totals</b>
231 - 232	X X	Month 1-12
233 - 234	X X	Day 1-31
235 - 236	X X	Hour 1-23
237 - 238	X X	Minute 1-59
239 - 242	X X X X	Year <b>Current date</b>
243 - 244	X X	Month 1-12
245 - 246	X X	Day 1-31
247 - 248	X X	Hour 1-23
249 - 250	X X	Minute 1-59

## cancel totals:

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 6	IDENT = EXECPAR
5 - 6	4 2 ( 4 3 )	SID = <b>all Section Day (Month)</b>

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 6	IDENT = EXECPAR
5 - 6	4 4 ( 4 5 )	SID = <b>Alle Waiter Day (Month)</b>

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 6	IDENT = EXECPAR
5 - 6	4 6 ( 4 7 )	SID = <b>Waiter Section Day (Month)</b>
7 - 10	X X X X	INDEX = Waiter

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 6	IDENT = EXECPAR
5 - 6	5 5 ( 5 6 )	SID = <b>All Waiter+ all section Day (Month)</b>

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 6	IDENT = EXECPAR
5 - 6	5 7	SID = Div. extra memory

## Request: Credit-memory

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	6 4	SID = Creditmemory
7 - 10	X X X X	INDEX = Waiter

### NODE:

if **NODE = 0** you will get the data from global credit node.

in case of credit lokal (we can setup this in the schank) you must request every node

### INDEX (Waiter):

if Waiter=0 you get the complete credit memory

if waiter is any number between 1 and 127 you get the creditmemory from the corresponding waiter.

### Response:

Position	Value	Comment
1 - 2	0 0	NODE = Global
3 - 4	0 5	IDENT = PROGPAR
5 - 6	6 4	SID = Creditmemory
7 - 10	X X X X	INDEX = Waiter
11 - 13	X X X	Waiter
14 - 18	X X X X X	Quantity
19 - 23	X X X X X	Plu
24 - 26	X X X	Waiter
27 - 31	X X X X X	Quantity
32 - 36	X X X X X	Plu
37 -		.
-		.
-		.