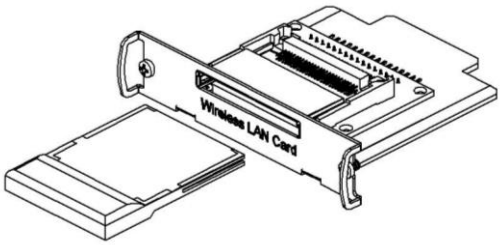
CL5000/CL5500 Series

Protocol's Manual

(English)

Rev. 2008. 09.25
CL5000 Latest F/W Version : V1.62.0
CL5500 Latest F/W Version : V2.91.0



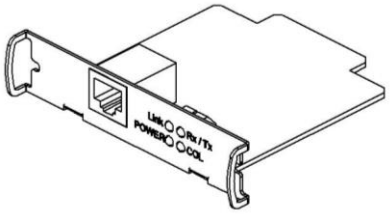
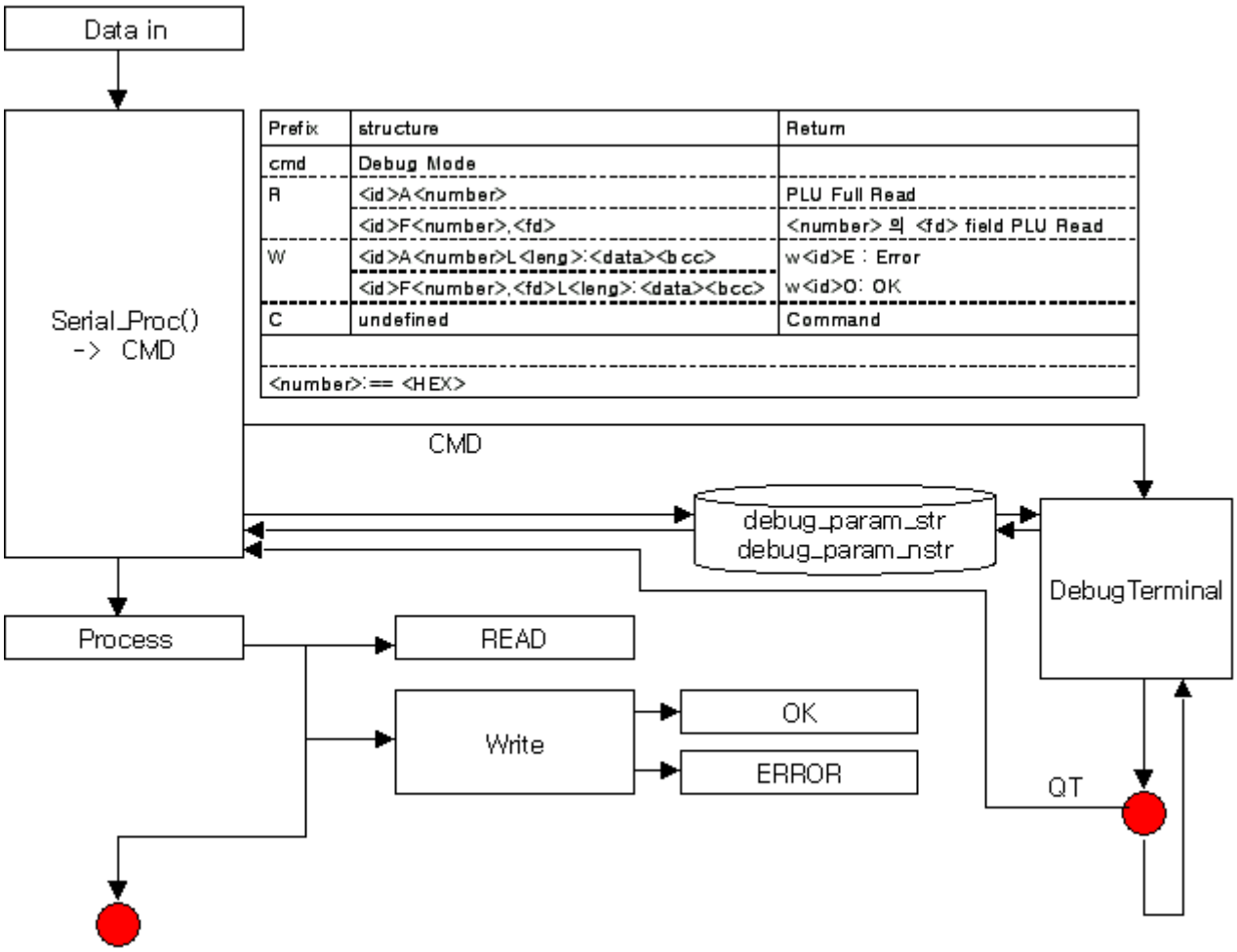


Table of Contents

1. General

CL5000 has Prefix command to calculate data. The “C” has debug command.

Network protocol has 4 types of command structure to operate.



“R” = Read, “W” =Write, “C” =Command, “I” =Information

Error command???

When Data transmitting the Checksum Error, and resending error massage to confirm;

EX) “Read” command requesting the data

R<xx>:E<error code><0x0a>

Write command:

EX) W<xx>:E<error code><0x0a>

Error code table

Read	80	Range Overflow
	81	Access Deny
	82	Mismach Receive Data or Invalid Value
	84	No Command
Command Result	77	Label Reload Complete
	45	Not exist PLU that delete
Write		
	FE	Check sum Error

Scale transmit data can have following "W" command

Scale ID "^=<scale id>."

Department ID : "*=<department id>."

Ethernet IP : "$=0."

Ethernet IP : "&=<ipaddress>"

TCP Port : "@=4F50."

2. PLU (Price Look up)

2.1. General Structure

- Download

W02A<pluno>,<deptno>L<data blocks size>:<data blocks><bcc> <data blocks> := <data block><data block>…. <data block> := "F="<ptype>.<stype>,<data size>:<data> <bcc> :== <data blocks> for all text line "xor"

Description:

"ptype" has following value;
For PLU No apply "F=02.04:<B1><B2><B3><b4>" format
EX) PLU No is 1000 <B1> Ox03 binary value
<B2> 0xE8 binary value
<B3>, <B4> = 0
<data> composed with 4byte

Name	Identify	Ptype	Max Size
Department No.	1	W	2
PLU No.	2	L	5
PLU Type	4	M	1
Name	10	S	40
Name2	30	S	40
Name3	31	S	40
Group No.	9	W	2
Ext.Barcode	49	S	20
Label No.	80	W	2
Aux. Label No.	81	W	2
Origin No.	55	W	3
Unit Weight	5	B	1
FixedWeight	100	L	6
Prefix	3	S	2

ItemCode	11	L	6
Pieces	14	W	3
Qty Unit No.	15	B	1
Use Fixed Price Type	26	B	1
Price	6	L	7
SpecialPrice	91	L	7
TAX No.	8	B	1
Tare	13	L	6
Tare No.	12	B	2
%Tare	24	W	5
Tare % limit	23	W	5
Barcode No.	85	W	2
Barcode2 No.	86	W	2
Picture No.	36	B	1
ProducedDate	20	W	3
Packed Date	18	W	3
Packed Time	19	B	2
Sell By Date	16	L	4
Sell By Time	17	B	2
CookByDate	22	W	4
Ingredient No.	25	W	4
Traceability No.	35	W	3
Bonus	50	W	3
NutriFact No.	70	W	3
Sales Msg No.	90	B	2
Reference PLU Dept	71	W	2
Reference PLU No.	69	L	5
Coupled PLU Dept	64	W	2
Coupled PLU No.	68	L	5
# of LinkPLU	60	B	1
Link PLU1 Dept	61	W	2
Link PLU1 No.	65	L	5
Link PLU2 Dept	62	W	2
Link PLU2	66	L	5

stype can have ‘S’, ‘W’, ‘L’, ‘D’, ‘T’,’B’, value

‘S’ = text line
‘W’ = 2byte short Type
‘L’ = 4byte long type

'D' = 3byte of date

'T' = 3byte of time

'B' = 1byte char Type

Return

Error : 0x82 = pluno Mismatch

NOTE: Return 0x82 , if PLU mismatch

- Upload

1. Uploading each PLU

NOTE: You need to know PLU# and department#

Each PLU Read	R13F<plunumber>,<ptype><0x0a> <plunumber> :== <2byte department number> & <6 byte plu number> R14F<plunumber>,<ptype><0x0a> Scale trasmition When PLU has been Updated Send all Plu Field when Ptype=0 Send specific Plu Field when Ptype!=0

Return:

R14F command sends the following result

W13:051<0x0a>: PLU has been erased or not exist

W13:O52<0x0a>: PLU data is already sent

For normal data send: send "W02

and scale send back: W02:O01<0x0a>

2. Reading PLU data start to end

NOTE: When scale information is unknown reading PLU data until request info its not exist.

Reading start to end	R02F<nth>,<ptype><0x0a> <nth> :== <2byte department number> <6 byte nth

	number>

➢ Delete

Thre are 3 ways to delete PLUs

1. Delete each PLU

Send	C<xx>F13,<pluno><0x0a> <pluno> := <2byte Deparment><6byte plu number> ex) deptno = 1,pluno = 16 "01000010"

2. Delete each Department

Send	C<xx>F12,<department id><0x0a>

Receive :

C<xx>

3. Delete all

Delete All	C<xx>A02<0x0a>

2.2. Stream Protocol

➢ Download

W52A<pluno>,<deptno>L<data blocks size>:<data blocks><bcc> <bcc> :== <data blocks> for all text line "xor"

Description:

<data blocks> := 0100000200123410000100000000<40 byte chars><40 byte chars>

Data components is

Byte	Items	Value
2	Department	00~99
6	PLU #	000000~999999

6	Itemcode	000000~999999
1	PLU Type	1~3
2	PCS	00~99
6	Price	000000~999999
6	Tare	000000~999999
40	Name1	String
40	Name2	String

2.2. PLU Image File Name (CL7200-S only)

➢ **Download**

Send

W02A<pluno>,<deptno>L<data blocks size>:<data blocks><bcc>

<pluno> := <5byte plu number>

<deptno> := <2byte dept number>

<data blocks> := <data block><data block>....

<data block> := "F="<Identify>.<Ptype>,<data size>:<data(binary)>

<bcc> :== XOR of <data blocks>

Receive

Success

W02:O01[0x0a][0x0a]

Error

W02:E<error code>[0x0a][0x0a]

<error code> := 0x82 : pluno mismatch

0x84 : length error

0x97 : length error(larger than buffer size)

0x99 : memory full

0xfe : checksum error

Description:

Name	Identify	Data type	Ptype	Max Size	Remark
Department No.	1	W	57	2	
PLU No.	2	L	4C	5	

PLU IMAEG File Name	101	S	53	40	Only CL7200

<Ptype> can have hexadecimal of 'S', 'W', 'L', 'D', 'T', 'B', value

S => "53" : string

W => "57" : 2byte integer

L => "4C": 4byte integer

D => "44" : 3byte of date

T => "54" : 3byte of time

B => "42" : 1byte integer

Ex) For "PLU No" apply "F=02.4C,4:<B1><B2><B3><b4>" format

"PLU No" is 1000 <B1> OxE8 binary value

<B2> 0x03 binary value

<B3>, <B4> = 0

<data> composed with 4byte

➢ **example**

- Dept 1, PLU 1, image file name : 3_6.jpg

<Send>

```
57 30 32 41 30 30 30 30 31 2C 30 31 4C 30 30 32   W02A00001,01L002
42 3A 46 3D 30 31 2E 35 37 2C 32 3A 01 00 46 3D   B:F=01.57,2:..F=
30 32 2E 34 43 2C 34 3A 01 00 00 00 46 3D 36 35   02.4C,4:....F=65
2E 35 33 2C 37 3A 33 5F 36 2E 6A 70 67 08          .53,7:3_6.jpg.
```

<Receive>

```
5E 3D 30 31 2E 2A 3D 30 31 2E 24 3D 30 2E 26 3D   ^=01.*=01.$=0.&=
30 41 30 41 33 43 37 32 2E 40 3D 30 30 30 30 2E   0A0A3C72.@=0000.
3F 3D 33 2E 57 30 32 3A 4F 30 31 0A 0A             ?=3.W02:001..
```

➢ **Upload**

3. Uploading each PLU

NOTE: You need to know PLU number and department number

Send

R13F<deptpluno>,00[0x0a]

<deptpluno> :== <2byte department number><6byte plu number>

Receive

W02A<pluno>,<deptno>L<data blocks size>:<data blocks><bcc>

<data blocks> :== <header block><plu-nth block><data block><data block>....
<plu-nth block> :== "N="<4byte plu-nth> ; if plu-nth is 0, plu not exist
<bcc> :== XOR of <data blocks>

4. Reading PLU data start to end

NOTE: When scale information is unknown reading PLU data until request info its not exist.

Send

R02F<nth>,00[0x0a]

<nth> :== <2byte department number><6byte nth number>

Receive

W02A<pluno>,<deptno>L<data blocks size>:<data blocks><bcc>

<data blocks> :== <header block><plu-nth block><data block><data block>....

<plu-nth block> :== "N="<4byte plu-nth> ; if plu-nth is 0, plu not exist

3. Table 1

3.1. Department

Department is component of name (description) and speed key.
You can set up to 5 departments.

➢ Download

W20F01,<id>L<data block size>:<data block><bcc>
<data block> :==<description><speedkey><error>
<description> :== D=<data size>.<data>
<speedkey> :== K=<data>.
<error> :== B=<0|1>[.]
<bcc> :== <data blocks> for all text line "xor"

Description

Error Code "B=1." (Do not save when this massage is send)

<speedkey> is between 1…5 value of speed key set number

Return:

For normal operation:

W20:O00001<0x0a>

For other error

W20:O<return code><0x0a>

0x99 : id,part value out of range

0x97 : wrong data

0x96 : error address calculation

or

W20F01,<id>L<data block size>:<data block>

In case of Datablock may be B=1.

- Upload

Send R20F01,<id><0x0a> Sequencial Command R21F01,<id><0x0a> <id> :== <hexadecimal value>

R21F01 … command is when requested <id>="1" and data is not exist.
This is useful for continues uploading to find request <id> and return
When Read Error

R<20|21>:E<error code><0x0a>

When Sequencial Command reaches the end, return R21:E99<0x0a> data

Receive

Transmit same data as write data

- Delete

Delete ID	C<xx>F21,01<4 byte department id><0x0a>
Delete All	C<xx>F20,001<0x0a>

Description

Department #1 is not allow to delete

You can only change data for minimize operation load

,and Its for protecting other PLU and Table data.

Return

C01:O21<0x0a>

Error :

W21:E<error code><0x0a>

0x86 : When you tiring to delete Department #1

(for #1 can be changed not delete)

0x82 : Error

3.2. Group

Benifit: Grouped PLUs are useful to make a selective report information.
Group is smallist report segment.

- Download

W20F02,<id>L<data block size>:<data block><bcc>
<data block> :==<description><error>
<description> :== D=<data size>.<data>
<error> :== B=<0|1>[.]
<bcc> :== <data blocks> for all text line "xor"

Return

Reference Department

- Upload

Send
R20F02,<id><0x0a>
R21F02,<id><0x0a>
Receive
Same as Write data

- Delete

Delete ID	C<xx>F21,02<4 byte group id><0x0a>
Delete All	C<xx>F20,002<0x0a>

Description

Return

C01:O21<0x0a>

Error :

W21:E<error code><0x0a>

0x82 : out of range

3.3. Sales Message

- Download

W20F03,<id>L<data block size>:<data block><bcc> <data block> :==<description><error> <description> :== D=<data size>.<data> <error> :== B=<0|1>[.] <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R20F03,<id><0x0a> R21F03,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F21,03<4 byte message id><0x0a>
Delete All	C<xx>F20,03<0x0a>

3.4. Origin

Downloading Origin country

- Download

W20F04,<id>L<data block size>:<data block><bcc> <data block> :==<description><error> <description> :== D=<data size>.<data> <error> :== B=<0|1>[.] <bcc> :== <data blocks> for all text line "xor"

- Upload

Send R20F04,<id><0x0a> R21F04,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F21,04<4 byte ORIGIN id><0x0a>
Delete All	C<xx>F20,04<0x0a>

3.5. Unit Symbol

- Download

W20F05,<id>L<data block size>:<data block><bcc> <data block> :==<description><error> <description> :== D=<data size>.<data> <error> :== B=<0|1>[.] <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R20F05,<id><0x0a> R21F05,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F21,05<4 byte UNIT SYMBOL id><0x0a>
Delete All	C<xx>F20,05<0x0a>

3.6. Tax

- Download

W20F06,<id>L<data block size>:<data block><bcc>
<data block> :== <tax type><tax value><error>
<tax type> :== T=<data>.
<tax value> :== V=<data>.
<error> :== B=<0|1>[.]
<bcc> :== <data blocks> for all text line “xor”

- Upload

Send
R20F06,<id><0x0a>
R21F06,<id><0x0a>
Receive
Same as Write data

- Delete

Delete ID	C<xx>F21,06<4 byte TAX id><0x0a>
Delete All	C<xx>F20,06<0x0a>

3.7. Tare

- Download

W20F07,<id>L<data block size>:<data block><bcc> <data block> :== <description><tare type><tare value><error> <description> :== D=<data size>.<data> <tare type> :== T=<data>. <tare value> :== V=<data>. <error> :== B=<0|1>[.] <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R20F07,<id><0x0a> R21F07,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F21,07<4 byte TARE id><0x0a>
Delete All	C<xx>F20,07<0x0a>

3.8. Barcode Type

You can set Barcode type for each PLUs.
Also you need to select barcode type. (Register barcode type 1~99)

- Download

W20F08,<id>L<data block size>:<data block><bcc>
<data block> :== <description><barcode type><barcode formnumber >
<barcode format><error>
<description> :== D=<data size>.<data>
<barcode type> :== T=<data>.
<barcode formnumber> :== N=<data>.
<barcode format> := F=<data size>.<data>
<error> :== B=<0|1>[.]
<bcc> :== <data blocks> for all text line "xor"

Description
When N=0 is F(there are meaning)
When N=0 is must download format otherwise in real usage may cause problem.
When N≠0 is Barcode Format must input within 30 text letters

Return

Reference

- Upload

Send
R20F08,<id><0x0a>
R21F08,<id><0x0a>
Receive
Same as Write data

- Delete

Delete ID	C<xx>F21,08<4 byte BARCODE id><0x0a>
Delete All	C<xx>F20,08<0x0a>

4. Table2

Table2 is one of extension of PLU's Table contains; Ingredient, Nutrition-Facts, Traceability (Country, Slaughter House, Cutting Hall)

4.1. Ingredient

Each PLU has individual ingredient factors in table format. This table links with ingredient number.

- Download

```
W30F01,<id>L<data block size>:<data block><bcc>
  <data block>       :==  <block number><text block>
  <text block>       :==  D=<text data size>.<text data>
  <block number>     :==  X=<nth block>.
  <nth block>        :==  0 : Start
                          1 block size = 512
  Sending data: text data must send at last
  <error>            :==  B=<0|1>[.]
  <bcc> :== <data blocks> for all text line "xor"
```

Description

<id> can have value 1~9999

<text data> data size can not be bigger then 512Byte per 1 transition

<block number> is for sending Text data which is greater then 512

1 Block (512 Byte) is Block number

Sending smaller then 512 Byte, you don't need to send <block number>

Return

0x82 <id> is invalid

0x99 <data> sending error, Format (byte)size not allow to decode

0x99 <id> exceed max ingredient number

0x83 no format exists

0x90 <nth block> exceed max block number

0x01 Write OK.

➢ Upload

Send

R30F01,<id><0x0a> : If <id> don't exist, return <error code>

R31F01,<id><0x0a> : If <id> don't exist, return next <id>'s data

<id> :== more then 4Byte text lines

000100 is first 4letter=("0001") indicates Ingredient의ID

Next 2 text ("00") indicates Data Block number

Receive

W30F01,<id>L<data block size>:<data block>

<data block> :== <block number><text block>

<text block> :== D=<text data size>.<text data>

<block number> :== X=<nth block>.

<nth block> :== 0 : Start, 0xFF : End

1 block size = 512

R30:E<error code><0x0a> or R31:E<error code><0x0a>

<error code> :== 0x99 : <id> exceed max. ingredient number

0x95 : <nth> exceed valid block number

0x82 : <id> is 0 or <id> don't exist

Description

"R31F01,.." recall request Ingredient ID, if not search next ID and return

If there are no other value receive Wxx:E99

➢ Delete

Delete ID	C<xx>F31,01<4 byte INGREDIENT id><0x0a>
Delete All	C<xx>F30,01<0x0a>

Description

Return

After Delete All command Return time may take a while to receive return massage

OK: "C001:O0030"

4.2. Nutrition Facts

USA FDA(Food and Drug Administration) regulates certain products must print following factors.

Download

W30F02,<id>L<data block size>:<data block><bcc>
<data block> :== D=<data size>.<data>
:== T=<nutri-fact Type>. (0=SHORT,1=LONG)
:== S=<data size>.<data> ; Serving Size
:== P=<data size>.<data> ; Serving Per
:== Z=<nutrifact-id>:<4 byte value>.
<bcc> :== <data blocks> for all text line "xor"

Description

<nutrifact-id> table

00	calories
01	Calories fat
02	Total Fat
03	Saturated Fat
04	Cholesterol
05	Sodium
06	Total Carbon
07	Dietary Fibers
08	Sugers
09	Protein
0A	Vitamin A
0B	Cacium
0C	Vitamin C
0D	Iron
0E	Etc

- Read

Send
R30F02,<id><0x0a>
R31F02,<id><0x0a>
Receive
Same as Write data.

- Delete

Delete ID	C<xx>F31,02<4 byte nutrition id><0x0a>
Delete All	C<xx>F30,02<0x0a>

Return

OK : "C001:O0030"

4.3. Traceability

This is Meet, Fish, other related product history coding.
This code contains; Cutting hall, Slaughter house, bred country, born country)
You must input all the information of Traceability factor; (4-4,4-5,4-6) code must be register.
*Born, Bred, is follows each country standard.

- Download

```
W30F03,<id>L<data block size>:<data block><bcc>
<data block> :==   D=<data size>.<data>
                :==   Z=<id>:<4 byte value>.
<id> : 0 <born in country no>
      : 1 <bred in country no>
      : 2 <Slaughter house no>
      : 3 <Cutting Hall No>
<bcc> :== <data blocks> for all text line "xor"
```

- Upload

```
Send
   R30F03,<id><0x0a>
   R31F03,<id><0x0a>
Receive
   Same as Write data
```

- Delete

Delete ID	C<xx>F31,03<4 byte traceability id><0x0a>
Delete All	C<xx>F30,03<0x0a>

Return
OK : "C001:O0030"

4.4. Country

For traceability code you must insert country.
You can set country code freely. Therefore need to maintain each country code update manually.

- Download

W30F04,<id>L<data block size>:<data block><bcc> <data block> :== D=<data size>.<data> <bcc> :== <data blocks> for all text line "xor"

- Upload

Send R30F04,<id><0x0a> R31F04,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F31,04<4 byte country id><0x0a>
Delete All	C<xx>F30,04<0x0a>

Return
OK : "C001:O0030"

4.5. Slaughter House

Traceability code element (you need to set country)

- Download

W30F05,<id>L<data block size>:<data block><bcc> <data block> :== <description><country> <description> :== D=<data size>.<data> <country> :== C=<country number>. <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R30F05,<id><0x0a> R31F05,<id><0x0a> Receive Same as Write data

- Delete

Delete ID	C<xx>F31,05<4 byte slaughter id><0x0a>
Delete All	C<xx>F30,05<0x0a>

OK : “C001:O0030”

4.6. Cutting Hall

Traceability code element (need to set country)

- Download

W30F06,<id>L<data block size>:<data block><bcc> <data block> :== <description><country> <description> :== D=<data size>.<data> <country> :== C=<country number>. <bcc> :== <data blocks> for all text line "xor"

- Upload

Send R30F06,<id><0x0a> R31F06,<id><0x0a> Receive Same as Write data

- Delete

Individual deleting command

Delete ID	C<xx>F31,06<4 byte cuttinghall id><0x0a>

Delete all command

Delete All	C<xx>F30,06<0x0a>

Return

OK : "C001:O0030"

5. Store,Customer,Scroll Message and Clerk Table

5.1. Store

Inset store data or delet

- Download

W32F01,<id>L<data block size>:<data block><bcc> <data block> :== P=<data size>.<data> ; Store name :== T=<data size>.<data> ; Store tel :== S=<data size>.<data> ; Store description <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R32F01,<id><0x0a> R33F01,<sequential no.><0x0a> Receive Same as Write data

- Delete

Individual deleting command

Delete ID	C<xx>F33,01<id><0x0a>

Delete all command

Delete All	C<xx>F32,01<0x0a>

Return

OK : “C001:O0033”

5.2. Customer

Inset Customer information or delete

Download

W32F02,<id>L<data block size>:<data block><bcc> <data block> :== P=<data size>.<data> ; Customer name :== T=<data size>.<data> ; Customer tel :== C=<Credit Limit>. ; Customer credit limit :== Q=<data size>.<data> ; Customer address 1 :== R=<data size>.<data> ; Customer address 2 :== S=<data size>.<data> ; Customer address 3 <bcc> :== <data blocks> for all text line "xor"

- Upload

Send R32F02,<id><0x0a> R33F02,<sequential no.><0x0a> Receive Same as Write data

- Delete

Individual deleting command

Delete ID	C<xx>F33,02<id><0x0a>

Clerk 1 impossible to delete

Delete all command

Delete All	C<xx>F32,02<0x0a>

Return

OK : "C001:O0033"

5.3. Scroll Message

Insert Clerk information and delete

- Download

W32F04,<id>L<data block size>:<data block><bcc>
<data block> :== E=<effect data>.
:== S=<data size>.<data> ; Scroll message
:== A=<daily>.
:== B=<use>.
:== w=<week>.
:== h=<start date>.
:== i=<end date>.
:== j=<start time>.
:== k=<end time>.
<bcc> :== <data blocks> for all text line "xor"

- Upload

Send
R32F03,<id><0x0a>
R33F03,<sequential no.><0x0a>

Receive
Same as Write data

- Delete

Individual deleting command

Delete ID	C<xx>F33,03<id><0x0a>

Clerk is impossible to delete

Delete all command

Delete All	C<xx>F32,03<0x0a>

Return
OK : "C001:O0033"

5.4. Clerk

Insert Clerk information and delete

- Download

```
W32F04,<id>L<data block size>:<data block><bcc>
<data block> :== P=<data size>.<data>   ; Clerk name
             :== Q=<data size>.<data>   ; Clerk nickname
             :== R=<data size>.<data>   ; Clerk password
             :== L=<level>.
<bcc> :== <data blocks> for all text line "xor"
```

- Upload

```
Send
   R32F04,<id><0x0a>
   R33F04,<sequential no.><0x0a>

Receive
   Same as Write data
```

- Delete
-

Individual deleting command

Delete ID	C<xx>F33,04<id><0x0a>

Clerk 1 is impossible to delete

Delete all command

Delete All	C<xx>F32,04<0x0a>

Return

OK : "C001:O0033"

Error:

W32:E<code><0x0a>

<code> : 0x86 : When delete Clerk 1

5.5. Currency

Insert Currency information and delete

- Download

```
W32F05,<id>L<data block size>:<data block><bcc>
<data block> :== P=<data size>.<data>   ; Primary Sign
             :== Q=<data size>.<data>   ; Last Sign
             :== h=<rate>.
             :== i=<decimal point>.
<bcc> :== <data blocks> for all text line "xor"
```

- Upload

```
Send
   R32F05,<id><0x0a>
   R33F05,<sequential no.><0x0a>

Receive
   Same as Write data
```

- Delete

Individual deleting command

Delete ID	C<xx>F33,05<id><0x0a>

Delete all command

Delete All	C<xx>F32,05<0x0a>

Return

OK : "C001:O0033"

5.6. Scanner

Insert Scanner information and delete

- Download

W32F06,<id>L<data block size>:<data block><bcc> <data block> :== P=<data size>.<data> ; Mapping code :== h=<4 byte department id> :== j=<6 byte PLU id> <bcc> :== <data blocks> for all text line “xor”

- Upload

Send R32F06,<id><0x0a> R33F06,<sequential no.><0x0a> Receive Same as Write data

- Delete

Individual deleting command

Delete ID	C<xx>F33,06<id><0x0a>

Delete all command

Delete All	C<xx>F32,06<0x0a>

Return

OK : “C001:O0033”

6. Discount Table

Setting PLU Discount value

➢ Download

W09F<dcno>,<mode>L<data block size>:<data block><bcc> <data block> :== <detail data> <detail data> :== <data index>=<value>. <bcc> :== <data blocks> for all text line "xor"

Description

Send Discount Table

<data index> must following table and write in hexadecimal

<mode> can have 0|1|2 value, for this case use "0"

1,2 use for scale to scale transition

<dcno> use temperate value and reserved

<data index>

sign	Description	Example
a	Department No	a=01.
b	Plu No.	b=01.
c	Discount Type	c=1.
d	Target 0	d=
e	Target 1	
f	Price 0	
g	Price 1	
h	Start Date <YY><MM><DD>	h=040717.
i	Start Time <HH><MM><SS>	i=080000.
J	End Date <YY><MM><DD>	J=040830.
k	End Time <HH><MM><SS>	k=000000.
l	Week (b0=Sun Day b1=Mon Day…)	
m	Use FSP(0=not use, 1=use)	M=01.

<data index>에서 Discount type 은 Discount Code가 다음과 같이 정의 된다.

Discount Code	Description
0x1CFF	Unselected no Discount

0x1C97	Unit price discount – weight
0x1C9B	Unit price discount - Count
0x1CA0	% Unit price discount - Weight
0x1CA1	% Unit price discount - Count
0x1C96	Total price discount - Weight
0x1C9A	Total price discount - Count
0x1C99	% Total price discount - Weight
0x1C9D	% Total price discount - Count
0x1C9E	Fixed price – Weight Fixed price – Count
0x1C9F	Free Item – Weight Free Item – Count
0x1C98	Free Addition – Weight
0x1C9C	Free Addition – Count
0x1C91	Total price by Total price – Weight Total price by Total price – Count
0x1C95	% Total price by Total price – Weight % Total price by Total price – Count
0x1C92	Unit price by Total price – Weight Unit price by Total price – Count
0x1C93	Extra by Total Price - Weight
0x1C94	Extra by Total Price - Count

Return

0x99 : error value

0x98 : no room for record

0x97 : Download error

- Upload

R09F<pluno>,Kdeptno><0x0a> R10F<xx>,<dc_no><0x0a>

Description

Upload method of <department number>;

You can enter start number to search the most closet list of discount plu

Receive

Same as Write data

Reading R10F if there no more of data sends Error 0x95

Error

R10:E<code>

- Delete

Delete ID	C<action>F09,<value><0x0a> <value> :== <2byte department id><6byte pluid>
Delete All	C<action><A|F>10<0x0a>

Return

OK: "C001:O0010<0x0a>"

ERROR:

7. Report

7.1 Report

- **Upload**

R43F<period and part>,<6byte id><0x0a>
<period and part>:== <period><part>
<period> :== < 2byte data>
<part> :== <2byte part (Scale,PLU,MISC,Group,Department,Hourly,Clerk) >
<6byte id> :== 0.. max

Description

Report contains Part, Period. You can upload each of it

For this purpose Part indicates report part, and Period indicates periodic time.

NOTE: for PC control select Z1 or Z2

Part

Part #	
1	Report from each scale
2	PLU and Non PLU Report
3	
4	Group Report
5	Department Report
6	Hourly Report
7	Clerk Report
8	Tax Report
9	All Part (Only for “Clear All”)

Period

Period	
0	X1 Report
1	X2 Report
2	All X1,X2 (Only for Clear)

Return

W43F<period and part>,<6byte id>L<data block size>:<data block>
<period and part>:== <period><part>
<period> :== < 2byte data>
<part> :== <2byte part (Scale,PLU,MISC,Group,Department,Hourly,Clerk) >
<data block> :== <part value>
<part value> :== N<report local id>=<transaction number>,<volume>.

<part value> contains; <report local id> can have 00~0F value and each Part has different value.

Scale report

<data block> :== <part value>
<part value> :== b=<scale id>. ; Scale ID (1..31)
N<report local id>=<transaction number>,<volume>.
W01=<weight>.
Q01=<quantity>.
K=<cash drawer summary>.
S<pay id>=<transaction number>,<volume>.
y=<date>. ; yymmdd
h=<time>. ; hhmmss

<report local id> in Scale Report

01	Scale Summary	
02	Void Summary	
03	Repack summary	
04	Override summary	
05	Discount Summary	
06	Prepack Summary	
07	Return Summary	
08	reserved	NO use
09	Change	
0A	Customer Summary	

<pay id> in Scale Report

01	Cash	
02	Pin/Chip	
03	Check	
04	Credit Card	
05	Credit Note	
06	Coupon	
07	Bonus Point	
08	Credit Sales	

PLU report

<id> : <plu save #>

<data block> :== <item value>

<item value> :== <identities>=<hex value>.

<identities> table

E	=0. normal/ =1. no data(end)	
n	Total sale count.	
P	PLU #	
D	Department #	
T	PLU Type	
V1	Total sales price	Period
W1	Total sales weight	Period
Q1	Total sales count.	Period
V2	Total Label sales	Period
C2	Total Label print-outs	Period
V3	Total Pre-pack sales	Period
W3	Total Pre-pack sales weight	Period
X	Total Tax price	Period

Group report

<id> : Group Number

<data block> :== M=<transaction number>,<volume>.
W=<weight>.
Q=<quantity>.
y=<date>. ; yymmdd
h=<time>. ; hhmmss

Department report

<id> : Department Number

<data block> :== M=<transaction number>,<volume>.
W=<weight>.
Q=<quantity>.
y=<date>. ; yymmdd
h=<time>. ; hhmmss

Hourly report

<id> : 1..24 (HR)

<data block> :== M=<transaction number>,<volume>.
W=<weight>.
Q=<quantity>.
y=<date>. ; yymmdd
h=<time>. ; hhmmss

Clerk report

<id> : 1..99

<data block> :== <part value>

<part value> :== N<report local id>=<transaction number>,<volume>.
W01=<weight>.
Q01=<quantity>.
K=<cashdraw summary>.
S<pay id>=<transaction number>,<volume>.
y=<date>. ; yymmdd
h=<time>. ; hhmmss

<report local id> in Scale Report

01	Clerk Summary	
02	Void Summary	
03	Repack summary	
04	Override summary	
05	Discount Summary	

06	Prepack Summary	
07	Return Summary	
08	reserved	
09	Change	
0A	unused	
0B	Negative Summary	

Tax report

<id> : Tax Number

<data block> :== <part value>

<part value> :== Y<Tax id>=<tax code><tax rate>,<volume>.

y=<date>. ; yymmdd

h=<time>. ; hhmmss

- **Clear and Reset**

Clear by Period and Part	C00F43,<value><0x0a> <value> :== <2byte period>< 2 byte part>
Clear All	C00F43,09<0x0a>

Clear All command is same as C00A42<0x0a>

Return

O K: C001:O0043

ERR:

7.2. Real Time Report

- **Scale → PC**

I00F070,<id>L<data block size>:<data block >
<data block> : == <clerk Id><tail><transaction_record_type><cs>
<clerk Id> : == C=<2 byte data>. Ex) C=02. => clerk 2
<tail> : == T=<4 byte data>. Ex) T=0003.

Description

Scale에서 판매를 하면, 판매 데이터가 PC로 실시간으로 전송된다.

Transaction_record_type은 아래와 같다.

Transaction_record_type : 56 byte, binary

	Item	Size (Byte)
Transaction_record_type	Transaction Type (void..)	1
	Scale Id	1
	Serial Number	1
	Status	1
	PLU type	1
	Department Number	1
	PLU Number	4
	Group	1
	Operator (Clerk)	1
	Code	4
	Weight	4
	Qty	2
	PCS	2
	Unit Price	4
	Volume (Total price)	4
	Discount Volume	4
	Weight Unit	2
	Transaction Counter	4
	Tax Type	1
	Tax Number	1
	Tax	4
	Ticket Number	4
	Status2	1
	Individual No Index	1
	Credit Customer No	2

※ Item Definition

Item	Description	Value	Remark
Transaction Type	Normal type(Default)	0	기본판매

(1byte)	Repack type	1	연습판매
	Self Service type	2	셀프서비스 판매
Status (1byte)	Negative sale	0x01(Bit 0)	네거티브 판매
	Return	0x02(Bit 1)	반품
	Void	0x04(Bit 2)	판매취소
	Prepack	0x08(Bit 3)	포장
	Label	0x10(Bit 4)	라벨판매
	Override	0x20(Bit 5)	가격변경판매
	Add	0x40(Bit 6)	합산판매
	No Void	0x80(Bit 7)	판매취소 해제
Status2 (1byte)	Local sale	0x01(Bit 0)	로컬모드 판매
	Disconnection	0x02(Bit 1)	마스터-PC 연결끊김

➢ **PC → Scale(ACK)**

I00F070,<id>L<data block size>:<data block >
<data block> : == <clerk Id><tail><cs>
<clerk Id> : == C=<2 byte data>. Ex) C=02. => clerk 2
<tail> : == T=<4 byte data>. Ex) T=0003.

<주의>

1. <tail>에 해당하는 값은 저울로부터 수신한 <tail> 값과 동일한 값을 전송해야 한다.
2. V1.37.0 이전 버전은 <tail>이 2byte data로 되어 있다.

8. Label and Image

8.1. Label format

The maximum save Label format is 20. You can set Label ID, but Label ID 1~30 is already been set in scale system.

<Memory map>

Label Area	Size (byte)	Type	Qty	Subtotal	Pos	Define	Value
Label ID	2	word	20	480	0	LABEL_INFO_POINT	0
Width	2	word					
Height	2	word					
Label Name	16	byte					
Label Image size	2	word					
Label Image	4096	byte	20	81920	480	LABEL_IMAGE	480

8.2. Bitmap size (for printing size allowance)

When printing bitmap image on label; you need to save image in differently.
The max saving space is up to 14 images.
Also each image can set ID number 1~50.

<Memory map of Bitmap>

Label Area	Size (byte)	Type	Qty	Subtotal	Position	Define	Value
Bitmap ID	2	word	14	84	0	BITMAP_INFO_POINT	0
Bitmap Width	2	word					
Bitmap Height	2	word					
Bitmap Image	8192	byte	14	114688	84	BITMAP_IMAGE	84

Matching the Label format ID and bitmap ID will print image on the Label.

NOTE: In case of printing multiple images on one label bitmap ID must be different.

8.3. Data File

In CL5000 has Label data which contains; LFM and bitmap image

A. LFM Data Structure

LFM format contains Label Format.

This information must be decoded order to be download

Label Format File (LFM) structure

Label Header	Bitmap Header 5
4096 Byte Label Format (MAX 4096 byte)	
Bitmap Header size and location Bitmap Data	

LABEL HEADER

NAME	Type	Bytes	Description	
Format	char	6	Label format	Label Information
Version	char	2	Label make Version	
Company	char	10	Label maker information	
Model	char	4		
Usage	char	4		
id	ushort	2	Label ID (1~999)	
width	ushort	2	Label width	
height	ushort	2	Label height	
name	char	16	Label name	
size_label	ulong	4	Label Format size	
CMPHEADER	struct	14*5	Bitmap Structure	

Label Format can have 5 Bitmap-information in one label

This information can have different locations, saved address, size.

NAME	Type	Bytes	Description	
addr	long	4	Data address(location)	Bitmap Information
size	long	4	Data Size	

id	ushort	2	Bitmap ID	
width	ushort	2	Printable width	
height	ushort	2	Printable height	

B. Bitmap image

Download structure

NAME	Type	Bytes	Description	
Format	char	6	Label format	Label Information
Version	char	2	Label Version	
Company	char	10	Label Company	
Model	char	4		
Usage	char	4		
id	ushort	2	Label ID (1~999)	
width	ushort	2	Label width	
height	ushort	2	Label height	

8.4. Transfer

- Download

```
W06F<bin id>,<mode>L<data block size>:<data block>
<bin id> := <4byte id>
<mode> := <2byte form><2byte set>
<2byte form> := 1 : Label
                2 : Bitmap
<2byte set> := 1 : Header
             = 2: Data
<data block> :== <detail data>
<detail data>
          = "N="<nblock>.
          = "Z="<name>.
          = "S=<total image size>.
          = "W="<width>.
          = "H="<height>.
          = "D="<size>:<binary data>
```

Description

<nblock> is 1~99

Return

R06:E99 // data error

R06:E98 // no room for save

R06:O043 // Label Header save success

R06:O044 // Bitmap Header save success

Check and Apply

This present setting label format not applied onto scale yet. This following command will switch to new set data.

Check And Apply	C<action>A05<0x0a> ex) C01A05<0x0a>

Return

C<action>:O05<0x0a>

- Delete

Clear Label	C<action>F07,<value><0x0a> <value> :== <2byte type>< 4byte value>
Clear All	C<action>F06,<mode><0x0a>

<2byte type> = 1: Label
2: Bitmap
<4byte value> = <label or bitmap id>
<mode> = 1: Label , 2: Bitmap

9. Keypad

9.1. Speed Key Set

- **Upload**

R04F21,<key no><0x0a>	Speed Key set #1 Upload
R04F22,<key no><0x0a>	Speed Key set #2 Upload
R04F23,<key no><0x0a>	Speed Key set #3 Upload
R04F24,<key no><0x0a>	Speed Key set #4 Upload
R04F25,<key no><0x0a>	Speed Key set #5 Upload

<key no> := 00

If "00" not, allow 0x01 ~ 0xA0 value. Each set value need to Return

- **Download**

W04F21,<key no>L<data size>:<data block><bcc>
W04F22,<key no>L<data size>:<data block><bcc>
W04F23,<key no>L<data size>:<data block><bcc>
W04F24,<key no>L<data size>:<data block><bcc>
W04F25,<key no>L<data size>:<data block><bcc>

<data block> MAX 160 x 4 = 640 Byte able to send
(1 data if organized with 4byte)

Send data and download data has following structure.
<key no> = Set 0 to PLU #1
<key no> = Set 1 to PLU #2

4byte value 1	4byte value 2				

NOTE: DATA is following the "Intel save real number" format

9.2. Function Key setting

➢ **Download**

W04F<keyhwid>,<key hwno>L<data size>:<data block><bcc>

<keyhwid> := 01 -> Sale Key Normal Mode set
02 -> Sale Key Shift Mode set
03 -> Program Key Normal Mode set
04 -> Program Key Normal Mode set
05 -> Second Program Key Normal Mode set
06 -> Second Program Key Normal Mode set
07 -> Customer Key set (spare)
11 -> Sale Key Normal + Shift
12 -> Program Key Normal + Shift
13 -> Second Program Key Normal + Shift
31 -> Clerk Set
32 -> Department Set
33 -> Tare Set
34 -> Currency Set

<key hwno> = 0 sending all block data
<key hwno> <> 0 modify call ID
1 data structured with 2byte

➢ **Upload**

R04F<keyhwid>,<key hwno><0x0a>

10. Keypad (CL7200-S only)

10.1. CL7200-S Key Download Procedure

1. Download PLU Image Files (Using FTP)
   - Dowload Folder : /image
2. Download PLU Image File name(refer chapter 2)

3. Download Keypad
   A. Download Speed Key Set (refer chapter 3.2)
   B. Download Speed Key Properties (refer chapter 3.3)
   C. Download Product No (refer chapter 3.4)

※ FTP settings

PORT : 21

ID : cas

Password : cascl7200

10.2. Speed Key Set

➢ **Upload**

R04F15,<key no>[0x0a] ; Category #1 Upload
R04F16,<key no>[0x0a] ; Category #2 Upload
R04F17,<key no>[0x0a] ; Category #3 Upload
R04F18,<key no>[0x0a] ; Category #4 Upload

<key no> is a value that you want to set (1~200).

If value is 0, scale transamits all key value.

■ **example**

- Category 1(0x15) (Upload)

<Send> 52 30 34 46 31 35 2C 30 30 30 30 0A	R04F15,0000.
<Receive> 57 30 34 46 30 32 30 34 31 35 2C 30 30 30 30 4C 30 33 32 30 3A 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00 05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00 09 00 00 00 0A 00 00 00 0B 00 00 00 0C 00 00 00 0D 00 00 00 0E 00 00 00 0F 00 00 00 10 00 00 00 11 00 00 00 12 00 00 00 13 00 00 00 14 00 00 00 15 00 00 00 16 00 00 00 17 00 00 00 18 00 00 00 19 00 00 00 1A 00 00 00 1B 00 00 00 1C 00 00 00 1D 00 00 00 1E 00 00 00 1F 00 00 00 20 00 00 00 21 00 00 00 22 00 00 00 23 00 00 00 24 00 00 00 25 00 00 00 26 00 00 00 27 00 00 00 28 00 00 00[…] 28	W04F020415,0000L0320:.. ....!..."... #...$...%...&...'...(....[…](

➢ **Download**

Send

W04F<selfkey_type><scale_type>15,<key no>L<data size>:<data blocks><bcc>

W04F<selfkey_type><scale_type>16,<key no>L<data size>:<data blocks><bcc>

W04F<selfkey_type><scale_type>17,<key no>L<data size>:<data blocks><bcc>

W04F<selfkey_type><scale_type>18,<key no>L<data size>:<data blocks><bcc>

Receive

W60:O01[0x0a][0x0a] : success

<selfkey_type> 01:7X4, 02:8X5, 03:9X6, 04:6X3

<scale_type> 01:standard, 02:pole, 03:hanging, 04:self-service

<data blocks> structure (binary. MAX 200 x 4 = 800 Byte)

0	1	2	3	…	…	199
PLU No (4Byte)	PLU No (4Byte)					PLU No (4Byte)

NOTE: DATA is using the "little-endian" format

<bcc> XOR of <data blocks>

■ **example**

◆ Category 1(0x15) (Download)

<Send> 57 30 34 46 30 32 30 34 31 35 2C 30 30 30 30 4C 33 32 30 3A 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00 05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00 09 00 00 00 0A 00 00 00 0B 00 00 00 0C 00 00 00 0D 00 00 00 0E 00 00 00 0F 00 00 00 10 00 00 00 11 00 00 00 12 00 00 00 13 00 00 00 14 00 00 00 15 00 00 00 16 00 00 00 17 00 00 00 18 00 00 00 19 00 00 00 1A 00 00 00 1B 00 00 00 1C 00 00 00 1D 00 00 00 1E 00 00 00 1F 00 00 00 20 00 00 00 21 00 00 00 22 00 00 00 23 00 00 00 24 00 00 00 25 00 00 00 26 00 00 00 27 00 00 00 28 00 00 00 […] 00 00 00 00 28	W04F020415,0000L 320:..... ....!... "...#...$...%...&...'...(... .(
<Receive> 57 30 34 3A 4F 30 31 0A 0A	W04:001..

10.3. Set Speed Key property (speed key design)

➢ **Download**

Send

W60F<selfkey_type><scale_type><keyset no>,<key no>L<data size>:<data blocks><bcc>

Receive

success

W60:O01[0x0a][0x0a]

error

W60:E80[0x0a][0x0a] ; keyset no error

W60:E82[0x0a][0x0a] ; scale type mismatch(or self key type mismatch)

W60:E83[0x0a][0x0a] ; key no error

W60:E84[0x0a][0x0a] ; data range error

<selfkey_type> 01:7X4, 02:8X5, 03:9X6

<scale_type> 01:Standard, 02:Pole, 03:hanging, **04:Self service**

<keyset no>

keyset no	description
0x15	Category1's Properties
0x16	Category2's Properties
0x17	Category3's Properties
0x18	Category4's Properties

<key no> is a value that you want to set (1~200)

<data blocks> structure (binary)

Name		Size(byte)	
Font size		1	10~40 (default:13)
Key Size		1	**not use (set 1) in CL7200-S**
Font Color Code		1	1:black, 2:red, 3:green, 4:blue, 5:yellow, 6:aqua, 7:pink, 8:white, 9:gray (default:1)
Background Color Code		1	**not use (set 1) in CL7200-S**
Font Color RGB		3	**not use (set 0) in CL7200-S**
Background Color RGB		3	**not use (set 0) in CL7200-S**

<bcc> XOR of <data blocks>

■ **example**

◆ 02 : self key type (5X8), 04 : scale type (Self Service)

◆ Category number 1 / Speed key property for 1 number

◆ Font size 13, Font Color Code 1(Black)

<Send> 57 36 30 46 30 32 30 34 31 35 2C 30 30 30 31 4C 30 30 41 3A 0D 01 01 01 00 00 00 00 00 00 0C	W60F020415,0001L00A :.....
<Receive> 57 36 30 3A 4F 30 31 0A 0A	W60:001..

※ Font Color

Code	Color	Decimal			Hex		
		R	G	B	R	G	B
1	**black**	0	0	0	0	0	0
2	**red**	255	0	0	FF	0	0
3	**green**	0	255	0	0	FF	0
4	**blue**	0	0	255	0	0	FF
5	**yellow**	255	255	0	FF	FF	0
6	**aqua**	0	255	255	0	FF	FF
7	**pink**	255	0	255	FF	0	FF
8	**white**	255	255	255	FF	FF	FF
9	**gray**	178	178	178	B2	B2	B2

➢ **Upload**

Send

R60F<keyset no>,<key no>[0x0a] ; Category n번 Upload

Receive

success

W60F<selfkey_type><keyset no>,<key no>L<data size>:<data blocks><bcc>

error

R60:E80[0x0a] ; keyset no. error

R60:E83[0x0a] ; key no. error

■ **example**

◆ 02 : self key type (5X8), 04 : scale type (Self Service)

◆ Category number 1 / Speed key property for 1 number

◆ Font size 13, Font Color Code 1(Black)

<Send> 52 36 30 46 31 35 2C 30 30 30 31 0A	R60F15,0001.
<Receive> 57 36 30 46 30 32 30 34 31 35 2C 30 30 30 31 4C 30 30 30 41 3A 0D 01 01 01 00 00 00 00 00 00 0C	W60F020415,0001L000A :. . . . .

10.4. Set Product No.

➢ **Upload**

Send R04F<keyset no>,<key no>[0x0a] **Receive** W04F<selfkey_type><scale_type><keyset no>,<key no>L<data size>:<data blocks><bcc>

Ex)R04F33,00

keyset no	description
0x33	Category1's Product no table
0x34	Category2's Product no table
0x35	Category3's Product no table
0x36	Category4's Product no table

■ **example**

◆ Category number 1 /Product Number

<Send> 52 30 34 46 33 33 2C 30 30 30 30 0A	R04F33,0000.
<Receive> 57 30 34 46 30 32 30 34 33 33 2C 30 30 30 30 4C 30 31 39 30 3A 64 00 65 00 66 00 67 00 68 00 69 00 6A 00 6B 00 6C 00 6D 00 6E 00 6F 00 70 00 71 00 72 00 73 00 74 00 75 00 76 00 77 00 78 00 79 00 7A 00 7B 00 7C 00 7D 00 7E 00 7F 00 80 00 81 00 82 00 83 00 84 00 85 00 86 00 87 00 88 00 89 00 8A 00 8B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	W04F020433,0000L0190 :d.e.f.g.h.i.j.k.l.m .n.o.p.q.r.s.t.u.v.w .x.y.z.{.|.}.~.□□. □.,.ƒ.„.…. †.‡.ˆ. ‰.Š.‹. . . . .

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	. . . .

- **Download**

Send

W04F<selfkey_type><scale_type><keyset no>,<key no>L<data size>:<data blocks><bcc>

Receive

success

W04:O01[0x0a][0x0a]

error

W04:E80[0x0a][0x0a] ; keyset no error

W04:E81[0x0a][0x0a] ; data size error

W04:E82[0x0a][0x0a] ; scale type mismatch(or self key type mismatch)

W04:E83[0x0a][0x0a] ; key no error

W04:E84[0x0a][0x0a] ; data range error

<selfkey_type> 01:7X4, **02:8X5**, 03:9X6

<scale_type> 01:Standard, 02:Pole, 03:hanging, **04:Self service**

<data block> structure (binary. MAX 200 x 2 = 400 Byte)

0	1	2	3	...	...	199
Product No (2Byte)	Product No (2Byte)					Product No (2Byte)

NOTE: DATA is using the “little-endian” format

- **example**

◆ Category number 1 /Product Number

<table>
<tr><td><Send>
57 30 34 46 30 32 30 34 33 33 2C 30 30 30 30 4C 31 39 30 3A 64 00 65 00
66 00 67 00 68 00 69 00 6A 00 6B 00 6C 00 6D 00 6E 00 6F 00 70 00 71 00
72 00 73 00 74 00 75 00 76 00 77 00 78 00 79 00 7A 00 7B 00 7C 00 7D 00
7E 00 7F 00 80 00 81 00 82 00 83 00 84 00 85 00 86 00 87 00 88 00 89 00
8A 00 8B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00</td><td>
W04F020433,0000L
190:d.e.f.g.h.i.j.k.l.m.
n.o.p.q.r.s.t.u.v.w.x.y
.z.{.|.}.~.□□ .□ .,.ƒ.„.
....†.‡.ˆ.‰.Š.‹..............
....
....
....
....
....
....
....
....
........................</td></tr>
<tr><td><Receive>
57 30 34 3A 4F 30 31 0A 0A</td><td>
W04:001..</td></tr>
</table>

11 Remote Monitoring & Real Time Alarm

11.1. Weight

Send

I00A020[0x0a]

Receive

I00:L<data size>,<data blocks><bcc>

<data blocks>

:== "c=020". ; cmd

"W="<data>. ; Weight

"T="<data>. ; Tare Weight

"D="<data>. ; Weight decimal position

"U="<data>. ; Unit Price

"t=" <data>. ; Total Price

"d="<data>. ; Price decimal position

<bcc> :== XOR of <data blocks>

ex) I00:L02C,c=020.W=000204.T=000000.D=3.U=500.t=102.d=2.<bcc>

11.2. Printer

Send

I00A021[0x0a]

Receive

I00:L<data block size>,<data blocks><bcc>

<data blocks>

:== "c=021". ; cmd

"I="<data>. ; Gap Min Value

"A="<data>. ; Gap Max Value

"T="<data>. ; Gap Threshold Value

"i="<data>. ; Peel Min Value

"a="<data>. ; Peel Max Value

"t="<data>. ; Peel Threshold Value

<bcc> :== XOR of <data blocks>

ex) I00:L026,c=021.I=29.A=91.T=60.i=36.a=190.t=188.

11.3. Scale Error

Send

I00A022[0x0a]

Receive

I00:L<data block size>,<data block><bcc>

<data block>

:== "c=022". ; cmd

"H="<data>. ; Head up. 0=no error, 1=error

"P="<data>. ; Peel off. 0=no error, 1=error

"E="<data>. ; Paper end. 0=no error, 1=error

ex) I00:L012,c=022.H=0.P=0.E=0

11.4. Scale

Send

I00A023[0x0a]

Receive

I00:L<data block size>,<data blocks><bcc>

<data blocks>

:== "c=023". ; cmd

"F="<data size>.<data> ; Scale Firmware Version

"A="<data>. ; Max2 Weight

"a="<data>. ; Max1 Weight

"I="<data>. ; Min Weight

"E="<data>. ; e2

"e="<data>. ; e1

"T="<data>. ; Max Tare Weight

"D="<data>. ; Weight decimal position

"P="<data>. ; Printed TPH Length (mm)

"L="<data>. ; Printed Label qty after changing (EA)

<bcc> :== XOR of <data blocks>

ex) I00:L043,c=023.F=07.V3.00.9A=15000.a=6000.I=40.E=5.e=2.
T=5998.D=3.P=110.L=9.<bcc>

11.5. A/D Status & Value

Send

I00A024[0x0a]

Receive

I00:L<data block size>,<data blocks><bcc>

<data block>

:== "c=024". ; cmd

"Z="<data>. ; Zero Flag. 0=flag off, 1=flag on

"S="<data>. ; Stable Flag

"O="<data>. ; Overload Flag

"R="<data>. ; Raw AD Value

<bcc> :== XOR of <data blocks>

ex) I00:L019,c=024.Z=0.S=1.O=0.R=3985.

11.6. Remote Key

Send

C16F079,<4byte key code>[0x0a]

<4byte key code > :== Feed : 16, Chess : 26B, Print : 15

11.7. Real Time Alarm

➢ **Scale → PC**

Send

I01:L<data block size>,<data blocks><bcc>

<data block>

:== "c=025". ; cmd

"M="<data size>.<data> ; Model Name.

"I="<data>. ; IP address

"H="<data>. ; Head up. 0=no error, 1=error

"P="<data>. ; Peel off. 0=no error, 1=error

"E="<data>. ; Paper end. 0=no error, 1=error

<bcc> :== XOR of <data blocks>

Model name : CL(CL5000/CL5500), C4(CL5200), C7(CL7200/CL7200-S)

ex) I01:L02C,c=025.M=02.C7I=0A0A0321.H=0.P=1.E=0.<bcc>

=> CL7200 IP10.10.3.33 Pee off

12. Other

You can read scale's date/time/current weight value

Indicial setting – you can cheek firmware Version, History...., etc.

12.1. Date / Time

- **Date/Time Setting**

```
W45F01,00L<data block size>:<data block>
<data block> := <date time><error>
<date time>
        Y=<year>.
        M=<month>.
        D=<day>.
        h=<hour>.
        m=<minute>.
        s=<second>.
```

Description

Year = 00 ~ 99 value ex) 2004yr = return"04".

month = 1~ 12 value

Day = 1~31 value

h = 0 ~ 23

m = 0 ~ 59

s = 0 ~ 59 set value.

Ex) 47min; return m=2f. value

- **Read**

```
Send
   R45F01,00<0x0a>
Receive
```

Same as Write data

12.2. System Password

- **System Password Change**

W45F02,00L<data block size>:<data block>
<data block> := "P="<string length>.<string>

- **Upload**

Send
R45F02,<id><0x0a>
Receive
Same as Write data

12.3. Scale Information

Scale Information contains following information
F/W Version, Data Structure Version
Weight Digit, Price Digit
KGLB Mode
Capa

- **Read**

Send
R45F03,<id><0x0a>
Receive
W45F03,00L<data block size>:<data block><bcc>
<data block>
F=<version><reversion>.
V=<version><reversion>.
w=<digit>.

p=<digit>.
K=<digit>. // 0: Kg, 1: LB
C = <digit>. // 0: 6, 1:15, 2: 30, 3: 60

12.4. Weight

Return current weight information

- **Upload**

Send
R45F04,00<0x0a>
Receive
W45F03,00L<data size>:<data><bcc>
<data> := "W="<weight>."P="<digit>.

12.5. History

- **Upload**

Send
R45F05,<history id><0x0a>

Receive
W45F03,00L<data size>:<data><bcc>

<history id> = "0" newly updated set time and info.
<history id> = "1" just before set time and info. You can have 0~4 values

12.6. Label(menu 1510)

Return global label format no(menu 1510)

- **Upload**

```
Send
   R45F06,00<0x0a>
Receive
   W45F06,00L<data size>:<data><bcc>
   <data> := "U="<2 byte label priority> ;Use global label
          := "a="<3 byte data>. ; PLU Label No.
          := "b="<3 byte data>. ; MISC Label No.
          := "c="<3 byte data>. ; Total Label No.
          := "R="<1 byte l data >. ; Reverse Total Label
          := "S="<1 byte l data >. ; Reverse Item Label
```

12.7. Barcode(menu 1520)

Return global barcode no(menu 1520)

- **Upload**

```
Send
   R45F07,00<0x0a>
Receive
   W45F07,00L<data size>:<data><bcc>
   <data> := "U="<2 byte Barcode priority> ;Use global Barcode
          := "a="<2 byte data>. ; PLU weight barcode   No.
          := "b="<2 byte data>. ; PLU count barcode No.
          := "c="<2 byte data>. ; PLU PCS barcode No.
          := "d="<2 byte data>. ; PLU fixed barcode No.
          := "e="<2 byte data>. ; PLU Misc.weight barcode No.
          := "f="<2 byte data>. ; PLU Misc.count barcode No.
```

:= “g=”<2 byte data>. ; PLU Misc.pcs barcode No.
:= “h=”<2 byte data>. ; Add up TTL barcode No.
:= “i=”<2 byte data>. ; Floating TTL barcode No.

12.8. Discount(menu 1530)

Return priority of discount

- **Upload**

Send
R45F08,00<0x0a>
Receive
W45F08,00L<data size>:<data><bcc>
<data> := “U=”<2 byte discount priority> ; 01 – PLU discount, 02 – Global discount

12.9. Tax(menu 1540)

Return global tax rate

- **Upload**

Send
R45F09,00<0x0a>
Receive
W45F09,00L<data size>:<data><bcc>
<data> := “U=”<2 byte tax priority> ;Use global tax
:= “a=”<3 byte data>. ; Global Tax No.

12.10. Parameter and Menu Settings

➢ **Download**

Send

W85A<param id>,0001L<data size>:<data><bcc>

<data> :== "D="<value length>.<value>

<bcc> :== XOR of <data>

Receive

Success

W85:O0001[0x0a][0x0a]

Error

W85:O0000[0x0a][0x0a] ; error

W85:O0002[0x0a][0x0a] ; Calibration Parameter

➢ **Upload**

Send

R85A<param id>[0x0a]

Receive

W85A<param id>,<value length>L<data size>:<data><bcc>

<data> :== "D="<value length>.<value>

<bcc> :== XOR of <data>

➢ **example**

- Parameter 547(0x223), Set 0(Download)

<Send>

```
57 38 35 41 30 32 32 33 2C 30 30 30 31 4C 30 30    W85A0223,0001L00
30 36 3A 44 3D 30 31 2E 30 66                       06:D=01.0f
```

<Receive>

```
5E 3D 30 31 2E 2A 3D 30 31 2E 24 3D 30 2E 26 3D    ^=01.*=01.$=0.&=
30 41 30 41 30 34 45 44 2E 40 3D 34 46 35 30 2E    0A0A04ED.@=4F50.
3F 3D 33 2E 57 38 35 3A 4F 30 30 30 31 0A 0A       ?=3.W85:O0001..
```

➢ **example**

- Parameter 502(0x1F6), Request setting value (Upload)

<Send>

```
52 38 35 41 30 31 46 36 0A                              R85A01F6.
```

<Receive>

```
57 38 35 41 30 31 46 36 2C 30 41 4C 30 30 32 44    W85A01F6,0AL002D
3A 5E 3D 30 31 2E 2A 3D 30 31 2E 24 3D 30 2E 26    :^=01.*=01.$=0.&
3D 41 43 31 30 31 30 43 36 2E 40 3D 34 46 35 30    =AC1010C6.@=4F50
2E 3F 3D 33 2E 44 3D 30 34 2E 30 34 31 39 55       .?=3.D=04.0419U
```

12.11. Reboot

저울을 리부팅한다.

Send

W89A0001,09L0001:.<bcc>

Receive

Success

W88:O0001[0x0a][0x0a]

ex) W89A0001,09L0001:..

12.12. Category Name (CL7200-S Only)

➢ **Download**

Send

W32F08,<id>L<data block size>:<data blocks><bcc>

<data block> :== P=<data size>.<data> ; Category name

:== B=<data>. ; Category key on/off

<bcc> :== XOR of <data blocks>

■ **example**

◆ Category 3 (Upload), Category key on

<Send>

57 33 32 46 30 38 2C 30 30 33 4C 30 30 31 32 3A 50 3D 30 39 2E 43 61 74 65 67 6F 72 79 33
42 3D 31 2E 29 W32F08,003L0012:P=09.Category3B=1.)

<Receive>

57 33 32 3A 4F 30 30 30 31 0A

0A

W32:00001..

➢ **Upload**

Send

R32F08,<id>[0x0a]

R33F08,<sequential no.>[0x0a] ; if specific id don't exist, scale search next id and transmit data

Receive

W32F08,<id>L<data block size>:<data blocks><bcc>

<data blocks> :== P=<data size>.<data> ; Category name

:== B=<data>. ; Category key on/off

<bcc> :== XOR of <data blocks>

■ **example**

◆ Category 3 (Upload), Category key on

<Send>

52 33 32 46 30 38 2C 30 30 30 31 0A

R32F08,0001.

<Receive>

57 33 32 46 30 38 2C 30 30 31 4C 30 30 33 36 3A 5E 3D 30 31 2E 2A 3D 30 31 2E 24 3D 30 2E
26 3D 30 41 30 41 33 43 35 42 2E 40 3D 30 30 30 30 2E 3F 3D 33 2E 50 3D 30 39 2E 43 61 74
65 67 6F 72 79 31 42 3D 31 2E 26

W32F08,001L0036:^=01.*=01.$=0.&=0A0A3C5B.@=0000.?=3.P=09.Category1B=1.&

12.13. Download Logo File (CL7200-S Only)

1. Download Logo File (Using FTP)
   - ■ Donwload Folder : /logo
   - ■ Filename(fix) : "logo.png"

※ FTP settings

PORT : 21

ID : cas

Password : cascl7200

13. PLU Field Control

Set allowance or not allow PLU Field.
Before using CL5000 set PLU item field.

- **Download (setting)**

Send

Setting
W36F01,<id>L<data size>:<data><bcc>
<data> := "P="<ptype number>."S="<ptype number>.

Receive

Apply	C<xx>F36,01<0x0a>

- **Upload (status)**

Send
R36A<ptype number><0x0a>
or
R36F<ptype number>,00<0x0a>

Read
R37A<series number><0x0a>

Continue reading

Receive

W36F01,01L<data size>:<data><bcc>

14. Internal Communication

12.1 Transaction Number

Call up new Ticket Number from Remote Server

i00F026,01

Return

I00F036,L<length>:<data><bcc>

<data>:=”T=<counter>.”

12.2 Master Connection Check

Check connection of master-slave

- **Slave → Master**

  i00F010,01L

- **Master → Slave(ACK)**

  i00F010,02L

15. Example

13.1. PLU

Ex1) Upload PLU

Uploading Department ID 1#, PLU #5.

Send : "R13F01000005,00<0x0a>"

Receive: No data

"W02A00000,00L0027:^=03.*=01.$=0.&=0A0A0321.@=4F50.N=0000.="

N=0000. Searching existence of information

If data exist;

"W02A00005,00L0198:^=03.*=01.$=0.&=0A0A0321.@=4F50.N=0002."

"F=02.4C,0004: F=04.4D,0001: F=0A.53,0006:BANANA

"F=1E.53,0000:F=1F.53,0000:F=09.57,0002:

"F=50.57,0002: F=37.57,0002: F=05.42,0001:

"F=64.4C,0004: F=0B.4C,0004:d F=0E.57,0002: F=06.4C,0004:?

"F=08.42,0001: F=0D.4C,0004: F=18.57,0002: F=17.57,0002:

"F=55.57,0002: F=10.4C,0004: F=11.42,0001: F=19.57,0002:

"F=5A.42,0001: F=47.57,0002: F=45.4C,0004: "

"F=" following each space data(binary, total length is 0x198)
Each data are displayed "F=02.4C,0004:..." means,
PLU Item <02>
= means PLU Number and this data is return as binary(Intel) format.

Ex2) Upload PLU by Sequential

Searching unknown PLUs information;

Send : "R02F01000005,00<0x0a>"

Receiving 5th PLU information more information is following Ex1)

Ex3) Delete PLU

Department ID : 1
PLU NO: delete 5

Send: "C43F13,01000005"

All PLU delete

Send: "C43A02<0x0a>"
Receive : "C003:O02"

Est. 1min. to delete all PLU(3000 PLUs)

13.2. Department

Ex1) Upload #8 Department

Send : "R20F01,008<0x0a>"

Receive: "W20F08,000L0024:^=03.*=01.$=0.&=0A0A0321.@=4F50.B=1"

discription : Scale ID = 3

Department = 1

IP = 10.10.3.33

Port = 20304.

NO data

Upload #1 Department

Send : "R20F01,008<0x0a>"

Receive: "W20F01,001L00031:^=03.*=01.

$=0.&=0A0A0321.@=4F50.D=07.DefaultK=01."

Description = "Default"

Speed Key ID = 1

Ex2) Department #2 delete

Send:

C01F21,010001<0x0a>

Return

C01:O0021<0x0a>

Error

W01:O0021,R01<0x0a>

13.3. Ingredient

Ex1) Upload #1 Ingredient
Send : "R20F01,008<0x0a>"
Receive: "W20F08,000L0024:^=03.*=01.$=0.&=0A0A0321.@=4F50.B=1"

Ex2) Download #1 Ingredient
discription : Ingredient Text
Send : "W30F01,0001L014:D=0F.Ingredient Text "
Receive: "W30:O0001"

Description part must not above 512Byte.
If so, cut a part 512Byte each to send following example.
Send : "W30F01,0001L014:X=000.D=200.<512 byte first Block>"
Send : "W30F01,0001L014:X=001.D=200.<512 byte second Block>"

When Ingredient is downloading (ID registering X=000) (Data saving X=001)

Ex3) Delete #1 Ingredient
Send : "C41F31,010001<0x0a>"
Receive: "C001:O31<0x0a>"

13.4. Barcode Format

Ex1) Upload #1 Barcode
Send : "R20F0008,0001"

Receive : If there are no Data
"W20F08,000L0024:^=03.*=01.$=0.&=0A0A0321.@=4F50.B=1"
Data is exist:
"W20F08,001L004C:^=03.*=01.$=0.&=0A0A0321.@=4F50.

D=0A.BARCODE/01F=0D.DDIIIIIVPPPPCN=001.T=01.c”

Description = “BARCODE/01”
Format =”DDIIIIIVPPPPC”
Barcode Format Number = 1
Type = 1 (UPC)

Ex2) Download #1 Barcode

Description : DownloadBar
Format Userdefine
Type = 2

Send : “W20F08,0001L02E:T=02.N=0000.”
“D=0B.DownloadBarF=0D.DDIIIIIVPPPPC%”
Receive: “W30:O0001”

Description part must not reach 512Byte.
512Byte so, cut a part 512Byte each to send following example.
Send : “W30F01,0001L014:X=000.D=200.<512 byte 의 첫번째 Block>”
Send : “W30F01,0001L014:X=001.D=200.<512 byte 의 두번째 Block>”

When Ingredient is downloading (ID registering X=000) (Data saving X=001)

Ex3) Delete #1 Barcode
Send : “C41F21,080001<0x0a>”

Receive: “C001:O21<0x0a>”
For Error massage
Receive : “W21:E99<0x0a>”

13.5. Discount

Ex1) Upload PLU 1, Dept 1 of Discount information

Send: “R09F0001,0001”

Receive : No data

“R09:E98<0x0a>”

Receive : Discount Setting value exist

“W09A0001,00L0072:^=03.*=01.$=0.&=0A0A0321.@=4F50.”

“a=01.b=1.c=1C97.d=64.e=C8.f=64.g=C8.h=000000.i=000000.”

“j=000000.k=000000.l=FE.S=00.c”

Ex2) Upload First Discount Information

Send: “R10F0001,00”

Receive : No data

“R10:E95<0x0a>”

Receive : Discount set value exist

“W09A0001,00L0072:^=03.*=01.$=0.&=0A0A0321.@=4F50.”

“a=01.b=1.c=1C97.d=64.e=C8.f=64.g=C8.h=000000.i=000000.”

“j=000000.k=000000.l=FE.S=00.c”

Ex3) Delete Private Discount

Dept #2 PLU Number #1 Discount info delete

Send : “C41F09,02000001”

Return : “ C001:O09”

Ex4) Delete All Discount

Send : “C01A10<0x0a>”

Receive : delete all

“C001:O10<0x0a>

Ex5) Download Discount

data : deptno =1

pluno =2

Discount Type = 0x1c2b

First Target = 100

First Discount Value = 1000

Second Target = 200

Second Discount Value = 2000

Send : “W09F01,0000L028:a=01.b=2.c=1C2B.d=064.f=3E8.e=0C8.g=7D0.6”

Receive : register normally

“W09:O01”

13.6. Report

Ex1) Upload Department 1, PLU 1

Send:

"R09F0001,0001<0x0a>"

Receive :

"W09A0001,00L0072:^=03.*=01.$=0.&=0A0A0321.@=4F50."

"a=01.b=1.c=1C97.d=64.e=C8.f=64.g=C8.h=000000.i=000000."

"j=000000.k=000000.l=FE.S=00.c"

Ex2) Delete

13.7. Label and Image

```
firstsend()
{
	LABELHEADER lh;
	int p,p_start;
	/*
	    filename <- GetFilename();
	*/
	fp=fopen(filename,"rb");
	fread(&lh,1,sizeof(lh),fp);
	fclose(fp);
	p=18;
	p_start=p;

	// "\0" include text
	sprintf(&msg[p],"S=%04X.",lh.size_label);
	p+=strlen(&msg[p]);
	sprintf(&msg[p],"W=%04X.",lh.width);
	p+=strlen(&msg[p]);
	sprintf(&msg[p],"H=%04X.",lh.height);
	p+=strlen(&msg[p]);
	sprintf(&msg[p],"Z=%s.",lh.name);
	p+=strlen(&msg[p]);
	data_size=strlen(&msg[p_start]);
	bcc = get_bcc(&msg[p_start],data_size);
	msg[p++]=bcc;
	msg[p]=0;
	data_send_leng = p;
	sprintf(msg,"W06F%04X,%02X%02XL%03X",lh.id,m_labelform,1,data_size);
	msg[p_start-1]=':';
	SendData(msg,data_send_leng);
}
```

// Return : Rxx:O043

```
sendlabel()
{
}
```

15. Reference

Ref 1. Use of Terms

Speed Key Set Number

You can set 5 types of Speed key, this we call “SET NUMBER.” You can set for each department

Bridge

Transparent bridge: a packet-forwarding device that gets it forwarding instructions from the Destination Address Field in the MAC header. Transparent bridges learn about the location of nodes on a network by examining the Source Address Field of packets sent on the network. Transparent bridges are currently used in both the Token-Ring and Ethernet environments. End nodes need not be aware that transparent bridges exist on the network.

Router

A system responsible for making decisions about which of several paths network traffic will take, and for keeping track of routing information which is being passed along a network be one of several different possible protocols. To do this a router uses a routing protocol to gain information about the network and uses algorithms to choose the best router based on several criteria known as route metrics. In OSI terminology, a router is a Network Layer intermediate system.

Hub

The center of a star topology network or cabling system. The term Ethernet hub typically refers to a shared-media hub. Supports shared Ethernet in a "star" topology over Category 5 twisted-pair wire terminated by RJ-45 data jacks.

Repeater

A repeater connects two segments of your network cable. It retimes and regenerates the signals to proper amplitudes and sends them to the other segments. When talking about, Ethernet topology, you are probably talking about using a hub as a repeater. Repeaters require a small amount of time to regenerate the signal. This can cause a propagation delay which can affect network communication when there are several repeaters in a row. Many network architectures limit the number of repeaters that can be used in a row. Repeaters work only at the physical layer of the OSI network model.

Gateway

A gateway can translate information between different network data formats or network architectures. It can translate TCP/IP to AppleTalk so computers supporting TCP/IP can communicate with Apple brand computers. Most gateways operate at the application layer, but can operate at the network or session layer of the OSI model. Gateways will start at the lower level and strip information until it gets to the required level and repackage the information and work its way back toward the hardware layer of the OSI model.

TCP/IP

The part of the network that does the job of transporting and managing the data across the network is called TCP/IP which stands for Transmission Control Protocol (TCP) and Internet Protocol (IP). There are other alternative mechanisms for managing network traffic, but most, such as IPX/SPX for Netware, will not be described here in much detail. The IP layer requires a 4 (IPv4) or 6 (IPv6) byte address to be assigned to each network interface card on each computer. This can be done automatically using network software such as dynamic host configuration protocol (DHCP) or by manually entering static addresses into the computer.

UDP

User Datagram Protocol (UDP) supports the network at the transport layer. User Datagram Protocol (UDP) is an unreliable connection-less protocol and is defined by RFC 768 and 1122. It is a datagram service. There is no guarantee that the data will reach its destination. UDP is meant to provide service with very little transmission overhead. It adds very little to IP data packets except for some error checking and port direction (Remember, UDP encapsulates IP packets).

DHCP

This protocol is used to assign IP addresses to hosts or workstations on the network. Usually a DHCP server on the network performs this function. Basically it "leases" out address for specific times to the various hosts. If a host does not use a given address for some period of time, that IP address can then be assigned to another machine by the DHCP server. When assignments are made or changed, the DHCP server must update the information in the DNS server.

Access Point

Wireless access points (APs or WAPs) are specially configured nodes on wireless local area networks (WLANs). Access points act as a central transmitter and receiver of WLAN radio signals.

Access points used in home or small business networks are generally small, dedicated hardware devices featuring a built-in network adapter, antenna, and radio transmitter. Access points support Wi-Fi wireless communication standards.

Revision

No.	Date	Chapter		
1	2006.08.31	4.3.1	Ingredient download Insert error code “0x90”	
2	2008.09.25	7.2	Add “7.2 Real Time Report”	J.K.Chung