Space/Comma Delimited ASCII Data Transfer

Table Of Contents

First Release . . . . . September 01, 1999
Latest Release . . . . . December 12, 1999

Space/Comma Delimited ASCII Data Transfer

1.0 General

The CPP responds to a set of commands to return time and date, stored data, status, calibration results and other stored parameters in a comma delimited ASCII string. This format allows collected data to be readily submitted into many data bases including an Excel spread sheet. Data and commands can also be sent to the CPP in a similar format. The command template is presented below. For clarity, the template is shown with commas as delimiters. These can also be spaces. The CPP checks the first character after the D_C direction character for either a comma or a space and uses this character as the delimiter. Whatever is detected is used as the delimiter in the returned transmission. The line feed (lf) may or may not be included in the transmission.

D_C,III,VVV,NNN,[field,]CC<crlf>

1.1 Definition Of String Template

Presented below is a definition of the command string template.

D_C,III,VVV,NNN,[field,]CC<crlf>

Where;

D_C is a greater than sign (>) or direction code, indicating that the string originated at the central,

III is a three byte remote ID code - 000 - 999.
(000 -009 reserved for global commands),
CPP defaults are 010.

VVV is a numbered command code indicating what is to be done, the range is from 000 to FFF,

NNN has various uses depending on the command,

[field,] augments the VVV command code for given commands. The brackets are for clarity only and **should not be included** in the string.

CC is a checksum of the command. Sum all eight bit bytes (including delimiters and direction characters) into an initially cleared 8 bit register, take the two's complement, send in hexadecimal, most significant nibble, first. The checksum should be followed with a carriage return. Over comm port #4, the command may be invoked by sending a carriage return after the field instead of the checksum characters.

1.2 CPP Response

For a remote to respond it must properly detect the direction character, a comma or space delimiter, it's station ID, a command that it understands, and either a correct checksum and carriage return or a carriage return. The return string is very similar to that given above, but varies somewhat depending on what is being returned. The CPP always returns a checksum and a carriage return and a line feed (crlf). The CPP also returns an end of transmission. This string is defined in each section

2.0 Send Commands To CPP

Data or commands can be sent from the central to the remote using a similar format. The template format changes slightly when data is being sent to the remote. The template is given below is:

D_C,III,VVV,NNN,[field],CC<crlf>

Where;

D_C is direction code,

III is three digit remote ID code

VVV = is a command code (defined in Table 1.0 at the end of this tech note),
500 = set time & date mmddyy or (time & date may have separators as defined below),
501 = set time &date ddmmyy,

NNN = number of bytes in [field], including delimiters,

[field] = The brackets are for clarity only, and should not be included in the actual string.

CC = checksum as defined before.

2.1 Set Time & Date In CPP

The time and date can be set in the CPP with the following command:

D_C,III,500,014,[mmddyy,hhmmss,]CC<cr>

Where;

The 500 indicates that the CPP is to set it's time and date to that in the field. The 014 is the number of bytes in the field. The brackets in the above are for clarity only, and should not be included in the actual transmission. This command can also be sent without the checksum, by following the field with a carriage return. The month, day and year can be separated with slashes or dashes, and the hour, minute and seconds can be separated with colons. Note, this affects the count - NNN.

A command code of 500 indicates that the date is being sent as mmddyy. A command code of 501 indicates that the date is being sent as ddmmyy.

If the CPP is set up as mm/dd/yy and receives a 501 command, it will parse and store the date in an mm/dd/yy format. The converse is true as well.

If the time/date download is accepted and executed by the CPP, the CPP responds with an end of message string.

D_R,III,500,N,W,CC<crlf>

Where;

N equal zero = time date was accepted
N equal 0ne = error in time or date field, not accepted

2.2 Time & Date Return
The following command requests time and date from a CPP.

D_C,III,012,000,CC<cr>

Time and date are returned in the following format. In the return transmission NNN is set to 020 (twenty).

D_R,III,012,020,[Y,mm/dd/yy,hh:mm:ss,]CC<crlf>

Where;
Y = time date format.
mm/dd/yy = month, day, year of data string being returned, if Y above is a Y, then mm/dd/yy format returned. If Y is an E, then dd/mm/yy format is returned

The CPP does not return the brackets in the field. The end of message string is as follows;

D_R,III,012,0,W,CC<crlf>

Where;
W is an EOT character (04 in hexadecimal).

2.3 Return I/O Settings
The following command commands the CPP to return a snapshot of the current I/O settings.

D_C,III,013,000,CC<cr>

The CPP responds with the following string.

D_R,III,013,005,[0000000000,0000000000,]CC<cr>

Where;
The returned NNN defines the number of I/O bits in the CPP. The 005 shown above indicates five groups of eight bits each, or 40 digital input and 40 digital output bits.

[field] the output bits are presented first and comma delimited between the output and input bits. The I/O bits are converted to hexidecimal and transmitted.

I/O bits are returned in the following format;

8 5 4 1 16 13 9
(0000 0000) (0000 0000)
(1110 1000) = E8

Therefore, if bits 8,7,6 and 4 are set the CPP would return
E8 for these 8 bit settings.

The CPP does not return brackets in the field. The end of message string is as follows;

D_R,III,013,N,W,CC<crlf>

Where;

N = 0

W is an EOT character (04 in hexadecimal).

2.4 Set/Reset Output Bits

The following command allows the output bit settings of the CPP to be set and reset over the comm port.

D_C,III,014,004,[S01,R03,S14,M04,]CC<cr>

The S01 indicates that output bit #01 is to be set. The R03 indicates that output bit #03 is to be reset. The M04 indicates that bit 04 is to be set for 1 second, and then reset. This command is useful when a bit has been set up to start a sequence. The bits can be set and reset in any order in the field. Up to nine bits can be set or reset in a single transmission. In the example above, the 004 in the N field, indicates that four bit settings are being sent. As always, the brackets, [ ] are for clarity only.

It should be noted, a bit set in this fashion remains set until it is reset either in this fashion or by an operator using the CPP front panel or the BS and BR (bit set and bit reset) inputs.

The CPP will return an end of string message with the N field set to zero if it can perform all bit settings. If a bit number out of range is received, the CPP will return an end of string message with the N field set to the bit number received that is out of range, *e.g.*, (B41), and none of the bit settings/resettings are executed. If a character other than an S, an R, or an M is received the N field is returned E0X, where the X is the character received.

3.0 CPP Data Return

The CPP's contain more memory than is required for storing set up or configuration parameters. This additional memory is referenced as excess internal memory. If a CFM card (Compact Flash) is installed in the CPP, then all read and write commands are directed to the CFM memory. If a CFM is not installed, then all read and write commands are directed to the internal excess memory. This internal excess memory is used as an internal CFM when a card is not plugged into the front panel. The CPP also contains a section of active memory where the most recently collected averages are stored. A certain amount of preliminary, interim and final averages are stored in the active memory area. The user can set up which of these averages are to be stored in either the CFM or the excess internal memory. If the CPP receives a request to return data, this data is retrieved from the long term data storage and returned. If this average is not being stored in the long term memory, then the CPP retrieves and returns the latest averages from the active memory area. To a central polling computer, it is irrelevant from which memory data is being returned.

Data stored either in the excess internal memory, or in the CFM can be retrieved in a comma delimited ASCII data transfer. The CPP supports two types of cartridge data retrieval. The first starts at the most current record recorded on the cartridge, and sends a requested number of records going back in time. The second uses a starting time and date, and an ending time and date. The requested records recorded during that time frame are returned. Section 3.1, below, describes the request commands, section 3.2 describes the CPP response

using the number of records to return format, and section 3.7 describes the command structure for the start and ending times format.

When a CFM is installed in the CPP, the CPP records a header on the CFM. This header consists of the station name, station ID, and the time and date that the CFM was installed. A similar header is available from the internal memory. A command for retrieving this header information is given in section 3.8 below.

3.1 Data Request Command Structure

The request command is presented below. The CPP also supports a command that returns all calibration results recorded during the final time period. For hourly averages this is the last 24 hours, for 30 minute final averages this is the last 12 hours, and for 15 minute final averages this is the last 6 hours. This is described in section 3.4.2.

D_C,III,$V_1V_2V_3$,NNN,[Y,#xxxx,]CC<crlf> >,010,F80,000,Y,#0004,<cr>

Where:

$V_1V_2V_3$ (the subscripts are to provide clarity) is a command code for what information to return. The 1st V_1 is an E or F indicating a read cartridge command. The remaining V's are bit encoded to determine what information to return. The E is used to return an instantaneous sample reading, reference section 3.9. An F00 command instructs the CPP to return the recorded header as described in section 3.8 below.

V_1	$V_2$1234	$V_3$1234	Return
F	1xxx	xxxx	preliminary averages
F	x1xx	xxxx	interim averages
F	xx1x	xxxx	final averages
F	xxx1	xxxx	alarms
F	xxxx	1xxx	calibrations
F	xxxx	x1xx	max/mins
F	xxxx	xx1x	digital I/O
F	xxxx	xxx1	events (includes PF's & time changes)

The bits given above may be combined in any fashion. As some examples;

FA0	return preliminary & final averages
F28	return final averages & calibration results
FE8	return all averages & calibration results
FFF	return all parameters

Returning data from a cartridge can result in very large blocks of data being transferred. The CPP sends a checksum with each record of data to ensure the data integrity. Over telephone lines, it is not uncommon to occasionally detect errors in the transmission. In such cases, the line of data needs to be transmitted again. The NNN field is used to select this format. With the NNN field set to 0xx, the CPP starts returning cartridge data and continues sending data packets until the ending criteria is detected and then it sends an end of message string. If the NNN field is set to a 1xx, the CPP pauses after each record of data is sent, and awaits an acknowledge from the central. If a >,OK,CC,<cr>, or a >,OK,<cr> is received, the CPP proceeds to send the next line of data. If one of the above is not properly received within 4 seconds, the CPP retransmits the same line again. If an acknowledge is not received twice in a row, the CPP sends an end of message string and aborts the transmission.

If the central responds with a command, but not the OK presented above, (*e.g.,* >,NAK,<cr>) the CPP will resend the same string. The CPP will repeat sending the same string for 6 times, after which it will send a end of string message and abort the transmission.

The xx in the NNN field is a reserved function that, if implemented, can be used to direct the CPP how many channels of data to return. If xx is set to 00, then the CPP returns all channels enabled in the unit. If xx is greater than the number of channels enabled in the CPP, the CPP only returns the number of channels enabled. The CPP sets the xx to 00.

The starting criteria, Y or E should be followed by a pound sign (#) and then a four digit number indicating how many entries are to be returned. The pound sign is used to differentiate this command from the command based on time presented in section 3.7. An entry is a returned field. Returned data would be an entry, returned calibrations would be an entry, returned alarms would be an entry, and so forth. If the xxxx number is ALL, then the CPP sends all of the requested parameters contained in the cartridge. The Y indicates that the date is to be returned mm/dd/yy. If the Y is replaced with an E, then date is returned dd/mm/yy.

The brackets are for clarity only and should not be included in the string.

3.2 Data Return Format

Data is returned from a remote in the following format. Since more than one set of averages can be requested with one command, each set of averages will be sent as follows. After all have been sent the CPP will send an end of transmission message to apprise the central that it has sent all data requested.

D_R,III,F20,020,[Y,mm/dd/yy,hh:mm:ss,$S_1S_2S_3S_4$,±$D_1D_2D_3D_4$E±XX, . . . , . . ,]CC<crlf>

Where;

D_R is a less than sign (<) indicating that the transmission initiated at a remote,

III is the remote's three digit ID,

VVV indicates the command that is being returned,

NNN indicates the number of channels being returned in the field,
with delimiters 15 bytes per channel,

NNN	Indication
0xx =	starting with channel 01, (01-20)
1xx =	starting with channel 21, (21-40)
xx =	number of channels being returned
Y =	time date format.

mm/dd/yy = month, day, year of data string being returned, if Y above is Y, then mm/dd/yy format returned. If Y is an E, then dd/mm/yy format is returned.

hhmmss = hour, minute, second of data being returned,

$S_1S_2S_3S_4$ = data status, (defined below)

± = sign of data point,

D_1 - D_4 = data value, (**a decimal point is assumed at the end of D_4,)**

E = exponent (powers of ten),

± = sign of exponent,

X X = exponent (00 through ±99).

CC = checksum as defined above, the checksum is returned even if the transmission was initiated without one.

The CPP does not return brackets in the field. The end of message string is as follows:

D_R,III,VVV,N,W,CC<crlf>

Where;
VVV is command that was received
N is error code, 0 = no error
W is an EOT character (04 in hexadecimal).

3.3 Status $S_1S_2S_3S_4$

As indicated above, each data point returned has 16 bits of status information returned with it. The status is delimiter separated from the actual data point for ease in parsing. The two byte (16 bit) status information stored with each data point is returned as four ASCII characters. As an example, if the two status bytes were C1 4A, they would be transmitted as;

ASCII	Hex	Decimal
C	43	067
1	31	049
4	34	052
A	41	065

3.4 Return Calibration

The CPP responds to two separate commands and returns calibration data in two different ways. The first, presented in section 3.4.1 follows the other data retrieval protocol. The second, presented in section 3.4.2, responds to a single command and returns all calibrations conducted during a period.

3.4.1 Standard Protocol

The CPP returns calibration data in the following format;

D_R,III,F08,NNN,[nnnnnn,Y,mm/dd/yy,hh:mm:ss,Y,mm/dd/yy,hh:mm:ss,S,V_1,E_1,T,Y,C_1,C_2,]CC<crlf>

Where;
NNN = CPP channel for which calibration data is being returned,

nnnnnn	=	channel name
Y	=	defines date format (E)
mm/dd/yy	=	start date, (dd/mm/yy if E received),
hh:mm:ss	=	start time,
Y	=	defines date format (E)
mm/dd/yy	=	stop time, (dd/mm/yy if E received),
hh:mm:ss	=	stop time,
S	=	0= zero, 1= span #1, 2= span#2, 3= span#3, 4= span#4, etc
V_1	=	Cal value ($\pm D_1D_2D_3D_4E\pm XX$ format),
E_1	=	expected value, ($\pm D_1D_2D_3D_4E\pm XX$ format)
T	=	type, A= auto, M= manual, I= internal, E= external, O=aborted,
Y	=	Y= cals corrected, N= not corrected,
C_1	=	if corrected = offset, else = +0000E+00
C_2	=	if corrected = slope, else = +1000E -03
CC	=	checksum.

3.4.2 Second Calibration Protocol

The CPP responds to a single command and returns all calibration results stored for the period. The command string is similar to that presented previously.

D_C,III,$V_1V_2V_3$,NNN,[Y,#Cxx,]CC<crlf>

Where;

V_1 = F
V_2 = 0
V_3 = 8
NNN = 001

The command F08 commands the CPP to return all stored calibrations. The N field is used to select the period for which to return the calibration results. The CPP stores 7 periods of final data in internal memory. Associated with each period is any calibration results that occurred during that period. If the final average being formed is an hourly average, then the 8 periods are 7 days. If 30 minute averages are being formed then 4 days of averages are stored, as 0000 until 1730 and 1800 until 2330. If fifteen minute averages are being formed then 2 days of data is stored with the days being divided into 4 parts, 0000 until 0545, 0600 until 1145, 1200 until 1745, and 1800 until 2345.

The N field is used to select the period from which to return calibration data. A one or zero selects the first or most recent period. A two goes back one period and so forth. Anything greater than an 7 is set to the 7^{th} period. Data is returned in the same format as presented in section 3.4.1 above. The C following the # sign in the field is what selects the second calibration protocol. In all command receptions, the received Y or E selects the format in which the CPP will return data.

If set up to be recorded, as calibrations occur the results are spooled into the CFM. In this fashion all calibrations are stored. In the internal memory, only the results of the last calibration are retained in each time frame. If more than one calibration is conducted in a time frame, the latest results overwrite the oldest. When using the second protocol only the last calibration results can be obtained.

The CPP returns an end of message after all stored calibrations have been returned.

3.5 Return Events

Power fails, time changes and I/O settings are all returned as events. The format for each is presented below.

Power fail/Time change

D_R,III,F01,NNN,[field,]CC<crlf>

Where;

[field,] = [X,Y,mm/dd/yy,hh:mm:ss,Y,mm/dd/yy,hh:mm:ss,]

X = F=Power fail, T=Time change, O=Off line, B=Brownout

The Y would be an E for dd/mm/yy. The first time/date is the from time and the second is the until (to) time.

I/O Settings

D_R,III,F01,NNN,[field,]CC<crlf>

Where;

NNN = # of groups of 8 I/O bits

[field,] = [E,Y,mm/dd/yy,hh:mm:ss,0000000000,0000000000,]

3.6 Return Alarms

Alarms are returned as shown below.

D_R,III,F10,NNN,[nnnnnn,Y,mm/dd/yy,hh:mm:ss,ED_1D_2,A_1,$A_2A_3A_4$,ED_1D_2,]CC<crlf>

Where:

NNN = channel number

nnnnnn = channel name

Y = defines date format

ED_1D_2 = data value

A_1 = S, P,I,F

A_2 = H = high, L = low

A_3 = A = alarm, W = warning

A_4 = E = entering, L = leaving

ED_1D_2 = set point

3.7 Return Data Using Start/Stop Time Criteria

The request command is presented below. All other commands respond immediately. This command, depending on how far back in time the starting time is from the current time, and the number of records recorded on the cartridge, could take several seconds for the CPP to locate the starting time and start responding.

D_C,III,$V_1V_2V_3$,NNN,[field,]CC<crlf>

Where:

$V_1V_2V_3$ and NNN are as presented in sections above.

The [field] in the above transmission is used to determine which, and how much data is returned from the cartridge. The field consists of two attributes, A_1 (start), and A_2 (end). The A_1 attribute defines the starting criteria based on time, and is input in the following format; Y,mmddyy,hhmmss, or E,ddmmyy,hhmmss,. The A_2 attribute defines the stop sending data criteria, again based on time, and is input in the following format; mmddyy,hhmmss, or ddmmyy,hhmmss,.

The CPP checks the starting time against the last time recorded on the cartridge. If the starting time is not a time before the current time, the CPP sends a can not find starting criteria message. The CPP then checks the ending time against the last time recorded on the cartridge. If the ending time is before the cartridge time, then the cartridge is searched, going back in time, until either the ending time or a time before the ending time is found. If the ending time is a time after the last time recorded on the cartridge, the CPP starts at the current setting.

Records are output from the ending time, going back in time until a time before the starting time is detected.

An example of a field transmission is presented below. The time and date values should not be separated with standard symbols. Both digits must be sent even if the first is zero, such as 08 for August. To maintain ease of use and consistency, the {Y,} in the string presented below is optional. It may, but need not be included in the string. As with the brackets.[ ], the braces,{ } are for clarity only, and **must not be included** in the string.

[Y,010104,000000,{Y,}020104,000000,]
|
Optional

This field commands the CPP to return data for the entire month of January 2004. The $V_1V_2V_3$ command, discussed in section 3.1 above, defines what information the CPP is to return.

3.8 Return Header Information

The CFM header information is requested with the following command.

D_C,III,$V_1V_2V_3$,NNN,CC<crlf>

Where;

$V_1V_2V_3$ = F00

The header information is returned as follows.

D_R,III,$V_1V_2V_3$,NNN,[Y,mm/dd/yy,hh:mm:ss,SSSSSSSSSSSSSSS,III,]CC<crlf>

Where;

Y = time/date CFM was installed
time/date returned in format set into CPP
S…S = 16 bytes of station name
III = station ID

3.9 Return Instantaneous Reading

This command results in the CPP returning an instantaneous reading for the requested number of channels. Channels set up to receive data from an A/D channel will update every ten seconds. Channels set up as computed channels or rolling averages will update every minute. Channels polling instruments will update at the instrument poll rate.

The command to return the instantaneous reading is given below. An I should follow the pound sign in the field.

D_C,III,$V_1V_2V_3$,$N_1N_2N_3$,[Y,#Ixx,]CC<crlf>

Where;

V_1 = E
V_2 = 0
V_3 = 1
$N_1N_2N_3$ =

The N_1 functions as presented in section 3.1, and selects either an acknowledge or no acknowledge data transfer. If the $N_1N_2N_3$ field is set to a 1xx, the CPP pauses after each record of data is sent, and awaits an acknowledge from the central. If a >,OK,CC,<cr>, or a >,OK,<cr> is received, the CPP proceeds to send the next line of data. If one of the above is not properly received within 4 seconds, the CPP retransmits the same line again. If an acknowledge is not received twice in a row, the CPP sends an end of message string and aborts the transmission.

The N_2N_3 in the NNN field is reserved, and if implemented, can be used to direct the CPP how many channels of data to return. If N_2N_3 is set to 00, then the CPP returns all channels enabled in the unit. If N_2N_3 is greater than the number of channels enabled in the CPP, the CPP only returns the number of channels enabled. The CPP sets the N_2N_3 to 00.

Data is returned from a remote in the following format. After all data has been sent the CPP will send an end of transmission message to apprise the central that it has sent all data requested.

D_R,III,E01,020,[Y,mm/dd/yy,hh:mm:ss,$S_1S_2S_3S_4$,±$D_1D_2D_3D_4$E±XX, . . . , . . ,]CC<crlf>

Where;

D_R is a less than sign (<) indicating that the transmission initiated at a remote,

III is the remote's three digit ID,

VVV indicates the command that is being returned,

NNN indicates the number of channels being returned in the field,
with delimiters 15 bytes per channel,

NNN	Indication
0xx =	starting with channel 01, (01-20)
1xx =	starting with channel 21, (21-40)
xx =	number of channels being returned
Y =	time date format.

mm/dd/yy = month, day, year of data string being returned, if Y above is Y, then mm/dd/yy format returned. If Y is an E, then dd/mm/yy format is returned.

hhmmss = hour, minute, second of data being returned,

$S_1S_2S_3S_4$ = data status, (defined below)

± = sign of data point,

D_1 - D_4 = data value, (**a decimal point is assumed at the end of D_4,)**

E = exponent (powers of ten),

± = sign of exponent,

X X = exponent (00 through ±99).

CC = checksum as defined above, the checksum is returned even if the transmission was initiated without one.

The CPP does not return brackets in the field. The end of message string is as follows:

D_R,III,VVV,N,W,CC<crlf>

Where;

VVV is command that was received
N is error code, 0 = no error
W is an EOT character (04 in hexadecimal).

3.9 End Of Message

Upon completing the requested transfer the CPP sends the end of message string;

D_R,III,VVV,N,W,CC<cr>

Where;

VVV is the command received, and W is an EOT character (04 in hexadecimal).
N is set to zero if the CPP could honor the request. If N is not equal to zero, then some condition existed, as presented below.

N	**Condition**
0	no error condition
1	could not find starting criteria
2	could not find ending criteria
3	end of data detected
4	can not find data
5	checksum error found in some record
6	too many checksum errors
7	no acknowledge twice in a row or resend 6 times
8	error in number to return
9	CF removed
A	Data request too far back
B	CFM error in response

4.0 Configuration Upload/Download

The configuration upload/download is a subset of these commands, and follows the guidelines presented above. However, it is sufficiently lengthy, and very seldom of interest to anyone but H2NS personnel, and is therefore, covered in a separate tech note. Please request tech note TN21.

5.0 CPP Error Indications

The CPP keeps an internal register that can be very useful when developing an interface using this protocol. Obviously the command string and checksum need to be correct before the CPP can respond. Should the CPP not respond, it loads this internal register with a code. This code can be reviewed with the M keyin. The register name is ASCIEC, and its absolute memory location can be found with the M keyin. Entering an M[ASCIEC Addresss- 9C62H, e.g.] results in the CPP printing the contents of this register. The brackets are for clarity only an are not to be included in the key in. The various error conditions are given below in Table 1.0. Reported error conditions could also be the result of a missing delimiter.

Table 1.0
CPP Interface Error Conditions

Code	Indication
1	Not our address
2	Not defined
3	V field in error
4	N field in error
5	Unknown command
6	If B, then in error
7	If Z, then in error
8	If checksum, then in error
9	Error in Bxxx
A	Error in Zxxx
B	Error in final request
C	Error in interim request
D	Error in preliminary request
E	Error in field

6.0 Table of Commands

Presented below in Table 2.0 is a definition of commands recognized by the CPP. If additional definition is needed to properly define the field, it is presented in above sections.

Table 2.0
Commands recognized By The CPP

Command #	Definition
000	Reserved for global commands
↓	"
009	Reserved for global commands
012	Return time & date
013	Return I/O settings
014	Set/Reset output bits
017	Return final averages
018	Return interim averages
019	Return preliminary averages
030	Return calibration results
040	Return present alarm settings
500	Set time & date in mmddyy,hhmmss, format
501	Set time & date in ddmmyy,hhmmss, format
C00	Configuration upload/download
↓	"
CFF	Configuration upload/download
E00	Reserved for cartridge
↓	"
EFF	Reserved for cartridge
F00	Read header information
F01	Read cartridge
↓	"
FFF	Read cartridge