

Data Hog Plus-G

Instruction Manual

Version : 1.04

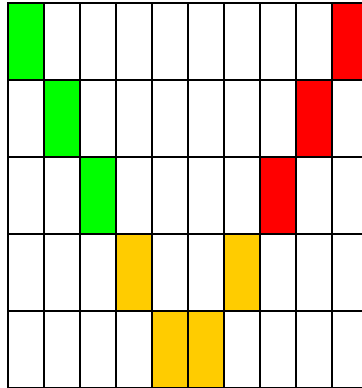
Date : 08/31/2020

Data Hog Plus: High speed Serial port to SD Card Adapter, Serial Data Logger with 1ms Accuracy, Clock/Calendar, RS232 Port Control, Upload/Download Files and Firmware, and customizable serial programming system.

PART 1 - LED Display:

The Data Hog Plus (DH+) indicates its current status with different LED displays as follows:

1. **Power on/Initialization** - Two LEDs travel from the left and right towards the center, and then back again. This shows that the unit is powering on and initializing.



2. SD Card Status

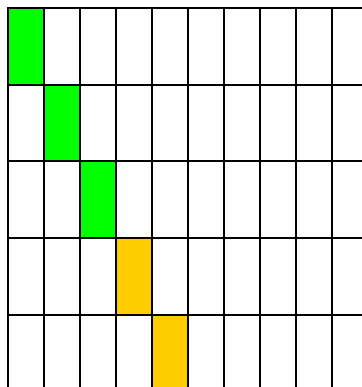
a. SD CARD INSTALLED

LED's will display from left to right (on a 1-10 scale) how much memory is used. For example, if 63% of the memory on the card is used to first 6 LED's will blink:



b. SD NOT INSTALLED

A single LED will blink rapidly from left to right again until a SD card is plugged in. Once SD is recognized, LEDs will show memory used and continue to step 3.



(and so on to 10th LED then back again)

3. **Serial Cable Not Connected** - The middle 4 LED's will blink while waiting for a serial cable to be plugged in:



4. Connected, Step 1:

- a. The Data Hog Plus by default will wait to receive a Ctrl+E (0x05) at 19200bps from the connected device. While this process is happening the 4th LED will blink:



- b. Once the Ctrl+E is received, it will switch to command mode indicated by the outside 2 LED's staying ON, with the 4th LED blinking.



NOTE: The Data Hog Plus can be configured to operate in many different ways when a cable is plugged in. It can automatically upload a file, send and receive custom strings and characters, and do almost anything else under your control. Refer to the “Serial Protocol” and “Batch File” sections for more information. By default, it simply waits for commands to be sent to it via the serial port.

5. **Connected, Step 2:** If passwords have been enabled, the 5th LED will start blinking. Send the password (with “PWD=<password>” command) to continue:



6. **Connected, Step 3:** If the DH+ has been configured to start an upload or download, or it receives a command to start an upload or download, the 6th LED will start blinking until the download or upload starts:



Once the download or upload starts, it switches to the 7th LED blinking and LED's 3, 4, & 5 showing a changing graph of each block sent or received:



When the file is done downloading or uploading, it will switch back to the 6th LED blinking and wait for the next command.

7. **Connected, Step 4:** If the DH+ has been configured to start an upload or download immediately on power up, then at the end of that process it will pause with no LED's blinking and show the amount of memory left on the SD card like this:



(This shows ~90% of the memory used – No LED's are blinking!)

This display of memory used is stopped with another command being received (or receiving a Ctrl+E).

PART 2 – Battery, SD Card, & Serial Port

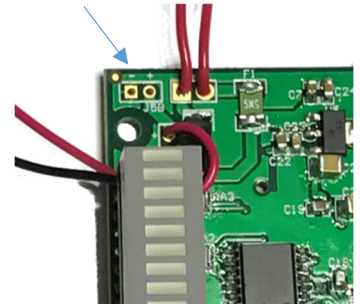
The Data Hog Plus (DH+) will operate from an input voltage of around 4V (5V or higher recommended) to as high as 16V DC. It draws ~8mA to as much as 70mA when all the LED's are lit and the serial port is in use. To make the unit portable and easy to use, a small 9V battery compartment is located at the top end of the case. Simply open the port and install a 9V battery to use the DH+.

Depending on what the unit is being used for, a typical 9V battery will last from 1 to 2 days (24 – 48 hours) of continuous use. Many users only turn on the unit when it is needed, so the DH+ lasts much longer under typical circumstances.

There are several different ways to power the DH+ other than using a standard 9V alkaline battery:

- 1) Connect a “9V Battery Adapter” such as the “Pinpoint 9-Volt Battery Universal AC Adapter Kit” (available on Amazon and other retailers). This simple and inexpensive device plugs into a wall socket and outputs 9V to a connector that plugs right into the 9V connector already on the DH+.
- 2) Use a higher capacity 9V battery such as the Energizer L522BP2 9V Lithium battery. This battery gives about 40% longer run time than a standard 9V battery. Another alternative is the 1200mAh Lithium Battery from PKCELL (Li-SOCl₂) which will last twice as long as a conventional 9V battery.
- 3) For longer term use, the DH+ also has secondary power input pads on the circuit board inside the case. This is located as shown in the image on the right.

Simply solder wires to the board (+ and -) and connect the other end to a 4V to 16V source. Make sure you connect the pad labeled “+” to positive voltage and the pad labeled “-” to ground.



In all cases using the **LEDS**, **SLEEP**, and **IDLE** commands can help lower the power draw of the DH+ for those applications that need the lowest power possible. See PART 3 for more information.

CLOCK BATTERY:

The DH+ has a high accuracy temperature compensated real-time clock which tracks the current time and date. This is set at the factory to the US West Coast/Pacific Time, but can be changed at any time by the user with the **TIME**, **DATE**, and **CK** commands (see PART 3).

The clock is run from a 3V coin cell battery built into the circuit board which preserves the current time/date even when the switch is in the OFF position. This battery will typically last from 5 to 8 years before it must be replaced.

If the clock battery gets low, the right Red LED will blink quickly with no other LED's being lit. You must replace the clock battery or return the unit to the factory for a non-warranty replacement to continue using the DH+. The battery used is a Panasonic # CR-2032/VCN (or equivalent).

SD CARD:

The Data Hog Plus is designed to be used with a wide variety of SD Cards. Everything from high speed Class 10 cards to low speed Class 1 cards can normally be used without issue. However, use caution when purchasing or using very inexpensive SD Cards. Some manufacturers utilize low-quality memory which either fails outright or does not last more than a few read/write cycles. This is not a failure of the Data Hog Plus, but a result of a badly made memory card.

It is recommended to only use name brand SD Cards manufactured by companies like SanDisk, Kingston, Sony, and Lexar. Avoid any off-brand card which is sold extremely cheaply. These cards will not last and can give the false impression that the Data Hog Plus is not working, not to mention may cause the loss of collected data.

In addition, if you plan on using an SD Card for storing large amounts of frequently changing data (such as video or image files), select an "Industrial" or "High Endurance" type of card. These most often contain advanced features to prevent loss of data after many read/write cycles.

It should be noted that even a high-quality SD Card it will not last forever. Frequent write, read, erase cycles will eventually degrade even the best memory and will require replacement.

SERIAL PORT & NULL MODEM:

To enable communication between two serial devices the transmitter from Device A must be tied to the receiver from Device B, and the receiver on Device A must be tied to the transmitter on Device B. For a typical setup like a computer connected to a modem, the computer has a Male (pins) DB9 connector which is then plugged into a Female (sockets) DB9 connector on a modem. The computer is called the "Data Terminal Equipment or DTE" and the modem is called a "Data Communication Equipment or DCE". All this means is that the wiring inside the DB9 connector and the cable is setup to tie the receive and transmit lines correctly to each other.

The Data Hog Plus has a male DB9 connector and acts just like a computer. It acts in the role of a DTE device with a male DB9 connector and is designed to be plugged into things just like a computer is. But this can cause a problem if you want to connect the DH+ directly to another DTE device (like your computer) because you are now trying to plug to two DTE devices with male DB9 connectors together.

To get around this problem, the Data Hog Plus-G/P comes with a "NULL MODEM ADAPTER". This has two female DB9 connectors on it and reverses the connection so that the DH+ can be connected directly to another DTE device like your computer. 99% of the time this is all that is required.

Unfortunately, some manufacturers don't follow the rule of male pins being DTE devices and female pins being DCE devices. In this case, you may need to customize your serial connection. The following chart shows the exact pinout of the Data Hog Plus DB9 connector:

9 Pin Sub-D

#1 - Carrier Detect	DCD	<-	(Input to Data Hog Plus)
#2 - Receive	RXD	<-	(Input to Data Hog Plus)
#3 - Transmit	TXD	->	(Output from Data Hog Plus)
#4 - Data Term. Rdy	DTR	->	(Output from Data Hog Plus)
#5 - Ground	GND	<->	(Common Ground)
#6 - Data Set Ready	DSR	<-	(Unused by Data Hog Plus)
#7 - Ready To Send	RTS	->	(Output from Data Hog Plus)
#8 - Clear To Send	CTS	<-	(Unused by Data Hog Plus)
#9 - Ring Detect	RNG	<-	(Unused by Data Hog Plus)

PART 3 - Serial Communications Protocol

The Data Hog Plus (DH+) has a single serial port that can range in speed from 300 to 2Mbps (921.6Kbps is the max supported by most other serial products). The serial port is normally used to talk to other devices (like the serial port on a device you want to log data from) where data files are downloaded or settings are changed by the Data Hog Plus automatically.

The DH+ also supports a set of terminal commands that any program or communication system can send it while it in "Command Mode". This mode becomes active when the DH+ receives a single Ctrl+E character (0x05) which is usually sent when plugged into a computer through a NULL-MODEM adapter.

Sign-on (or Log-on) to Terminal Command Mode

Send a single Ctrl+E (0x05) character to the DH+ to enter command mode. While in Command Mode, you can send different commands that control the DH+ like downloading files, formatting the SD card, etc.

The response to the Ctrl+E is a Sign On string like this:

DHP#100a 07/27/20

The values after the "**DHP#**" are the firmware version and date of firmware release. This signals the DH+ is now in *terminal command mode* and is ready to process new commands (see below).

NOTE: If passwords are enabled (they are disabled by default), an asterisk ("*") will appear at the end of the sign-on string. You must send the password with the "PWD=<password>" before any other command will be processed.

Terminal Commands

The following commands are available for general device management:

File & Directory Specific Commands:

- CD <dir> - Changes the current working directory
- TYPE <file> - Outputs a text file.
TABS=<size> - Sets the size of tabs (0=Default, 1-100 fixed size with spaces)
- TYPEH <file> - Outputs a file as hexadecimal values.
- DEL <file> - Deletes a file.
- MD/RD <path> - Makes a directory or removes a directory
- REN <src><dst> - Renames <src> to <dst>
- COPY <src><dst> - Copies <src> to <dst>
- DIR/DIR <path> - Displays current file directory or the files in the sub-directory <path>
- TREE - Displays a tree view of all existing directories from current directory.

SD Card Specific Commands:

- FORMAT=YES! - Reformats the SD Card.
- MEM - Display SD Card status (Size, Amount Used, Disk Status)
- MEM=INIT! - Re-initializes SD Card link.

Upload/Download Commands:

- DL / DLY - Starts a YModem download (stores data on Data Hog Plus). You can optionally specify a <path> and/or <file> name, even though YModem supports this itself.
- DLX <file> - Starts XModem download (stores data on Data Hog Plus) into <file>.
- UL/ULY <file> - Starts YModem upload (data from Data Hog Plus to external device) of <file(s)>.
- ULY <file> - Starts XModem upload (data from Data Hog Plus to external device) of <file(s)>.
- DLDIR <path> - Sets the default download directory for downloading files into DH+.
- ULDIR <path> - Sets the default upload directory for uploading files from DH+.
- STORE <file> - Starts a text file download into file named <file>. If <file> is blank, will create a file named "yyyy-mm-dd hhmmss.txt"

System Commands:

- CK - Clock Get or Set (time or date)
- TIME - Gets or Sets the Time
- DATE - Gets or Sets the Date
- BAUD - Changes the current baud rate (will revert to original baud on power up)
- BOOTBAUD - Changes the current baud rate and sets the default baud rate on power up.
- MODEL - Gets or Sets the type of Data Hog Plus and initializes various options associated with that model.
- DEFDIR <path> - Sets the default directory when system powers on.
- SERIALNUM - Gets/sets the last 10 characters of the serial number (first 10 are set by DET and should not be changed by user).
- LEDS - Turns On or Off LED's (turn off to save power).
- SLEEP - Enters low power sleep mode.
- PWD+ - Set Password
- PWD= - Send Password
- NF=YES! - Start Firmware Upload via YModem.
- RESET - Restores all settings to their original value (except Serial # and Manufacture Date)
- TIMER=<action> - Starts, Stops, Resets, or displays the high speed timer.

Direct Control (most often used with batch file execution):

- SEND <text> - Sends out a specific string. This is normally used in combination with a Batch file.
- RCV <text> - Waits for a specific string to be sent back.
- IF <text> - Checks if the last thing received is equal to <text> and then jumps to <label> if it is.
GOTO <label> See the "Batch File Execution" section for more information.
- LOOP <x><label> - Repeats <x> number of times by jumping to <label>.
- GOTO <label> - Jumps to a point in the batch file.
- REM <text> - Marks the rest of the line as a "remark".
- IDLE <time> - Idles the system until a specific time or for a set number of seconds.
- RUN <file> - Runs a different batch file.
- RESTART <time> - Reboots the Data Hog Plus immediately or after a certain amount of time.
- EXIT <code> - Exits out of running batch file.

File Read/Write (most often used with batch file execution):

- OPENN <file> - Opens a new file for writing. If the file exists, it is automatically overwritten.
- OPENA <file> - Opens a file for writing. If the file exists, new data is appended to the end.
- OPEN <file> - Opens an existing file for reading. If the file does not exist, an error is returned.
- CLOSE - Closes the file opened by OPENN, OPENA, or OPEN.
- OUTPUT <data> - Writes data to an open file.
- INPUT <data> - Reads data from an open file.

Summary of Error Codes

In the case of a terminal command formatting error or if something else goes wrong, the DH+ sends out one of the following error responses instead of either “OK” for a command or the data requested.

ERR#00 (Unknown Command!)
ERR#01 (Password not sent!)
ERR#02 (Incorrect Parameters!)
ERR#03 (SD Card not installed!)
ERR#04 (SD Card not working!)
ERR#05 (SD Card Error!)
ERR#06 (File does not exist!)
ERR#07 (Directory error!)
ERR#08 (Directory does not exist!)
ERR#09 (Directory already exists!)
ERR#10 (Program Flash Upload Error!)
ERR#11 (Rename error!)
ERR#12 (Formatting error!)
ERR#13 (Can't open file!)
ERR#14 (File Delete Error!)
ERR#15 (Download Error!)
ERR#16 (Upload Error!)
ERR#17 (Copy Error)

Batch File Errors:

ERR#20 (Goto Label Error)
ERR#21 (Send format error)
ERR#22 (If/Goto format error)
ERR#23 (CALL nesting error)
ERR#24 (Loop Error)
ERR#25 (File OPEN/CLOSE error)

Terminal Command – Clock (CK)**Format: CK****CK=<time>****CK=<date>****CK=<date>,<time>**

This Command/Enquire returns the current time & date (just “CK” on the command line) or will set the time or the date to a specific value. For example:

CK=12:23:00 (Sets the time to 12:23:00pm)

CK=08/11/17 (Sets the date to August 11th, 2017)

CK=12:23:00,08/11/17 (Sets the time to 12:23:00pm and date to August 11th, 2017)

Just using CK on the command line by itself returns the current time & date like this:

CK

Time=07:01:55 Date=09/12/17

Terminal Command – Clock Time (TIME)**Format: TIME****TIME=<time>**

This is identical to **CK=<time>** and is included to make it easier to remember the command for setting the system time.

Terminal Command – Clock Date (DATE)**Format: DATE****DATE=<time>**

This is identical to **CK=<date>** and is included to make it easier to remember the command for setting the system date.

Terminal Command – New Flash (NF)**Format: NF=YES!**

This function does a firmware update on the Data Hog Plus as follows:

- 1) After this command is sent, the program will send “YModem Start ...” and wait for the initial header block to be sent. Nothing has been done to the original code yet - and won't be until a valid header block is received. Power can be removed and reapplied and system will return to normal operation at this point.
- 2) Once the header block is received, the current program in memory will be erased and the system will download and store the uploaded code into the program flash memory.
- 3) At the end, system sends either “ERR!” on failure or “GO!” if succeeded.
- 4) Data Hog Plus does a hard reset and restarts.

A firmware update is a fairly complex process that must be performed in the exact steps indicated, otherwise a catastrophic failure could occur. You WILL BRICK THE UNIT AND VOID YOUR WARRANTY if you load something in other than approved Data Hog Plus code!

Make sure your battery is fresh before starting this process.

Terminal Command – Send/Set Password**Format: PWD=<x>
PWD+<x>**

Sends or sets the current password. Use “PWD=<x>” to send the current password and DH+ will respond with “OK” if it is correct, or “ERR#02” if it is not. Passwords are NOT case sensitive.

Use “PWD+<x>” to set the password to a new string. This also turns on passwords if they are currently disabled.

Use “PWD+” to disable the password.

Terminal Command – Set Model & Parameters (MODEL)**Format: MODEL
MODEL=<param>**

MODEL controls some basic ways the DH+ operates:

MODEL=G MODEL=G,<log>,<ul/dl> <file>	General purpose Data Hog Plus. Users can control the DH+ through the serial port directly, use a batch file on the SD card, or use the Upload/Download option on power up. There several options: <log> - Set to “0” to disable needing the Ctrl+E to login, or “1” to enable. <ul/dl> - Specifies starting an Upload (ul/ulx/uly) or Download (dl/dlx/dly) with <file> as soon as a SD card and cable is detected. NOTE: Start YModem downloads on power up without a file name like this: MODEL=G,DLY ...or... MODEL=G,DL The DH+ will store the file based on the file name included from the sender.
---	--

For example, to set the DH+ to require a Ctrl+E to logon before any other commands will be accepted::

```
model=g,1  
Model="G" (Login=1/'Yes')
```

To set the DH+ to automatically begin a YMODEM download (data to be stored on the SD card), send:

```
model=g,DLY  
Model="G" (Login=1/'Yes', DLY on Power Up="(determined by sender)")
```

To set the DH+ to automatically begin a YMODEM upload of file TEST.BIN on power up, send:

```
model=g,ULY TEST.BIN  
Model="G" (Login=1/'Yes', ULY on Power Up="TEST.BIN")
```

Terminal Command – LED Control**Format: LEDS**
LEDS=<x>

Turns ON or OFF the LED Display as follows:

- LEDS=0 - LED's are turned off when saves ~40mA of power draw from the battery.
- LEDS=1 - LED's are turned on and operate normally.
- LEDS=2 - LED's are turned on only at very beginning, and then just blink every so often until downloading is complete (when they go back on). They will also come on to indicate an error.
 This is intended as a lower battery usage option.

Terminal Command – Set Serial Number**Format: SERIALNUM**
SERIALNUM=<val>

Gets or sets the last 9 characters of the Data Hog Plus serial number. The first 10 characters are set by Diamond Edge Technology (DET) before shipping the DH+ and are not changeable by the user.

DET sets the serial number of the DH+ to the following:

- DHPmmyyxxx - Where “mm” is the month of manufacture, “yy” is the year of manufacture, and “xxx” is the sequence number (3 digit hexadecimal value).

Setting your own serial number automatically adds a dash (“-”) and then up to 9 characters of your choosing. For example, to set the last part to “MyDatahog”, you would send:

SERIALNUM=MyDataHog

Resulting in a new serial number that looks something:

DHP07200BC-MyDataHog

Note that you can request the Serial Number without first sending the password (if enabled). However, you cannot change it without sending the password.

Terminal Command – Reset to Defaults**Format: RESET**

Resets to the default settings. This does not affect the Serial Number, Baud Rate, or Model. All other settings are returned to their factory set values.

Terminal Command – Set Baud Rate**Format: BAUD=<bps>**
BOOTBAUD=<bps>

Sets the current baud rate (**BAUD**) or sets the current baud rate and sets the default rate the DH+ will use when it powers up (**BOOTBAUD**). Values can range from 300 to 2000000, but note that your serial port may not support rates above 115,200. Always use BAUD to check before using BOOTBAUD to make sure you can cycle power and get the unit back.

Terminal Command – Sleep**Format: SLEEP=<serial>,<time>**

Puts the DH+ into a low power sleep mode that will wake only when it receives a character on its serial port (if <serial>=1), or after <time> number of seconds has passed. <time> can also be set to a specific “hh:mm:ss” value or the word “top” to wakeup at the top of the next hour.

For example:

SLEEP	- Goes to sleep until a character is received.
SLEEP 60	- Goes to sleep until 60 seconds has elapsed.
SLEEP 1, 60	- Goes to sleep for 60 seconds or a character is received.
SLEEP 00:00:00	- Goes to sleep until midnight (serial is shutoff)
SLEEP top	- Goes to sleep until top of next hour (serial is shutoff). For example, if it currently 12:34 it would exit sleep at 13:00.
SLEEP 1, top	- Goes to sleep until top of next hour or a serial character is received.

Sleep power draw and how long a typical 9V battery will last depends on a variety of factors.

- 1) If serial port wakeup is enabled, power draw will be ~22mA.
- 2) If serial port wakeup is disabled, any of the time based wakeups will draw ~10mA if a serial cable is connected, or ~8mA if no cable is connected.
- 3) Estimated battery life with 9V / 550mAh
22mA = 25 Hours
10mA = 55 Hours

Note that the Data Hog Plus can accept any DC power input from 5V to 16V. You can hook up an alternate power source to the 9V battery terminals, or to “J5B” on PCB.

Terminal Command – Memory Status (MEM)**Format: MEM**

Returns the current SD Card status information, or an error message if no card has been found. The DH+ responds with something similar to this:

```
Total Memory: 59GB
Memory Used : 0GB (0%)
SD Card Name: 46HSG V8.0
```

Terminal Command – Memory Initialize**Format: MEM=INIT!**

Re-initializes the SD Card interface. Will return “OK” or “ERR#05” after the initialization.

Terminal Command – Format SD Card**Format: FORMAT=YES!**

Re-formats the installed SD Card. Will return “OK” or “ERR#05” after formatting to indicate the status of the SD Card. Note that cards are formatted to exFAT to maximize their broad usage and large files.

Terminal Command – Change Directory

Format: **CD**
CD <path>
CD ..

The DH+ starts by default in the root directory of the SD card, although this can be changed with the **DEFDIR** command. Sending **CD** by itself will return a string indicating the current directory:

```
cd
Current Directory = "(root)"
```

To change to a sub-directory send **CD <path>** where **<path>** is the name of the directory. For example:

```
cd Dudly
Current Directory = "Dudly"
```

If the directory does not exist, an error will be displayed. To go back up one level, send:

```
cd ..
```

To go back to the root directory regardless of where you currently are, send:

```
cd /
```

<paths> are relative to your current position, unless you put a slash in front of the path name. For example, suppose you are currently in `/Dudly/Test` and you want to go to `/Boo/Test/2` you would send:

```
cd /boo/test/2
```

Note that **<path>**'s are not case sensitive. The DH+ also treats forward slashes ("`/`") and backward slashes ("`\`") the same, but will always report directory names using a forward slash.

You can optionally surround path names with quote marks. However, this is rarely necessary since the DH+ assumes everything after **CD** is the directory name and includes spaces automatically. Only with the **REN** command are quote marks usually needed.

Terminal Command – Directory Listing

Format: **DIR**
DIR <path>

To list all the files and directories in directory send the **DIR** command. Sending it without a specified **<path>** will list files in the current directory, sending it with a **<path>** shows files from that directory. For example:

```
dir
-----
Directory Listing of "dudly":
-----
2020-07-19 23:15:12 <DIR>    test 1
2020-07-19 23:15:14 <DIR>    test 2
2020-07-19 23:15:22 <DIR>    Test 3

          0 Files          0 bytes
          3 Dirs 15931015168 bytes free
```

DIR works almost exactly the same as the Command Prompt in Windows. The DH+ starts by default in the root directory of the SD card, although this can be changed with the **DEFDIR** command.

With firmware V1.04 (and higher) **DIR** also supports wildcard characters such as:

```
DIR *.txt    Displays only files ending in ".txt" from current directory.
```

Terminal Command – Directory Tree View**Format: TREE
TREE <path>**

Displays a tree structure view of all the directories on the SD card. With nothing specified for <path> it starts at the current directory, otherwise you can use:

- TREE \ - Always display entire directory tree from the root directory.
- TREE "\Foo" - Displays tree view of the Foo directory off the root.
- TREE ".." - Displays a tree view starting in the directory up one level from this one.
- TREE "Foo" - Displays a tree view starting in sub-directory Foo from current directory.

Terminal Command – Make Directory**Format: MD <path>**

Makes a new directory. For example, to make a new directory from the current directory send:

```
md Foo Bar
```

Or you can send:

```
md "Foo Bar"
```

Another way to always make the directory from the root:

```
md "\Foo Bar"
```

Terminal Command – Remove Directory**Format: RD <path>**

Removes a directory and all files and sub-directories in that directory. For example, to remove the director "Foo Bar" from the root you can send:

```
rd "\Foo Bar"
```

This can be a dangerous command because it can quickly remove a lot of files. Use it carefully!

Terminal Command – Delete File**Format: DEL <file>**

Deletes a file from the SD card. For example, to remove the file "Dud.txt" from the root you can send:

```
del "\Dud.txt"
```

To delete a file in the current directory, send:

```
del "dud.txt"
```

You can also use wild cards in the file name. To erase all files ending in ".txt". send:

```
del *.txt
```

Terminal Command – Rename File or Directory**Format: REN <src> <dst>**

Renames <src> to <dst>. This command works on files or directories, and can also move a file from one directory to another.

This command benefits from surrounding the <src> and <dst> with quote marks (helps to identify them), and is required if the <src> has any spaces in the name.

To rename file "Foo Bar.txt" to "Foo.bin", send the command:

```
ren "Foo Bar.txt" "Foo.bin"
```

Terminal Command – Copy File or Directory	Format: COPY <src> <dst>
---	--------------------------

Copies <src> to <dst>. This command works on files or directories, and wildcards can be used for the <src> but not for the <dst>.

This command benefits from surrounding the <src> and <dst> with quote marks (helps to identify them), and is required if the <src> has any spaces in the name.

To copy the file “Foo Bar.txt” to “Foo.bin”, send the command:

```
copy "Foo Bar.txt" "Foo.bin"
```

NOTE: COPY requires firmware V1.04 and higher.

Terminal Command – Type out a File	Format: TYPE <file> TYPEH <file> TABS <size>
------------------------------------	--

Outputs the contents of <file> to the serial port. Output can be aborted by sending a Ctrl+X (0x18) or Ctrl+Z (0x1A). Renames <src> to <dst>.

TYPE outputs the file as straight text. **TYPEH** outputs it as a hexadecimal conversion.

By default **TYPE** outputs tabs as a straight character (0x09). However, you can set it with the **TABS** command as follows:

- TABS=0 - Output tab characters as a straight 0x09.
- TABS=<size> - Replace tabs with spaces and set <size> to 1-100 in width.

Terminal Command – Default Directory	Format: DEFDIR DEFDIR <path> DLDIR DLDIR <path> ULDIR ULDIR <path>
Download Directory	
Upload Directory	

These commands display or set the default file directory to use on power up (**DEFDIR**), the directory to store files in when doing a download (**DLDIR**), and the default directory to use when uploading files from the DH+ (**ULDIR**).

Note that when downloading files using the YMODEM protocol, a file directory can be part of the name. In this case, the file is always stored as though the **DLDIR** is the root directory.

In a similar way, when uploading files with YMODEM the **ULDIR** is treated as the root directory and is not included in any path name for the file.

The DEFDIR value will automatically be created on any SD Card when it is installed in the DH+.

Terminal Command – Store Text File**Format: STORE <file>
STOREA <file>**

Creates a new file named <file> and beginning storing everything received from the serial port into the file. File is closed automatically on receipt of Ctrl+X (0x18) or Ctrl+Z (0x1A) or if the cable is unplugged. If no file name is given, then a file named “yyyy-mm-dd hhmmss.txt” is automatically created.

STOREA differs from STORE in that it will automatically append new data to the end of <file> if it exists (otherwise it creates a new file). STORE will always overwrite any existing file with the new data.

Note that characters are not echoed back during storage. The DH+ assumes this is being done under software control and does not need to see the actual data being stored. To view the text after it has been stored, use the TYPE command.

The file directory used to store the file uses the following logic:

- 1) If the file name starts with a slash (“/”) and then a file name, then it will always go in the root directory.
- 2) If the file name is just a single slash, then a file “yyyy-mm-dd hhmmss.txt” will go in the root directory.
- 3) If the default download directory is blank, or you start the file name with “./”, then the file will be stored in the current directory plus whatever directory has been specified in <file>.
- 4) All other situations will store the file in the default download directory plus whatever directory has been specified in file.

Some examples:

STORE

Creates a “yyyy-mm-dd hhmmss.txt” file in the default download directory.

STORE /

Creates a “yyyy-mm-dd hhmmss.txt” file in the root directory.

STORE ./

Creates a “yyyy-mm-dd hhmmss.txt” file in the current directory.

STORE Dud.txt

Creates file “Dud.txt” in the default download directory.

STORE /Dud.txt

Creates file “Dud.txt” in the root directory.

STORE /MyDir/Dud.txt

Creates file “Dud.txt” in the “/MyDir” directory.

STORE ./Dud.txt

Creates file “Dud.txt” in the current directory.

STORE ./MyDir/Dud.txt

Creates file “Dud.txt” in the sub-directory “MyDir” from the current directory.

STOREA Dud.txt

Appends new text data to file “Dud.txt” in the default download directory.

Terminal Command – Download File

Format: **DL** / **DL** <file>
DLY / **DLY** <file>
DLX <file>

Download transfers a file from a remote system (like a computer) to the Data Hog Plus SD card. The parameters for <file> are the same as the **STORE** command (see above).

DL and **DLY** are identical and start a YMODEM CRC protocol download (either 1K or 128 byte), including allowing the sending device to specify the <file> (use **DL** or **DLY** without a file name to allow this). The Data Hog Plus implements standard YMODEM with 1K block sizes and a CRC check.

DLX does an XMODEM download instead. You must specify the file name for XMODEM.

Refer to the following for specific details on how XMODEM and YMODEM works:

<http://textfiles.com/programming/ymodem.txt>

<http://web.mit.edu/6.115/www/amulet/xmodem.htm>

Once the download is complete the DH+ will send “OK (Download Complete!)”.

NOTE: **DLY** supports multiple file transfer. The sending device can send as many files as it wants and the DH+ will store them as received.

Terminal Command – Upload File

Format: **UL** <file>
ULY <file>
ULX <file>

Upload sends a file from the SD card on the DH+ to a remote device over the serial port. The options for <file> are the same as the **STORE** command (see above).

UL and **ULY** are identical and start a YMODEM 1K with CRC protocol upload with the file named. **ULX** starts a XMODEM 1K transfer with CRC upload.

Refer to the following for specific details on how XMODEM and YMODEM works:

<http://textfiles.com/programming/ymodem.txt>

<http://web.mit.edu/6.115/www/amulet/xmodem.htm>

Once the upload is complete the DH+ will send “OK (Upload Complete!)”.

NOTE: **UL** and **ULY** supports multiple file transfer. Simply specify a wildcard in the file name to select more than one file to send. For example:

UL *. *	Sends all files in the directory.
UL *.txt	Sends all files ending in “.txt”.

PART 4 - Batch File Protocol

The Data Hog Plus (DH+) supports a complete batch file system which allows almost complete control over the unit and what it is connected to. Simply by placing a file in the root directory of the inserted SD Card, you can instruct the Data Hog Plus to send out characters or strings, wait for certain responses, and upload/download different files depending on the responses received.

This feature allows users to insert different SD cards for different purposes, and makes the Data Hog Plus almost infinitely configurable as a portable data retriever, programmer, and system updater.

All batch files are formatted in standard text file format with one command per line ending in a CR/LF. Leading spaces or tabs are ignored, and comments can be inserted by starting the line with a semicolon (;), a forward slash (/), or the command "REM". For example:

```
REM This is my batch file to change to the \DATA directory
REM and display the file "HELLO.TXT".
CD \DATA
TYPE "HELLO.TXT"
```

An identical version of this batch file would be:

```
; This is my batch file to change to the \DATA directory
; and type out the file "HELLO.TXT".
CD \DATA
TYPE "HELLO.TXT"
```

Comments should be used liberally throughout the batch file to make clarification easier later on if you need to modify the file in any way.

Note that unless a batch file has a **GOTO** command, it will automatically end after the last command is executed. You can terminate a batch file before the last command using the **EXIT** command:

```
; This is my batch file to change to the \DATA directory
; and type out the file "HELLO.TXT".
CD \DATA
EXIT
TYPE "HELLO.TXT"
```

In this case the Data Hog Plus will change to the \DATA directory, but it won't type out "HELLO.TXT" because the **EXIT** command terminates it immediately.

Batch files can be any length and can do all sorts of advanced things like:

- Send out characters and wait for certain responses.
- Start uploads or downloads.
- Run other batch files.
- Run any regular terminal command (not just batch file commands).
- Use **IF/GOTO** and **LOOP** statements to branch around the batch file and execute different commands based on certain things happening.

Batch files are a combination of a very simple programming language with a list of file, directory, and data transmission commands that operate on the SD card and the connected device. See PART 4A – BATCH FILE EXAMPLES for more information.

NOTE: On power up (or when a new SD card is inserted) the DH+ looks to see if "AUTOEXEC.BAT" or "AUTORUN.BAT" exists in the root directory of the SD card. If found, it automatically runs it. Otherwise, batch files are only executed in response to the **RUN** command.

Batch File Command – Send Data Out

Format: SEND <format>

SEND can be a simple or complex command. In the simplest form, you simply list the characters you want to output like this:

```
SEND "hello"
```

And the DH+ would send out the serial port: hello

Note that the quote marks are not included unless you put a backslash in front of them like this:

```
SEND "\"hello\""
```

And the DH+ would output: "hello"

It is not required to start and end what you are going to send with quotes, but it is generally a good idea to avoid confusion. Here are some of the other values that can be included:

Code	Explanation
\r	Carriage Return (0x0D)
\n	New Line (0x0A)
\t	Tab Character (0x09)
\"	Double quote
\%	Percent (cannot just include these because they are used for parameters, see below)
\\	Backslash character
\xYY	A single character represented by the hex value of YY. For example, to output an 0xFE byte, you would use: SEND "\"xFE" You can string multiple characters together too: SEND "\"xFE\x7F\xFF\x00"
\pSSS	Does not send a character but pauses for SSS number of seconds. For example, to send the character R, pause for 10 seconds, and then a T, use: SEND "R\p010T"
\mNNN	Repeats the previous character NNN times. For example: SEND "-\m077\r\n*\m077\r\n" This would output a dash 78 total times, a new line (CR/LF), then an asterisk 78 times, then a final new line (CR/LF).

The percent character (%) is special purpose. It allows you to insert previously saved values that are usually loaded by the **RCV** command. Here are the % codes:

Code	Explanation
%1 - %9	Saved variables from the RCV command. For example, the following receives a line of text from the serial port ended by a Carriage Return (0x0D) and then returns it back followed by a CR/LF: RCV "%s" SEND "Received: %1\r\n" When this is run, the DH+ waits to receive a line of text and when it is received (indicated by getting a Carriage Return), it sends back what it got saying that is stored in %1: Received: TEXT Refer to the batch file examples and the RCV command for more information.
%0	The last EXIT code received ("0" is the default).
%h %n %s %H %N %S	The current Hour (%h), Minute (%n), and Second (%s) Same as the above except always as two digits.
%y %m %d %Y %M %D	The current Year (%y), Month (%m), and Day (%d) Same as the above except always 4 digit and 2 digit.
%r	Set to a "1" if any data is ready on serial port to be received, or a "0" if not.
%t %T %e %E	High Speed Timer as "days hh:mm:ss.msec" High Speed Timer as "days.fractime" High Speed Timer as "seconds.msec" High Speed Timer as "hh:mm:ss.msec"

Batch File Command – Receive Data**Format: RCV <format>**

RCV inputs data from the serial port. If a cable is detected when the command starts, then if that cable becomes disconnected during it will automatically abort the **RCV**.

There are two types of things that can be received – fixed characters and variables. Fixed characters are specific characters that must be read before moving on. Variables are different kinds of values that can be specified and then are stored in the %1 through %9 variables.

For example, to simply wait until the characters “GO” are received, use the command:

```
RCV "GO"
```

The DH+ will wait until “GO” is received before finishing. Note that it has to be ‘G’ then ‘O’ in that order, any other combination will not match. To wait for “GO” and then one more character of any type, use:

```
RCV "GO%c"
```

The “%c” in this case says to receive a single character. It is automatically saved in the %1 variable because it is the first thing being received. To do the same thing with two characters:

```
RCV "GO%c%c"
```

Now %1 will equal the first character received after “GO”, and %2 will equal the second character. Alternately, you can use:

```
RCV "GO%2c"
```

Both characters that are both stored in %1 (one after the other). See the **IF/GOTO** command to perform checks on the variables and change the batch file flow based on what is received.

Regular characters have special codes for certain characters (like the **SEND** command):

<i>Code</i>	<i>Explanation</i>
\r	Carriage Return (0x0D)
\n	New Line (0x0A)
\t	Tab Character (0x09)
\"	Double quote
\%	Percent (cannot just include these because they are used for parameters, see above)
\\	Backslash character
\xYY	A single character represented by the hex value of YY. For example, to wait to receive a 0xFE, you would say: RCV "\xFE"
\mNNN	Repeats previous character NNN times.

Format of the “%” commands:

<i>Code</i>	<i>Explanation</i>
%c	Single character (Tab, punction, or upper/lower case letter).
%<num>c	Multiple characters specified by <num>. For example, “%5c” loads five characters.
%b	Single byte of any value.
%<num>b	Multiple bytes specified by <num>. For example, “%5b” loads five bytes.
%s	String terminated by a CR (0x0D), LF (0x0A), Break (0x03), or Abort (0x18). RCV “%s” Note that the character that terminates the string is not included in the result.
%i	Integer terminated by a comma, CR (0x0D), LF (0x0A), Break (0x03), or Abort (0x18).
%<num>i	Receives a specific number of integer characters.
%f	Floating point number terminated by a comma, CR (0x0D), LF (0x0A), Break (0x03), or Abort (0x18).
%<num>f	Receives a specific number of floating point characters.
%x	Hexadecimal number terminated by a comma, CR (0x0D), LF (0x0A), Break (0x03), or Abort (0x18).
%<num>x	Receives a specific number of hex characters.

Batch File Command – Goto Label**Format: GOTO <label>**

The **GOTO** command jumps to a new line in the batch file. This line can be either before or after the **GOTO** command, and is identified by a unique <label> identifier that starts with a colon (":"). For example:

```
CD \  
GOTO NEWLABEL  
CD \NEWDATA  
EXIT  
:NEWLABEL  
CD \OLDDATA
```

This batch file jumps around the "CD \NEWDATA" to "NEWLABEL" and then runs "CD \OLDDATA". **GOTO** is especially useful when combined with **IF**, **CALL**, and **RUN** commands.

Batch File Command – If Then Goto**Format: IF <condition> GOTO <label>**

The **IF** command tests the results of previously stored values to see if they match specified conditions. If they do, then the batch file will branch to the label specified after the **GOTO**.

This command can be hard for those new to programming to understand, but it is really pretty simple. Consider the following batch file:

```
:Top  
SEND "Type 0 or 1: "  
RCV "%c"  
IF %1 = "0" GOTO GotZero  
IF %1 = "1" GOTO GotOne  
SEND "\r\nERROR: Did not type 0 or 1!\r\n"  
GOTO Top  
:GotZero  
SEND "\r\nGOT A ZERO!\r\n"  
EXIT  
:GotOne  
SEND "\r\nGOT A ONE!\r\n"
```

This code starts out by sending "Type 0 or 1: " out the serial port. It then waits for a character to be received back. Once that is received, it checks to see if it is a "0" and jumps to the line after the "GotZero" label if it is. If it is a "1", it jumps to the line after the "GotOne" label. If it is neither of these, it sends an error message and then retries!

You can see how the "RCV %c" is used to receive the character and it is automatically stored in the %1 variable. This %1 value is what the IF command uses to check against to see what was actually received.

IF supports the following comparisons:

= (Equals)	>= (Greater Than or Equals)
!= (Not Equals)	<= (Less Than or Equals)
< (Less Than)	
> (Greater Than)	

For example:

```
IF %4 >= "12" GOTO MyLabel
```

If %4 is greater than or equal to 12, it jumps to MyLabel.

Batch File Command – Loop Goto**Format: LOOP <count> GOTO <label>**

The **LOOP/GOTO** command allows the batch file to repeat a loop a specified number of times. Consider the following batch file:

```
:Top
SEND "Here!\r\n"
LOOP 10 GOTO Top
```

This would send out the string "Here!" ten times. You can also use a % variable for the count:

```
SEND "How many times? "
RCV "%i"
SEND "\r\n"
:Top
SEND "Here!\r\n"
LOOP %1 GOTO Top
```

The **RCV** command will store an integer in %1, which will control how many times it loops and sends the "Here!" line.

NOTE: A loop count of one or zero results in no action. The DH+ assumes the first time through you have done one loop. In addition, don't use nested **LOOPs** inside of **CALLs**. The DH+ only supports one **LOOP** at a time, and a loop inside a **CALL** may result in the main batch file **LOOP** being repeated infinitely.

Batch File Command – Idle**Format: IDLE=<serial>,<time>**

The **IDLE** command is identical to the **SLEEP** command except that if it is set to wake from a serial character received, that character is not read out and is available to the next command.

For example, suppose you enter:

```
IDLE
```

And the DH+ receives the character 'R'. That 'R' character will become the first character read out by the next RCV command instead of being discarded.

Batch File Command – Run Batch File**Format: RUN <file>**

Executes a batch file. By custom, batch files should end in ".BAT" but this is not required by the system. Note that running a batch file from another batch file is allowed, but when the new batch file ends it does NOT return to the old file (use the **CALL** command instead).

The standard file name conventions apply like this:

RUN TEST.BAT	- Runs the batch file named "TEST.BAT" in the current directory.
RUN \TEST.BAT	- Runs the batch file named "TEST.BAT" in the root directory.
RUN FOO\TEST.BAT	- Runs the batch file named "TEST.BAT" in the FOO sub-directory from the current directory.

NOTE: If the file does not exist exactly as entered, the DH+ will automatically search for a file ending in ".BAT" with the same name and run that instead. For example, if you sent:

```
RUN TEST
```

If a file named "TEST" does not exist but a file named "TEST.BAT" does, then DH+ will run "TEST.BAT".

Batch File Command – Exit Batch File**Format: EXIT**
EXIT <code>

Stops execution of batch file. Optionally, you can exit with a code which can be used in other batch files to do different things (see the “CALL” command). For example:

```
CD \DATA
EXIT
CD \
```

In the above batch file example, the current working directory will stay “\DATA” because it exits out of running the batch before getting to the last line.

EXIT is especially helpful with the **CALL**, **IF/GOTO**, and **RUN** batch file commands.

Batch File Command – Call Batch File**Format: CALL <file>**

Executes a different batch file, similar to RUN, but when the new batch file exits it returns to the current batch file at the same position. In addition, the %0 variable will be set to the **EXIT** code.

Consider the following two batch files:

File Main.BAT:

```
:Top
SEND "Type in a string and press Enter: "
RCV "%s"
IF %1="" GOTO Done
CALL ShowAnswer.BAT
GOTO Top
:Done
SEND "\r\nDONE!\r\n"
```

File ShowAnswer.BAT:

```
SEND "\r\nANSWER=\"%1%\r\n"
```

The DH+ will ask for a string and when the user types one in, it calls “ShowAnswer.BAT” to display the results. If the user types an empty string, it quits.

Note that all variables (%1 through %9) are common to all functions. The %0 variable will always contain the **EXIT** code of the last batch file that exited, with a default value of “0” if no other value specified.

You can **CALL** other batch files from inside of batch files that are themselves **CALL**ed. However, you cannot go more than ten layers deep.

Batch File Command – Remark (Command)**Format: REM ...**
; ...
/ ...

Simply marks the rest of the line as a comment and is ignored by the Data Hog Plus. For example:

```
REM This is my batch file to change
; This is my batch file
/ This is my batch file
```

All of these result in the rest of the line being ignored.

Batch File Command – Restart Data Hog Plus

Format: **RESTART**
RESTART=<sec>
RESTART=<time>
RESTART=TOP

Restarts the Data Hog Plus. This command is identical to switching the power switch off and then on again.

Optionally include the number of seconds you want to wait or a specific time you want to restart at. During this period the Data Hog Plus ignores everything and won't respond again until this time has passed.

For example:

RESTART	- Immediately restarts
RESTART 60	- Goes to sleep and restarts after 60 seconds.
RESTART 00:00:00	- Goes to sleep and restarts at midnight.
RESTART top	- Goes to sleep and restarts at top of next hour (serial is shutoff). For example, if it currently 12:34 it would exit and restart at 13:00.

Batch File Command – File Open

Format: **OPEN <file>**
OPENN <file>
OPENA <file>

Opens an existing file or creates a new file named <file> and beginning reading everything from it when the **INPUT** command is use or storing everything sent by the **OUTPUT** command to it.

OPENN creates a new file, and **OPENA** creates a new file if none exists or appends new data to the end of any existing file. **OPEN** is used to read data out of existing files.

The file directory used has the following logic:

- 1) If the file name starts with a slash ("/") and then a file name, then it will always go in the root directory.
- 2) If the file name is just a single slash, then a file "yyyy-mm-dd hhmmss.txt" will go in the root directory.
- 3) If the default download directory is blank, or you start the file name with "./", then the file will be stored in the current directory plus whatever directory has been specified in <file>.
- 4) All other situations will store the file in the default download directory plus whatever directory has been specified in file.

Some examples:

OPENN

Creates a "yyyy-mm-dd hhmmss.txt" file in the default download directory.

OPENN /

Creates a "yyyy-mm-dd hhmmss.txt" file in the root directory.

OPENN ./

Creates a "yyyy-mm-dd hhmmss.txt" file in the current directory.

OPENA Dud.txt

Creates file "Dud.txt" in the default download directory if it doesn't exist, otherwise adds to this file.

OPENA /Dud.txt

Creates file "Dud.txt" in the root directory if it doesn't exist, otherwise adds to this file.

OPEN /MyDir/Dud.txt

Opens the existing file "Dud.txt" in the "/MyDir" directory for the **INPUT** command.

Batch File Command – File Close**Format: CLOSE**

Closes any file that has been opened with the OPEN, OPENN, or OPENA command. This command is optional – the DH+ automatically closes the open file if another file is opened or the batch file exits.

Batch File Command – Output to File**Format: OUTPUT <text>**

The **OUTPUT** command sends the data specified by <text> to the file previously opened with the **OPENN** or **OPENA** command. The format of <text> is identical to that described by the RCV command (see above). For example:

```
OPENN "MyFile.txt"
OUTPUT "This is a line of text!\r\n"
CLOSE
```

At the end of this sequence, the file "MyFile.txt" will contain "This is a line of text!" followed by a CR/LF. You can include % variables in the output as well as any backslash code.

If you wanted to append to an existing file, you would use:

```
OPENA "MyFile.txt"
OUTPUT "This is a line of text!\r\n"
CLOSE
```

In this example, every time the sequence is run another "This is a line of text!" is added to the file.

Batch File Command – Input from File**Format: INPUT <text>**

The **INPUT** command is just like the **RCV** command, except it reads data from an existing file that has been opened with the **OPEN** command. The format of <text> is identical to the RCV command except the characters are read out from the file instead of from the serial port.

Note that any **INPUT** command automatically ends if the end of the file is reached and an error message is shown.

Batch File Command – High Speed Timer**Format: TIMER
TIMER=START/STOP
TIMER=RESET**

The timer is a high speed timer that can be reset, started, and stopped at any time. To access the current value, simply send TIMER or use the "%t", "%T", "%e", or "%E" variables.

The timer is useful for various functions and helps in storing timestamps in files.

As previously described, the Data Hog Plus has nine main variables for use during batch file operation numbered %1 to %9. In addition, %0 represents the value of the last **EXIT** code which is used in combination with the **CALL** function.

Most of the time % variables are set with the **RCV** and **INPUT** commands. These read data out of the serial port or from a file, respectively. For example:

```
RCV "%s"
```

This will read a string of characters out of the serial port terminated with a CR and store the result in the %1 variable. If you used instead something like:

```
RCV "%c%s"
```

Then the first character would be stored in the %1 variable and then a string terminated by a CR would be stored in the %2 variable.

Another option is to assign directly to a variable a different string or variable. For example:

```
SEND "Type in string and press Enter: "  
RCV "%s"  
%2 = "Got \"%1\"!\r\n"  
SEND "\r\n%2"
```

This would get a string and store it in %1. It then assigns to the %2 variable the word "Got ", a quote, whatever was received by the **RCV** command, and then a trailing quote, exclamation, and CR\LF. It then sends that new %2 value back out.

Assigning values to variables is especially useful if you want to preserve time stamps or other values that might change between accessing them.

PART 4a - Batch File Examples

Refer to the SAMPLES sub-directory on the SD card that came with the Data Hog Plus. This directory contains several sub-directories with different kinds of example batch files.

Other examples may be present, but the following is a brief description:

Send Time Once Per Hour

Will send the time when it first runs, and then send it again at the top of each hour.

Serial Data Logger (Text)

When DH+ powers up, it immediately opens a new text file with a name based on the current date and time. All text received is stored in this file. The file is closed if the DH+ receives a Ctrl+X, and then another new text file is opened with a new date and time.

This allows you to use the DH+ to simply record all text data that it receives.

Serial Data Logger (YModem)

When DH+ powers up, it immediately starts a YModem download. At the end of the download, it will start another YModem download.

This allows you to use the DH+ to receive files via YModem continuously from whatever it is connected to. Since YModem has the file name in the header, the device it is connected to can set the file name to be saved into.

Serial Data Logger (Text or YModem)

Similar to the previous two examples except it first sends out a message saying "Send 0 for Text or 1 for YModem". Once it receives one of those two characters, it starts either a Text or YModem download. When finished, it re-asks "Send 0 for Text or 1 for YModem".

Stop Watch

Sends the message "Stop Watch (SPACE to Start/Stop):". When it receives a SPACE or a CR, it starts the stop watch and sends out the hours, minutes, and seconds each second. Send another SPACE or CR to stop the timer.

Send Ctrl+X to abort and exit the Stop Watch example.

Stop Watch Logger

Same as Stop Watch except it also opens a text file and stores the stop watch times to a file named "STOPWATCH.TXT". New data is appended to existing data.

Send Ctrl+X to abort and exit the Stop Watch example.

Data Time Stamp

Opens a new file based on the current date and time. Any character that is received is immediately stored in the file along with an exact time stamp of that character.

If a Ctrl+X is received, the file is closed and a new file is opened with the current date time. If a Ctrl+Z is received the batch file terminates.

Refer to the SAMPLES sub-directory for more information and other examples. The DH+ is very customizable for your particular system needs!

PART 5 – Updating Data Hog Plus Firmware

The Data Hog Plus firmware can be updated fairly easily. Diamond Edge Technology provides updates periodically (refer to www.dettllc.com) for firmware labeled like this:

DHP_Vxxxxa.HEX

The “xxx” will be the version number and the “a” refers to any customization for specific customers. You can determine what version you are currently running by:

- 1) Open a terminal window on your computer set to the baud rate of the DH+ and the COM port you have connected your NULL MODEM cable to.
- 2) Send Ctrl+E to get the sign-on string like this:

DHP#100a 08/02/20

- 3) This shows the currently connected DH+ is V1.00a.

If you want to update the firmware on the Data Hog Plus, send the following command:

NF=YES!

The DH+ will respond with a YMODEM Upload message. Start your YMODEM Upload of the new version of firmware. Once the upload is complete the DH+ will automatically restart.

WARNING: You will “brick” your Data Hog Plus and void the warranty if you upload anything to it other than approved firmware from Diamond Edge. You should also be sure to have a fresh 9V battery installed before starting the process, and there is no chance that your computer will lose power during the transfer.