Heavy-duty Vehicle (HDV)

Data Streamer

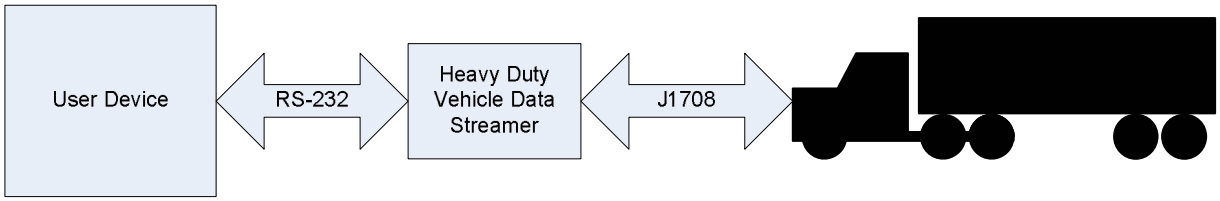
Command and Response Protocol

Table of Contents

Revision History

Version	Primary Author(s)	Description of Version	Date Completed
1.00	Mark Kloza	Initial High Level Software Design	11/10/2005
1.01	Mark Kloza	Updates to Firmware specification 1.09	11/16/2005
1.02	Mark Kloza	Updated after comments from B&B on V1.01	1/12/2005
1.03	Mark Kloza	Updated after first phase of development	1/30/2006
1.04	Mark Kloza	Updated RS-232 and broadcast commands	2/10/2006
1.05	Mark Kloza	Updated “send immediately” command	2/22/2006
1.06	Mark Kloza	Added “Firmware update” and J1708 transmit commands	3/01/2006
1.07	Kevin Nelson	Added General Command Format Section Corrected some errors Added Checksum to commands with constant strings Added RS-232 Baud Rate table	3/10/2006
1.08	Mark Kloza	Corrected Retrieve Filter Values command	4/04/2006
1.09	Kevin Nelson	Clarify some byte information in Start Polling Non-Broadcast Parameters command Change nomenclature from Gateway to Data Streamer Add list of default MID/PID filters	9/25/2006
1.10	Kevin Nelson	Added Calculated Trip Odometer read command Changed Stop Polling Non-Broadcast Parameters command to 2 data bytes.	3/27/2007
1.11	Kent Anthony	Add trip parameter commands to the list of valid ID codes.	10/8/2007

1. System overview



2. Operational Description

2.1. Firmware Functionality

Upon receiving a valid HDV Streamer IM/MID/PID setup command following action will be undertaken;

1. Table containing information about HDV Streamer acceptable MID/PID pair values and corresponding Immediate Send flags values will be setup and updated.
2. The same table values will be stored into non-volatile memory. MID/PID pairs and corresponding Immediate Send flag values will be restored from non-volatile memory at power up.
3. Broadcast status (if parameter value is being broadcast on J1708 bus) for each MID/PID pair will be maintained internally and retrieved together with PID values for given MID/PID pair.

Upon receiving a valid J1708 message, the Streamer **filter** functionality will include;

4. Compare received MID/PID pair against stored filter list of MID/PID pairs. If MID/PID pair is present in that list then received MID/PID parameter value will be stored for eventual farther retrieval.
5. If above MID/PID pair has corresponding Immediate Send flag set, then the message containing the parameter value for that MID/PID pair will be queued in the transmit buffer for RS-232 bus in a format as described in MID/PID value retrieval command.
6. Compare PID for Diagnostic Code 194. Store internally the parameter value together with the MID value.
7. If above PID (194) is configured to send information immediately then queue MID/PID194 pair parameter value to RS-232 bus.

Upon receiving valid **command to retrieve MID/PID (including PID194) pair parameter value**:

8. Message containing stored parameter value corresponding to requested MID/PID pair will be qued to transmit buffer for RS-232 bus in a format as described in MID/PID parameter value retrieval command.

9. Information if retrieved value had been updated since last retrieval command has been received will be provided.

Upon receiving valid **command to retrieve MID/PID/IM and PID194 setup:**

10. Message containing IM/PID/MID setup plus PID 194 Immediate Send flag will be qued to transmit buffer for RS-232 bus.

Upon receiving valid **command to set Unique Identifier:**

11. Unique identifier of 10 characters plus 'null' character will be set and stored into non-volatile memory.

Upon receiving valid **command to retrieve Unique Identifier:**

12. Message containing unique identifier of 10 characters plus 'null' character will be queued to transmit buffer for RS-232 bus.

Upon receiving valid **command to setup RS-232 Baud Rate Setup:**

13. Baud rate, Parity and Stop bit information will be stored into non-volatile memory. This information will be retrieved on reset.

14. Default values for MID/PID pair setup, RS-232 values, Unique Identifier will be implemented.

3. Command and Response Protocol

3.1. General Command Format

The Serial adapter uses a simple protocol to communicate to the vehicle bus. The protocol is divided into four parts:

Start Of Frame (SOF)
Control Field (CF, shown in light gray)
Data Field (DF, shown in dark gray)
Checksum (CS)

Message Structure

SOF	Number of Control Bytes	Control bytes	Number of Data Bytes	Data Bytes	CS
1 byte	1 byte	1 to 20 bytes	1 byte	1 to 100 bytes	1 byte

The SOF is the first byte in a valid frame and is always $01.

The Control Field sets up the function and control of the Serial adapter hardware, and is formatted as shown below

Control Field Structure

Number of Control Bytes to follow	ID byte	Function Byte	Control data bytes
1 byte	1 byte	1 byte	Up to 20 bytes

The first byte of the Control Field is the number of Control Bytes in the message. This value indicates the number of control bytes, excluding itself, to follow. If the message doesn't contain any control bytes, the Number of Control Bytes value will be set to $00 as a placeholder.

The next part of the message frame is the data field. The data field is preceded by the number of Data Bytes, excluding itself, to follow and is set to $00 if no data is present in the message. The data field is reserved for data that is to be communicated to the vehicle bus, or configuration data for the Serial Interface.

Data Field Structure

Number of Data Bytes	Data Bytes
1 byte	Up to 100 bytes

The last element of a valid message is the checksum. The checksum is the 8bit result of adding all of the bytes from the SOF to the last data byte.

3.2. Valid ID Codes and Function Codes

The following is a list of valid ID Codes and Function Codes. After the SOF and number of Control Bytes data is received, the next data byte is the ID Code. This byte tells the hardware where to direct the messages. After the ID Code comes the Function Code (Not always part of the message). The supported Function Codes are also listed below.

Valid ID Codes:

= $01 Message for J1708
= $02 Retrieve Model Number
= $05 Retrieve Firmware version
= $08 Bus-Independent Configuration
= $09 Reserved

Valid J1708 Functions Codes (ID $01):

= $05 Retrieve Trip Odometer Reading
= $06 Reset Trip Odometer
= $08 IM/MID/PID filter setup
= $09 IM/MID/PID filter setup retrieval
= $14 MID/PID parameter value retrieval
= $x7 Start/Stop Polling Non-broadcast Parameters
= $07 Stop Polling Non-broadcast Parameters
= $12 Message to Transmit to the J1708 Interface

Valid Bus-Independent Configuration Function Codes (ID $08):

= $01 Change RS-232 baud rate
= $05 Firmware Upgrade
= $12 Unique Identifier setup
= $13 Unique Identifier retrieval

3.3. Command Acknowledgement

All commands that do not have an immediate response from the hardware will be confirmed with an Acknowledgement reply from the Streamer. The acknowledgement will be formatted as follows:

Command Acknowledgement format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	ID	ID byte (01 or 08)
4	00	Number of data bytes
5	CS	RS-232 Message Checksum

3.4. Command Formatting Errors

If the hardware detects an error in the command from the host, an error notification will be returned. The error notification format and Error Code descriptions are shown below.

Error Notification Format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	03	Number of control bytes
3	05	Function code error = 05
4	ID	ID of message that caused error
5	EC	Error code (see chart below)
6	00	Number of data bytes
7	CS	RS-232 Message Checksum

Error Code Descriptions:

Error Code		Type of Error Generated	Meaning of Error Generated
DEC	HEX		
1	01	General (ID = $08)	Inbuffer overflow; more than 127 bytes
2	02	General (ID = $08)	Checksum mismatch
3	03	General (ID = $08)	Protocol error; function, ID, code bad
4	04	J1708 (ID = $01)	J1708 input buffer full missed bus message
5	05	J1708 (ID = $01)	J1708 command unknown
6	06	J1708 (ID = $01)	J1708 output buffer full
7	07		
8	08		
9	09		
10	0A	J1708 (ID = $01)	J1708 inbuffer overflow more than 127 bytes
11	0B		
12	0C		
13	0D		
14	0E		
15	0F		

4. Detailed Command Descriptions

4.1. IM/MID/PID Filter Setup Command

Up to 20 sets of setup information will be accepted and stored into non-volatile memory to filter parameters from the J1708 bus.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	08	Function code: 0x08 – IM/MID/PID setup
5	3D	Number of data bytes in a message (61dec = 3D hex)
6	IM1	Data set 1: IM flag (look IMX value description below)
7	MID1	Data set 1: MID value
8	PID1	Data set 1: PID value
…		Data sets: 2 – 19
63	IM20	Data set 20: IM flag (look IMX value description below)
64	MID20	Data set 20: MID value
65	PID20	Data set 20: PID value
66	PID194 IM	PID194 setup (look IMX value description below)
67	CS	RS-232 Message Checksum

IMX value (Binary)	Functionality
A 00000PIV	P = 0 –– page 1 variable P = 1 –– page 2 variable I = 0 –– do not send immediately to RS-232 bus I = 1 –– send immediately to RS-232 bus V = 0 – flag indicating that this data set does NOT contain valid MID/PID values to be used in Streamer filtering process V = 1 – flag indicating that this data set DOES contain valid MID/PID values to be used in Streamer filtering process

RS-232 message acknowledgement: <01><01><01><00><03>

4.2. IM/MID/PID Filter Setup Retrieval Command

Retrieves 20 sets of MID/PID/ Send Immediately setup information used to filter parameters flow from J1708 bus. One set of information will consist of MID value, PID value and corresponding Immediate Send information flag.

RS-232 message request format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID bytes 1 = J1708 command
4	09	Function code: 0x09 – IM/MID/PID filter setup retrieval
5	00	Number of data bytes in a message
6	0D	RS-232 Message Checksum

RS-232 message response format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	89	Function code: 0x89 – IM/MID/PID setup retrieval response
5	3D	Number of data bytes in a message (61 dec = 3D hex)
6	IM1	Data set 1: IMX flag (look IMX value description in filter setup)
7	MID1	Data set 1: MID value
8	PID1	Data set 1: PID value
…		Data sets: 2 – 19
63	IM20	Data set 20: IMX flag (look IMX value description in filter setup)
64	MID20	Data set 20: MID value
65	PID20	Data set 20: PID value
66	PID194 IM	PID 194 IM setup (look IMX value description in filter setup)
67	CS	RS-232 Message Checksum

4.3. Default MID/PID Filters

The following PIDs are factory configured to be part of the MID/PID filter list. All PIDs are from MID 128

- 51 – Throttle Position
- 84 – Road Speed
- 89 – Power Takeoff Status
- 91 – Percent Accelerator Pedal Position
- 96 – Fuel Level
- 133 – Average Fuel Rate
- 183 – Instantaneous Fuel Rate
- 184 – Instantaneous Fuel Economy
- 185 – Average Fuel Economy
- 190 – Engine Speed
- 235 – Total Idle Hours
- 236 – Total Idle Fuel Used
- 245 – Total Vehicle Distance
- 246 – Total Vehicle Hours
- 247 – Total Engine Hours
- 248 – Total PTO Hours

4.4. MID/PID Parameter Value Retrieval Command

This command requests stored internally MID/PID parameter values.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	03	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	14	Function code: 0x14 – retrieve MID/PID parameter value
5	1-21	Request value for MID/PID Data set (1-20) or MID/PID194 (21)
6	00	Number of data bytes in a message
7	CS	RS-232 Message Checksum

HDV Streamer will respond with following message:

Value Response - one byte PIDs:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	94	Function code: 0x94 – retrieve MID/PID parameter value
5	04	Number of data bytes in a message
6	MID	MID
7	PID	PID
8	ST	Status byte (see description below)
9	VALUE	PID parameter value
10	CS	RS-232 Message Checksum

Value Response - two byte PIDs:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	94	Function code: 0x94 – retrieve MID/PID parameter value
5	05	Number of data bytes in a message
6	MID	MID
7	PID	PID
8	ST	Status byte (see description below)
9	VALUE1	PID parameter value 1
10	VALUE2	PID parameter value 2
11	CS	RS-232 Message Checksum

Value Response - variable length PIDs:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	94	Function code: 0x94 – retrieve MID/PID parameter value
5	N	Number of data bytes in a message
6	MID	MID
7	PID	PID
8	ST	Status byte (see description below)
9	L	Length of PID variable
10	VALUE 1	PID parameter value 1
	VALUE …	
9 + L	VALUE L	PID parameter value L
10 + L	CS	RS-232 Message Checksum

Note: Status byte description

STATUS BYTE (binary)	Functionality
MY000PX0	P = 0 – page 1 variable P = 1 – page 2 variable Y = 0 – parameter value has NOT been broadcasted Y = 1 – parameter value has been broadcasted M = 0 – parameter value has NOT been modified since last read M = 1 – parameter value has been modified since last read X – used internally, do not use it

4.5. MID/PID Parameter Value Reporting for Send-Immediate

If the filter parameter (MID/PID combination) has been set to be sent immediately after reception, then data will be sent using the format as described above with the exception of function code (byte #4) being equal to hex 95.

4.6. Request Calculated Trip Odometer Command

This message will retrieve the Calculated Trip Odometer stored in the HDV Streamer. Trip Odometer is reported in 0.5 mile increments (Odometer = Data * 0.5).

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes = 2
3	01	ID byte: 1 = J1708 command
4	05	Function code: 05 – Retrieve Trip Odometer Reading
5	00	Number of data bytes = 0
6	09	RS-232 Message Checksum

Calculated Trip Odometer Value Response:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	85	Function code: 0x94 – retrieve MID/PID parameter value
5	04	Number of data bytes in a message
6	Data 1	Trip Odometer LSB
7	Data 2	Trip Odometer
8	Data 3	Trip Odometer
9	Data 4	Trip Odometer MSB
10	CS	RS-232 Message Checksum

4.7. Reset Calculated Trip Parameters Command

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes = 2
3	01	ID byte: 1 = J1708 command
4	06	Function code: 06 – Reset Trip Parameters
5	00	Number of data bytes = 0
6	0A	RS-232 Message Checksum

RS-232 message response: <01><01><01><00><03>

4.8. Start Polling Non-broadcast Parameters Command

This command starts broadcast requests for all valid MID/PID filter pairs that values have not been updated within specified time frame.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	04	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	X7	Function code: X7 – Polling time period base (see chart)
5	01	Fixed at 1
6	PP	Priority (see chart)
7	04	Number of Data bytes (Fixed at 4)
8	00	Dummy message contents: fill data with 0s
9	00	Dummy message contents: fill data with 0s
10	00	Dummy message contents: fill data with 0s
11	00	Dummy message contents: fill data with 0s
12	CS	Message Checksum

Polling time period base

Byte value (hex)	Time value (seconds)
07	Stop broadcast
17	0.5
27	1.0
37	1.5
47	2.0
57	5.0

Broadcast priority values

Byte Value (hex)	Priority
01	1
02	2
04	3
08	4
10	5
20	6
40	7
80	8

RS-232 message response: <01><01><01><00><03>

4.9. Stop Polling Non-broadcast Parameters Command

This command stops all broadcast requests for all MID/PID filter pairs.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	07	Function code: $07 Stop Polling Non-broadcast Parameters
5	00	Data length
6	CS	Message Checksum

RS-232 message response: <01><01><01><00><03>

4.10. Transmit J1708 Data to Vehicle Command

The Streamer supports standard J1708 commands of up to 21 bytes. Additionally the Streamer can send messages up to 100 bytes to the J1708 bus. The format to send information to the J1708 bus is shown below:

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	03	Number of control bytes
3	01	ID byte: 1 = J1708 command
4	12	Function code: 0x12 - transmit to J1708
5	02	Priority byte 00000010 set bit 0 to 7 see below
6	NDB	Number of Data Bytes (0 to 100)
7 to n-1		J1708 Message to transmit to vehicle (Up to 100 bytes)
n	CS	Message Checksum

J1708 message priority settings:

Bit Number	Byte Value (hex)	Priority value
0	01	1
1	02	2
2	04	3
3	08	4
4	10	5
5	20	6
6	40	7
7	80	8

After sending a transmit message to the Streamer you will get the confirmation message if the checksum is correct. (RS-232 message response: <01><01><01><00><03>)

A transmit complete message is sent to the PC after the device successfully transmits the message to the J1708 bus. A second message should not be sent to the device before confirming that the first message has been sent. The transmit confirmation is formatted as follows:

RS-232 Transmit confirmation message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	09	ID byte: 9 = transmit OK
4	00	Data length
5	CS	Message Checksum

4.11. Unique Identifier Setup Command

This command sets up and stores in non-volatile memory **Unique Identifier**.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	08	ID byte: 8 = General Configuration command
4	12	Function code: 0x12 set Unique Identifier
5	0A	Number of data bytes in a message
6	C1	Character #1 of Unique Identifier
7	C2	#2
8	C3	#3
9	C4	#4
10	C5	#5
11	C6	#6
12	C7	#7
13	C8	#8
14	C9	#9
15	C10	#10
16	CS	RS-232 Message Checksum

RS-232 message response: <01><01><08><00><0A>

4.12. Unique Identifier Retrieval Command

This command retrieves **Unique Identifier**.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	08	ID byte: 8 = General Configuration command
4	13	Function code: 0x13 retrieve Unique Identifier
5	00	Number of data bytes in a message
6	1E	RS-232 Message Checksum

RS-232 response message

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	02	Number of control bytes
3	08	ID byte: 8 = General Configuration command
4	93	Function code: 0x93 retrieve Unique Identifier
5	0A	Number of data bytes in a message
6	C1	Character #1 of Unique Identifier
7	C2	#2
8	C3	#3
9	C4	#4
10	C5	#5
11	C6	#6
12	C7	#7
13	C8	#8
14	C9	#9
15	C10	#10
16	CS	RS-232 Message Checksum

4.13. Firmware Version Retrieval Command

This command retrieves firmware version.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	05	Function code: 0x05 Retrieve Firmware version
4	00	Number of data bytes in a message
5	07	RS-232 Message Checksum

RS-232 response message

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	85	Function code: 0x85 retrieve Firmware version
4	03	Number of data bytes in a message
5	VER1	Character #1 of Firmware version
6	VER2	Character #2 of Firmware version
7	VER3	Character #3 of Firmware version
8	CS	RS-232 Message Checksum

4.14. Model Number Retrieval Command

This command retrieves Model Number.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	02	Function code: 0x02 retrieve Model Number
4	00	Number of data bytes in a message
5	04	RS-232 Message Checksum

RS-232 response message

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	01	Number of control bytes
3	82	Function code: 0x82 Retrieve Model Number
4	07	Number of data bytes in a message
5	48	Character #1 of Model Number: H
6	44	Character #2 of Model Number: D
7	56	Character #3 of Model Number: V
8	44	Character #4 of Model Number: D
9	53	Character #5 of Model Number: S
10	2D	Character #6 of Model Number: -
11	53	Character #7 of Model Number: S
8	CS	RS-232 Message Checksum

4.15. RS-232 Parameters Setup Command

This command sets up all RS-232 communication parameters. Parameters are stored into non-volatile memory and restored on power up. If information in non-volatile memory is incorrect (or never written to) baud rate will default to 9600. Allowed baud rates are listed in a chart below. If a command is sent with incorrect baud rate then error message will be returned.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	05	Number of control bytes
3	08	ID byte: 8 = General Configuration command
4	01	Function code: 1 – baud rate change
5	03	UB3 (see chart below)
6	Baud 1	Baud rate divisor LSB (see chart below)
7	Baud 2	Baud rate divisor MSB (see chart below)
8	00	Number of data bytes
9	CS	Message Checksum

Setting UB3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit1	Bit 0
Divisor Latch	Set break	Set Parity	Even Parity	Parity Enable	Stop Bits	Word Length 1	Word Length 2

Bit 7 = 1 allows baud rate divisor to be changed. 0 baud rate change is blocked. Bit 7 should not be set. The setting and resetting of this bit is handled internally.

Bit 6 = 1 forces the transmitter output to a logic 0 for alerting the remote receiver to a line break. 0 indicates no break condition.

Bit 5, 4 and 3 work together to set parity

Bit 5	Bit 4	Bit 3	Parity Selection
*	*	0	No parity (Default)
0	0	1	Odd parity
0	1	1	Even parity
1	0	1	Force parity “1”
1	1	1	Force parity “0”

Bit 2, stop bit works in conjunction with bits 1 and bit 0

Bit 2	Word Length (bits)	Stop Bit Length
0	5,6,7,8	1 (Default)
1	5	1 1/2
1	6,7,8	2

Bits 1 and 0 set the word length

Bit 0	Bit 1	Word Length
0	0	5
0	1	6
1	0	7
1	1	8 (Default)

Default value: The HDV Streamer should always be set to 8-bit word length.

RS-232 message response: <01><01><08><00><0A>

Setting Baud 1, Baud 2

Baud Rate Setup Table

Baud Rate	Decimal Divisor	Hex Divisor LSB (Baud 1)	Hex Divisor MSB (Baud 2)
9600 (Default)	48	$30	$00
14,400	32	$20	$00
19,200	24	$18	$00
38,400	12	$0C	$00
57,600	8	$08	$00
115,200	4	$04	$00

All baud rates are entered as hex numbers. To calculate your own decimal divisor, use the following formula. The decimal number must be converted to hexadecimal before entering the number.

The UART clock = 7,372,800 Hz

$$\text{Decimal Divisor} = \frac{\text{Clock frequency (7,372,800)}}{\text{Baud output x 16}}$$

If you want a baud rate of 9,600 baud

$$\text{Decimal Divisor} = \frac{7,372,800}{9,600 \times 16}$$

Decimal Divisor = 48 = 30 hex

Baud 1= 30
Baud 2= 00

4.16. Firmware Update Command

There are two parts of the firmware that can be upgraded, main code, and boot loader code. The first is the main code firmware that handles all of the functionality of the adapter. The other section is the boot loader code, which handles the upgrade of the main code firmware. When the main code firmware is upgraded the boot loader code does not get overwritten. Therefore if the main code is bad, the user will always be able to download a new main code. When the boot loader code is upgraded, first a copy of the new boot loader code is placed on the adapter. If the copy is completed with no errors, this copy will then overwrite the existing boot loader code. If power is removed during the overwrite process, the new boot loader copy will still be in the adapter and the main code will know that there is a newer version of the boot loader code and to start the overwrite process.

RS-232 message format:

Byte Number	Byte Value (hex)	Function
1	01	Start of frame
2	03	Number of control bytes
3	08	ID byte: 8 = General Configuration command
4	05	Function code: firmware upgrade
5	X	X=1 – upgrade main code firmware X=2 – upgrade boot loader code firmware
5	00	Data length
6	CS	RS-232 Message Checksum

RS-232 message response: <01><01><08><00><0A>