USB-ASC232

ASCII RS-232 Controlled USB Keyboard and Mouse Cable

User Manual

Thank you for purchasing the model USB-ASC232 Cable

HAGSTROM ELECTRONICS, INC.

is pleased that you have selected this product for your application. This unit is may be used a variety of ways in order to meet your specific requirements. Please take a few minutes to read this manual before using your USB-ASC232.

If you have any questions about the use of the USB-ASC232 not covered in this manual, please contact us directly. We offer toll free technical product support from 8:00am to 5:00pm M-F Eastern Time ***888-690-9080***. You may also send an email to

sales@hagstromelectronics.com

We respond to all email requests within one business day.

CONTENTS

Questions or Comments?
Please give us a call!

Toll Free
888-690-9080
or
www.hagstromelectronics.com
email: **sales@hagstromelectronics.com**

Introduction to the USB-ASC232

The USB-ASC232 Keyboard and Mouse Emulator is a product designed to allow RS-232 serial communication to produce USB keystroke and mouse actions on a target computer.

The USB end of this device can be connected to any USB port that supports a standard USB Keyboard and Mouse. No special drivers are required.

The USB-ASC232 is configurable to allow for different standard BAUD rates, stop bits, parity, handshaking, and mode of translation. Use the supplied "USBASC232.EXE" program to configure the unit.

Translation Modes of the USB-ASC232

There are three specific translation modes in which the USB-ASC232 can be programmed to operate. Incoming RS-232 data can be translated from ASCII Mode, Extended ASCII Mode, or Key Number Mode. Each of these modes vary in the way incoming characters are treated and their translation into USB keystrokes.

ASCII MODE (Default Mode)

The ASCII mode is the default factory setting for the mode of operation. In this mode, printable ASCII characters received, which are each a one byte value in the range of 0x00 to 0x7F (0 to 127 decimal), will generate their corresponding USB keystroke on the computer where the USB-ASC232 USB plug is connected. See the ASCII table on the next page for a list of the characters which are recognized and translated for the cable's ASCII mode of operation.

Example: If a one byte value of 0x41 (decimal 65) is received in this mode, a capital "A" character will be produced as a keystroke on the computer at the USB end of the cable.

RS-232 characters received which are out of the 0x00 to 0x7f range will be ignored in this mode.

ASCII Mode Translation Table

The ASCII mode table below lists the standard ASCII characters that will produce a corresponding USB keystroke when received by the USB-ASC232.

Value		Character	Value		Character	Value		Character	Value		Character
Dec	Hex		Dec	Hex		Dec	Hex		Dec	Hex	
00	00	none	32	20	Space	64	40	@	96	60	`
01	01	none	33	21	!	65	41	A	97	61	a
02	02	none	34	22	"	66	42	B	98	62	b
03	03	none	35	23	#	67	43	C	99	63	c
04	04	none	36	24	$	68	44	D	100	64	d
05	05	none	37	25	%	69	45	E	101	65	e
06	06	none	38	26	&	70	46	F	102	66	f
07	07	none	39	27	'	71	47	G	103	67	g
08	08	Backspace	40	28	(	72	48	H	104	68	h
09	09	Tab	41	29	)	73	49	I	105	69	i
10	0A	none	42	2A	*	74	4A	J	106	6A	j
11	0B	none	43	2B	+	75	4B	K	107	6B	k
12	0C	none	44	2C	,	76	4C	L	108	6C	l
13	0D	Return	45	2D	_	77	4D	M	109	6D	m
14	0E	none	46	2E	.	78	4E	N	110	6E	n
15	0F	none	47	2F	/	79	4F	O	111	6F	o
16	10	none	48	30	0	80	50	P	112	70	p
17	11	none	49	31	1	81	51	Q	113	71	q
18	12	none	50	32	2	82	52	R	114	72	r
19	13	none	51	33	3	83	53	S	115	73	s
20	14	none	52	34	4	84	54	T	116	74	t
21	15	none	53	35	5	85	55	U	117	75	u
22	16	none	54	36	6	86	56	V	118	76	v
23	17	none	55	37	7	87	57	W	119	77	w
24	18	none	56	38	8	88	58	X	120	78	x
25	19	none	57	39	9	89	59	Y	121	79	y
26	1A	none	58	3A	:	90	5A	Z	122	7A	z
27	1B	Esc	59	3B	;	91	5B	[	123	7B	{
28	1C	none	60	3C	<	92	5C	\	124	7C	|
29	1D	none	61	3D	=	93	5D	]	125	7D	}
30	1E	none	62	3E	>	94	5E	^	126	7E	~
31	1F	none	63	3F	?	95	5F	-	127	7F	none

Extended ASCII Mode

The USB-ASC232 Extended ASCII Mode of operation allows for the standard ASCII character set (character values 0x00 to 0x7F), in addition to an extended ASCII character set (character values 0x80 to 0xFF). When an RS-232 byte is received in this mode, it produces a USB keystroke on the PC which corresponds to the following table.

When using the Extended ASCII mode, characters are received which are in the range of value from 0x00 to 0xFF (0 to 255 decimal). The corresponding USB keystroke from the table will be generated.

Extended ASCII Mode allows for the ***ANSI/ISO Latin-1*** character group in the range of 0x80 to 0xFF (128 to 255 decimal) to be produced as well as the standard 0x00 through 0x7F ASCII codes. Additionally in this mode, the F1-F10 keys are supported for the character range of codes 0x11 to 0x1A values respectively. F11 and F12 keys are sent for received characters of 0x0E and 0x0F in the Extended ASCII mode.

Example 1: If the value 0x41 (decimal 65) is received in this mode, a capital "A" character will be produced on the target computer at the USB end of the cable.

Example 2: If the value 0xA9 (decimal 169) is received in this mode, a "©" character will be produced on the target computer at the USB end of the cable.

Key Number Mode

The Key Number Mode provides users with complete control of the generation of the make (activation) and break (deactivation) of any standard keyboard key. In this mode, a single byte will command the make or break of a specific keyboard key at the target computer. Using this mode allows for generation of any keystroke or combination of keystrokes on the target computer.

In general, to make a key (generate a press of a specific key), a one byte value between 0x00 and 0x7F is sent to the USB-ASC232. Once received, the key specified will be seen as held down on the computer just as if someone was physically holding that key on a keyboard.

Each time a make is sent for a key, a corresponding break (release) of that key must be done at a later time to deactivate it. The break code for a key is the same value as the make code plus 0x80 (128 decimal). The break code releases the key that was activated earlier by a make code. See the following table for make and break codes.

US Key Number Table (Decimal Values)

An international key number table is available on the CD provided with the USB-ASC232.

Key	Make	Break
`~	01	129
1!	02	130
2@	03	131
3#	04	132
4$	05	133
5%	06	134
6^	07	135
7&	08	136
8*	09	137
9(	10	138
0)	11	139
-_	12	140
=+	13	141
BS	15	143
Tab	16	144
Q	17	145
W	18	146
E	19	147
R	20	148
T	21	149
Y	22	150
U	23	151
I	24	152
O	25	153
P	26	154
[{	27	155
]}	28	156
|	29	157
Caps	30	158
A	31	159
S	32	160
D	33	161
F	34	162
G	35	163
H	36	164

Key	Make	Break
J	37	165
K	38	166
L	39	167
;:	40	168
‘“	41	169
Enter	43	171
L Shift	44	172
Z	46	174
X	47	175
C	48	176
V	49	177
B	50	178
N	51	179
M	52	180
,<	53	181
.>	54	182
/?	55	183
R Shift	57	185
L Ctrl	58	186
L Alt	60	188
Space	61	189
R Alt	62	190
R Ctrl	64	192
L Win	70	198
R Win	71	199
Win APL	72	200
Insert	75	203
Delete	76	204
L Arrow	79	207
Home	80	208
End	81	209
Up Arrow	83	211
Dn Arrow	84	212
Page Up	85	213
Page Dn	86	214

Key	Make	Break
R Arrow	89	217
NumLock	90	218
7 (Num)	91	219
4 (Num)	92	220
1 (Num)	93	221
/ (Num)	95	223
8 (Num)	96	224
5 (Num)	97	225
2 (Num)	98	226
0 (Num)	99	227
* (Num)	100	228
9 (Num)	101	229
6 (Num)	102	230
3 (Num)	103	231
. (Num)	104	232
- (Num)	105	233
+ (Num)	106	234
Enter (Num)	108	236
Esc	110	238
F1	112	240
F2	113	241
F3	114	242
F4	115	243
F5	116	244
F6	117	245
F7	118	246
F8	119	247
F9	120	248
F10	121	249
F11	122	250
F12	123	251
Prt Scr	124	252
Scrl Lk	125	253
Pause/Break	126	254

When sending data to the USB-ASC232 to generate keystrokes in Key Number Mode, use the values shown above to produce the “make” and “break” actions for the corresponding key.

By using the make and break commands in Key Number Mode, any sequence that can by manually typed on a keyboard can be produced with the USB-ASC232. Use this Key Number Mode to emulate single keystrokes or even combinations such as Ctrl+F1, or Shift-Alt+F5.

Additional Key Number Mode Control Commands

The USB-ASC232 features two additional commands for keyboard action. The first command provides a way to clear the keyboard buffer and is useful for ensuring that no keys are stuck in the "ON" state. The second command allows for the polling of the keyboard status LED states. This polling command is useful for checking shift case changes, or for making sure of the Num Lock state before using Num Lock affected keys.

0x38 - USB Buffer Clear Command. Sending 0x38 to the USB-ASC232 serial port results in the device's internal USB keyboard buffer being cleared. Use of this command ensures that all made keys currently in the keyboard buffer are released. The USB-ASC232 will respond to this command with 0xC7, which is the one's complement of the command.

0x7F - Status LED Read Command. Sending a code 0x7F to the USB-ASC232 serial port results in the return of a character in the range of ASCII "0" - "7". The USB-ASC232 response character reflects the current state of the Scroll Lock, Caps Lock, and Num Lock LEDs on the computer system as listed in the chart below.

ASCII Response	Scroll Lock Status	Caps Lock Status	Num Lock Status
"0"	Off	Off	Off
"1"	Off	Off	On
"2"	Off	On	Off
"3'	Off	On	On
"4"	On	Off	Off
"5"	On	Off	On
"6"	On	On	Off
"7"	On	On	On

Key Number Mode Examples

Generation of keystrokes on the target computer is done through the sending of special one byte codes in the Key Number Mode.

Each standard key of the PC keyboard is assigned a "make" code to emulate the press of a key, and a "break" code, which results in the release of the key. The Key Number Table on the page 7 lists each of the supported keys and their corresponding make and break codes.

From the keycode table, the value of 02 (0x02) can be sent to the unit to generate the "make" or press and hold of the "1" key. Sending a byte with a value of 130 (0x82) will result in the release of the "1" key.

Important: For any key that has been previously sent a "make" code, a "break" code of that key must be sent at a later time. Failure to send the corresponding break code will leave the key in the down state on the target computer, which may result in unintended keystrokes when new commands are sent. The last key left in the make state without a break will repeat until the break code is sent. Never command more than 60 keys in the "make" state at the same time as this will exceed the standard USB protocol keyboard buffer length for the USB-ASC232 device.

The use of make and break codes allows the user to create virtually any keystroke combination. For example, if a sequence of CTRL+ALT+F1 was needed, the following codes would be sent, **58** (L Ctrl make), **60** (L Alt make), **112** (Make F1), **240** (Break F1), **188** (Break L Alt), **186** (Break L Ctrl).

The example above assumes that the handshaking USB-ASC232 is seen as not busy before sending the next command byte to the unit.

Producing Mouse Action on the Target PC

The USB-ASC232 is capable of controlling the mouse cursor and mouse button states through a special four byte command sequence. This command sequence can be used in any of the operating modes of the USB-ASC232.

The four byte mouse control sequence is construced as follows:

Byte #1	Byte #2	Byte #3	Byte #4
[Command]	[X Movement]	[Y Movement]	[Scroll/Buttons]

Byte #1 - Always a 0x00 value, indicates start of packet.

Byte #2 - Signed byte for the magnitude of X direction movement. Values 0x01 to 0x7F move the cursor right, values 0xFF to 0x81 move it left. A 0x00 Value produces no X direction movement.

Byte #3 - Signed byte for the magnitude of Y direction movement. Values 0x01 to 0x7F move the cursor down, values 0xFF to 0x81 move the cursor up. A 0x00 produces no Y movement.

Byte #4 - Byte for Scroll Wheel movement and mouse button control. The upper 4 bits of this byte contain a signed value for scroll wheel movement, while the lower 3 bits command the mouse Left, Right, and Middle button activation and deactivation.

This Byte #4 is constructed as shown,

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
[Ws]	[W2]	[W1]	[W0]	[1]	[MM]	[MR]	[ML]

[Ws] - The sign of the scroll wheel movement. A zero in this bit commands up movement on the scroll wheel, a one in this bit produces down scroll wheel movement. Movement magnitude based on the signed value in Ws through W0 values.

[W2] - The most significant bit of the scroll wheel movement magnitude.

[W1] - Bit 1 of the three bit value for scroll wheel movement.

[W0] - Least significant bit of the scroll wheel movement value.

[1] - Bit 3 of this #4 byte is always "1". Sending "0" for this bit will cause the entire mouse control packet to be ignored.

[MM] - State of the middle mouse button. Set to 1 for middle mouse button on, 0 for the button to be off.

[MR] - State of the right mouse button. Set to 1 for right mouse button on, 0 for the button to be off.

[ML] - State of the left mouse button. Set to 1 for left mouse button on, 0 for the button to be off.

NOTES - When sending this 4 byte mouse command packet, the three bytes following the first 0x00 byte should be sent with no more than 50msec between the bytes. The 0x00 must always be followed by 3 additional bytes to complete the mouse control packet and avoid any confusion between mouse control and keystroke commands.

The scroll wheel magnitude command is a 4 bit signed value located in bits Ws through W0, and in the range of 0x00 to 0x07 for up movement, 0x0F to 0x08 for down movement.

The three mouse button bits command the states of the mouse buttons as seen by the target computer. A "1" in the respective bit indicates the mouse button is on, while a "0" indicates the button is not on. Be sure to release any mouse buttons that were in the on state at a later time once the mouse button task has completed.

When sending a mouse control packet for cursor movement only, be sure to send all "0" values for the scroll wheel and mouse button bits, if those options are not being used for mouse actions at that time.

Mouse Packet Examples

The packet below is an example of a packet for the USB-ASC232 to command X movement of 15 to the right, and a Y movement of 5 in the up direction.

0x00, 0x0F, 0xFB, 0x08

The movement values shown above do not necesarily coordinate to a specific number of pixels moved as the computer mouse settings for speed and acceleration determine the actual cursor movement for the target computer. The movement of the mouse cursor for the values sent will be repeatable for a specifc computer.

All mouse cusor control movements function just as a standard mouse in that all movement is relative from the current cursor position. The USB-ASC232 does not send or receive any specific screen coordinates from the computer.

To position the mouse cursor to a specific point on the computer screen, first send enough movement to put the cursor into a known position, such as the upper left of the screen, and then use additional mouse movement commands to get the cursor to the desired location. The values used to get the cursor to the desired screen position from the known (upper left in this case) position will be repeatable each time for a specific computer and other computers with the exact same screen settings and mouse settings.

The following packet can be used to send the scroll wheel command of up one position to the computer.

How far the scroll moves on the computer screen depends on the scroll system settings of the target computer.

0x00, 0x00, 0x00, 0x18

The next example four RS-232 mouse packets are sent to emulate a double left mouse click on the target computer.

Packet 1 - 0x00, 0x00, 0x00, 0x09 (left mouse button on)
Now delay around 150 msec for the system to see the button

Packet 2 - 0x00, 0x00, 0x00, 0x08 (left mouse button off)
Delay 150 msec again for the system to see the button release

Packet 3 - 0x00, 0x00, 0x00, 0x09 (left mouse button on)
Delay around 150 msec for the system to see the button

Packet 4 - 0x00, 0x00, 0x00, 0x08 (left mouse button off)

Note that cursor movement, scroll wheel movement and button actions may all be implemented within the same packet. The examples above focus only on a particular action at a time for clarity. For instance, a command that sends both X and Y cusor movement can also send button information and/or scroll wheel movement.

Sending Serial Control Commands

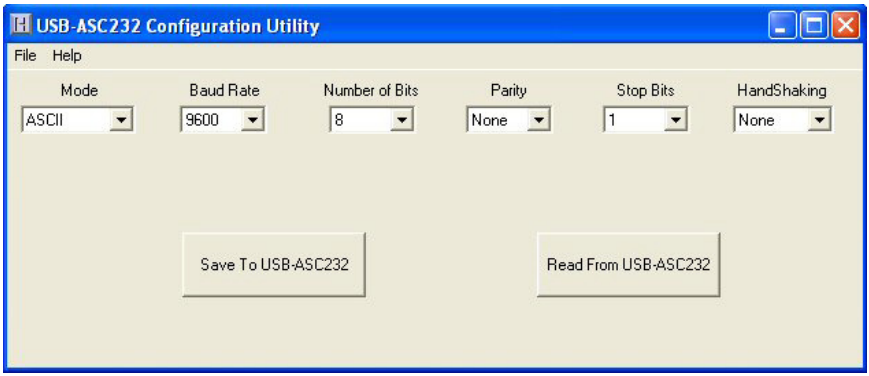
The CD included with the unit contains source code examples in various programming languages to help the user get started on their own Com Port control software. Use these examples as a starting point for programs which run the communication to the USB-ASC232 from the computer attached to the USB-ASC232 Com Port DB9 connector.

USB-ASC232 Configuration Program

The USB-ASC232 has programmable settings for adapting the interface to the user's application.

A program, "USBASC232.EXE" is included on the supplied CD. Copy the entire contents of the CD into a folder on the (USB end) target computer's hard disk and run the .exe file from that folder.

There are a variety of protocol settings which can be selected from the USBASC232.EXE configuration utility.



To create a configuration using the utility, first select the "Mode" menu and choose ASCII, Extended ASCII, or Key Number Mode based on the desired conversion mode.

Next, set the RS-232 communication parameters for the Baud Rate, Number of Bits in the character, Parity, Number of Stop Bits, and Handshaking mode.

The standard BAUD rates available are 2400, 4800, 9600, 14400, 19200, and 38400.

The number of bits in the character sent to the USB-ASC232 may be 7 bits or 8 bits. NOTE: 8 bit characters are required for Extended ASCII and Key Number modes.

Parity may be set to “None”, “Odd”, or “Even”.

The number of stop bits may be set to 1 or 2 bits.

Handshaking may be set to “None”, “RTS/CTS”, or “Echo”.

When the handshaking is set to “None”, the computer sending the information must be careful not to overflow the receiving buffer of the USB-ASC232. The best way to do avoid the buffer overflow in this mode is to delay 50msec after each byte is sent to the USB-ASC232.

With handshaking set to “RTS/CTS”, the hardware controls the flow of data between the computer sending the RS-232 data and the USB-ASC232. This hardware handshaking prevents overflow of the receiving buffer on the USB-ASC232 unit. When “RTS/CTS” is used for handshaking, the DTR options for the serial port being used on the computer sending the RTS-232 should be disabled.

The “Echo” setting means that the USB-ASC232 will send a one’s complement response byte to each command. For instance, if a value of 0x32 is sent to the USB-ASC232 serial port, it will reply with a value of 0xCD in response. Use this handshaking mode as a way to not only keep the buffer from overflowing, but is also a confirmation of the value of the code that was received.

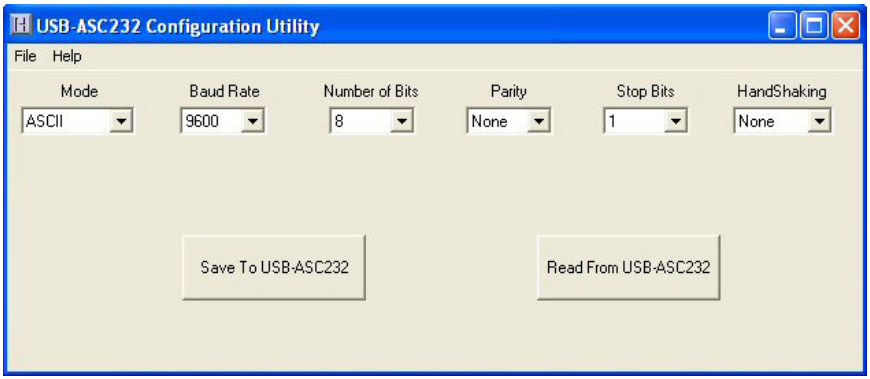
Note that in the “Echo” handshake mode, the response to the 0x7F request for LED status in Key Number Mode will be replied with the status byte as outlined on page 8.

In “Echo” mode, the individual bytes of the mouse control packet do not generate a reply. Only after the 4th and last byte of the mouse packet has been received is an echo value of 0xFF returned to indicate reception of the mouse packet.

Saving the Configuration to the USB-ASC232

Once the desired parameters have been set on the USBASC232 Configuration Utility Screen, it must be saved to the USB-ASC232 unit to begin operation according to those settings.

Select the button "Save To USBASC232" to write the configuration to the unit. Once the configuration has been written, the USB-ASC232 will begin immediately operating according to the loaded parameters.



The configuration may also be read from the USB-ASC232 unit by selecting the button "Read From USB-ASC232". The configuration in the USB-ASC232 attached to the computer's USB port will be read into the configuration parameters on the screen.

Once a configuration has been created, it is recommended that it be stored on the computer so that it may be recalled at a later time. Use the "File" menu to perform saving and opening of configurations.

Saving the configuration to a file on the computer provides an easy way to recall the same configuration to save into additional USB-ASC232 units.

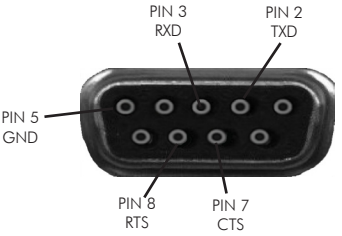
Custom USB-ASC232 Options

We offer special modifications to our standard USB-ASC232 unit to conform to your exact specifications. Potential modifications include, but are not limited to: special cable lengths, different gender DB9, RS-422, etc. Let us know if you have special requirements. Please call or email with your specific custom needs.

Serial Connector Pinout

The USB-ASC232 serial connection is a 9 pin female D type connector. The pinout is designed such that it will attach directly to a standard PC com port.

Pin 1	=	Unused
Pin 2	=	Serial Data out of USB-ASC232
Pin 3	=	Serial Data into USB-ASC232
Pin 4	=	Unused
Pin 5	=	Logic Ground
Pin 6	=	Unused
Pin 7	=	CTS into USB-ASC232
Pin 8	=	RTS out of USB-ASC232
Pin 9	=	Unused



Looking into connector face

USB-ASC232 Specifications

Operating Voltage	5 Volts DC +/- 5% (Powered from USB port)
Operating Current	Less than 100 ma
Operating Temp.	0 to 70 Degrees C
PC Interface	USB - Target Computer RS-232 - Serial Source
Cable Length	6 feet

Warranty

HAGSTROM ELECTRONICS, INC. warrants this product against defects in material or workmanship for a period of ONE YEAR from the original purchase date. We will repair or replace (at our option) the returned defective unit at no charge during this warranty period.

No responsibility is assumed for any special, incidental, or consequential damage resulting from the use of or inability to use this product. In no case is **HAGSTROM ELECTRONICS, INC.** to be liable for any amount which exceeds the purchase price of the unit, regardless of the claim.

No other warranty, written or verbal, is authorized. This warranty is applicable only to units sold in the United States. Units sold outside the United States are covered by a similar warranty.

Depending on the state in which you live, you may have additional rights.

Great care has been taken during the assembly, testing, and burn-in of your USB-ASC232 to ensure its performance. If you have any questions, please send us an email or give us a call. Support is available Monday through Friday, 8:00 am to 5:00 pm (EST).

customer service email*:* ***sales@hagstromelectronics.com***

Call Toll Free **888-690-9080**, or **(540) 465-4677**

NOTICE: The USB-ASC232 product is designed to be used by technically oriented computer users.

HAGSTROM
ELECTRONICS, INC.

Toll Free 888-690-9080
Phone: **(540) 465-4677** Fax: **(540) 465-4678**
Monday through Friday, 8:00 am to 5:00 pm (EST)

sales@hagstromelectronics.com
www.hagstromelectronics.com
1986 Junction Road, Strasburg, VA 22657

V. 04.20