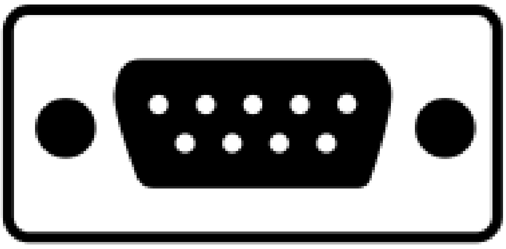
RS232 PROTOCOL



SUMMARY

INTRODUCTION

RS-232 is a protocol that standardizes a serial type communication channel. Available on almost all PCs from 1981 until the mid-2000s, it is commonly referred as the "serial port". On MS-DOS and Windows operating systems, RS-232 ports are referred to as COM1, COM2, etc. This has earned them the nickname "COM ports".

WHAT IS RS232

RS232 defines a protocol that details how a stream of data bits is sequentially transmitted onto a wire i.e. a bit stream or byte stream. The order and meaning of each bit is defined by the protocol.
RS232 is a serial information transfer protocol standard that defines both the protocol (method of transmission of data) and the physical hardware to do it.

It is a serial transmission method of transferring data across a single wire, data is only transmitted in one direction for each wire so for bi-directional communication (two directions) you need two wires.

RS232 is an asynchronous communication protocol as there is no clock transmitted at all between the receiver and the transmitter.

Technical data:

Basically, it can transfer a single byte of data over a serial cable having between 3 to 22 signal wires and running at speeds from 100 to 20k baud. Common baud rates used are 2.4k, 9.6k, 19.2k, the cable length can be up to 15 meters.

To transfer a block of data individual bytes are transmitted one after another.

RS232 PROTOCOL

Data is transmitted serially in one direction over a pair of wires. Data going out is labelled Tx (indicating transmission) while data coming in is labelled Rx (indicating reception). To create a two-way communication system a minimum of three wires are needed Tx, Rx and GND (ground). Crossing over Tx & Rx between the two systems lets each unit talk to the opposite one.
Each byte can be transmitted at any time in a sequence.

To establish effective communication via RS-232, it is necessary to define the protocol used and the data sequence used.

BAUD RATE

The baud rate is simply the transmission speed measured in bits per second. It defines the frequency of each bit period.
For a baud rate of 2400 (2400 bps) the frequency is 2400Hz and the bit period is 1/2400 or 416.6us. This is the information that a receiver uses to recover the bits from the data stream.

RECEIVER THRESHOLD VOLTAGE LEVELS

At the receiver the input the minimum voltage levels are defined as ±3V and can reach up to ±25V. i.e.:
To receive a logic 0 the voltage must be greater than 3V.
To receive a logic 1 the voltage must be smaller than -3V.
This allows for losses as the signal travels down the cable and provides noise immunity i.e. any spurious noise up to a level of ±3V can be tolerated without it having any effect on the receiver and the data.

START BIT

At the beginning of each transmission a start bit is transmitted indicating to the receiver that a byte of data is about to follow.
The start bit lets the receiver synchronize to the data bits since it can see the rising edge of the signal on the line.

Once the start bit is found, the receiver knows where the following bits will be as it is given the sample period (derived from the baud rate) as part of the initialization process.

This is why you must set the same settings in both devices under communication. i.e. baud rate, number of stop bits, number of data bits, and parity bit (on or off).

DATA BITS

Data bits follow the start bit. There will usually be seven or eight data bits with the LSB (least significant bit) transmitted first.
The reason you can choose between seven or eight is that ASCII is made up of the alphabet within the first seven bits (as well as the control characters).
The eighth bit extends the character set for graphical symbols.

Other data bit sizes are 5, 6, 8, and 9 bits. However, bit length is usually set to 8 bits - this is very commonly used.

THE PARITY BIT

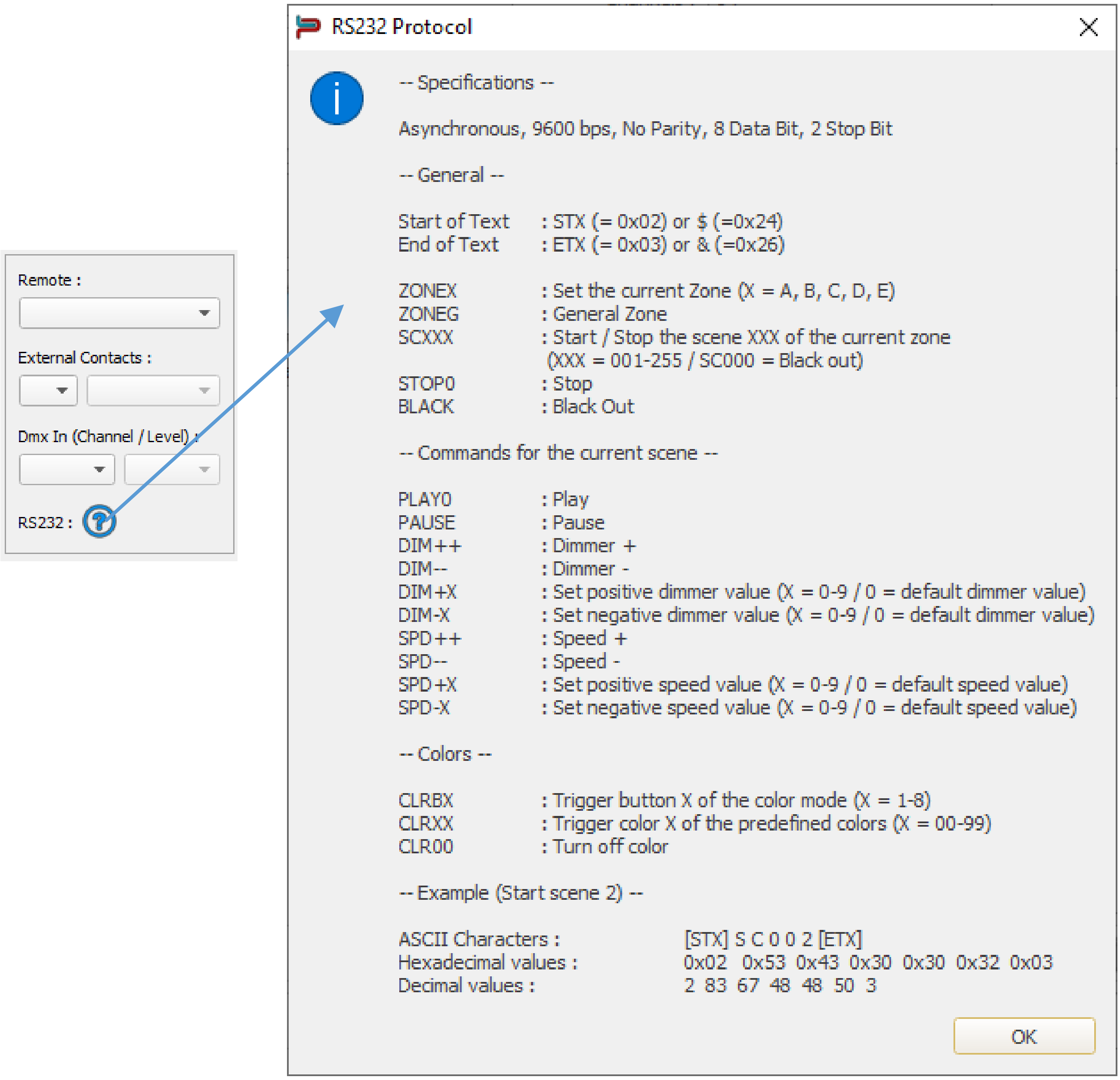
The RS232 parity bit is an error detection mechanism. You can use either odd parity or even parity or none at all.
At the receiver the parity bit is used to tell if an error occurred during transmission.
Even parity: the bit added to the data is positioned in such a way that the number of states 1 is even on the given set + parity bit
Odd parity: the bit added to the data is positioned in such a way that the number of states 1 is odd on the given set + parity bit

THE STOP BIT

The RS232 stop bit merely gives a period of time before the next start bit can be transmitted. It is the opposite sense to the start bit and because of this allows the start bit to be seen then.
If there was no stop bit then the last bit in the data stream would be the parity bit (or data bit if parity is not active).
The stop bit can be set choosing from 1, 1.5, or 2-bit periods.

HOW TO USE RS232 WITH OUR INTERFACES

The Standalone mode allows to use the RS232 protocol as receiver to control the DMX interface via another device with the commands describe in the help topic of the software.



-- Specifications --

Asynchronous, 9600 bps, No Parity, 8 Data Bit, 2 Stop Bit

-- General --

Start of Text	: STX (= 0x02) or $ (=0x24)
End of Text	: ETX (= 0x03) or & (=0x26)
ZONEX	: Set the current Zone (X = A, B, C, D, E)
ZONEG	: General Zone
SCXXX	: Start / Stop the scene XXX of the current zone (XXX = 001-255 / SC000 = Black out)
STOP0	: Stop
BLACK	: Black Out

-- Commands for the current scene --

PLAY0	: Play
PAUSE	: Pause
DIM++	: Dimmer +
DIM--	: Dimmer -
DIM+X	: Set positive dimmer value (X = 0-9 / 0 = default dimmer value)
DIM-X	: Set negative dimmer value (X = 0-9 / 0 = default dimmer value)
SPD++	: Speed +
SPD--	: Speed -
SPD+X	: Set positive speed value (X = 0-9 / 0 = default speed value)
SPD-X	: Set negative speed value (X = 0-9 / 0 = default speed value)

-- Colors --

CLRBX	: Trigger button X of the color mode (X = 1-8)
CLRXX	: Trigger color X of the predefined colors (X = 00-99)
CLR00	: Turn off color

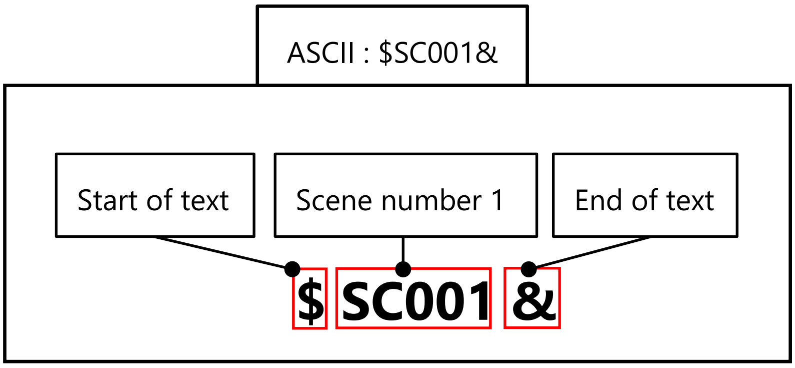
-- Example (Start scene 2) --

ASCII Characters :	[STX] S C 0 0 2 [ETX]
Hexadecimal values :	0x02 0x53 0x43 0x30 0x30 0x32 0x03
Decimal values :	2 83 67 48 48 50 3

Connect the RS232 transmitter to the interface RS232 (Tx, Rx and GND pins) and send the dedicated ASCII commands lines that you need.

Setup the good settings to the RS232 transmitter: Asynchronous, 9600 bps, No Parity, 8 Data Bit, 2 Stop Bit.
The ASCII commands need to be sent one time only to be processed by the interface.

EXAMPLE: START SCENE 1



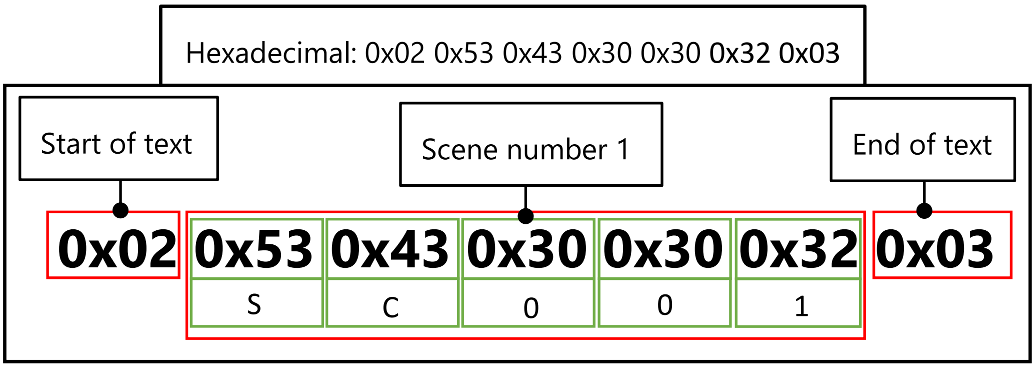
ASCII : $SC001&

Start of text

Scene number 1

End of text

$ SC001 &



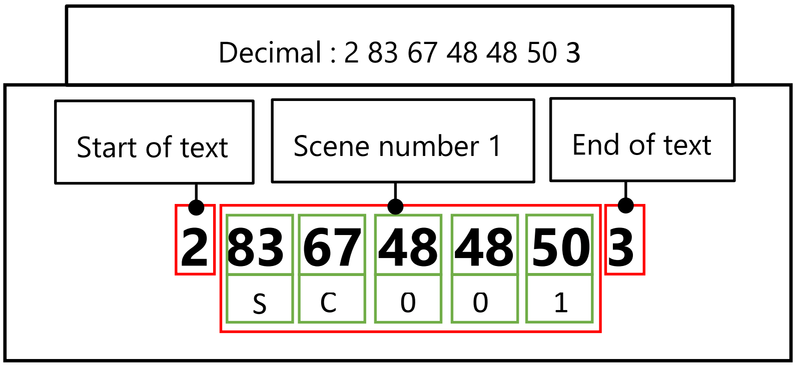
Hexadecimal: 0x02 0x53 0x43 0x30 0x30 0x32 0x03

Start of text

Scene number 1

End of text

0x02	0x53	0x43	0x30	0x30	0x32	0x03
	S	C	0	0	1	



Decimal : 2 83 67 48 48 50 3

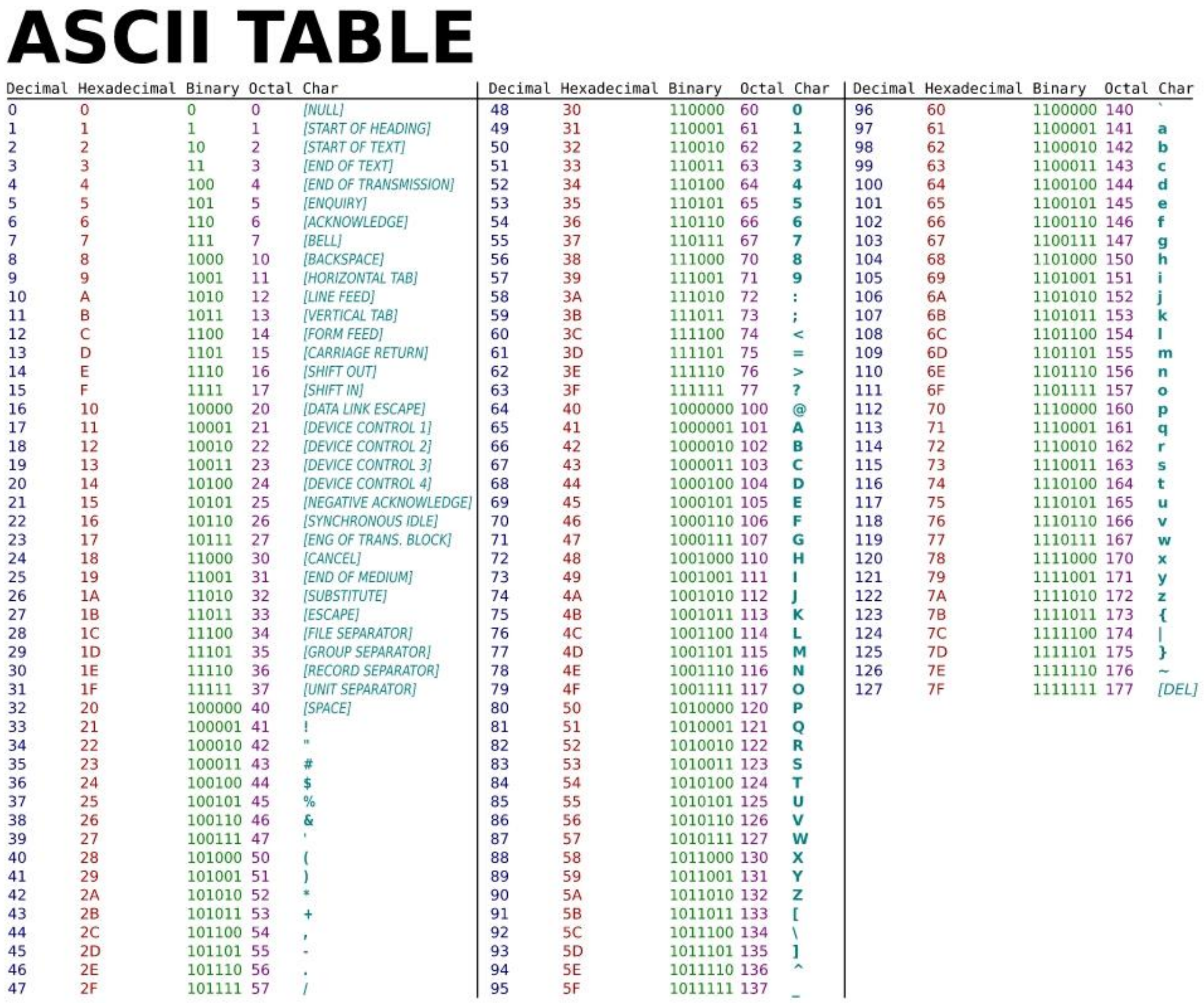
Start of text

Scene number 1

End of text

2	83	67	48	48	50	3
	S	C	0	0	1	

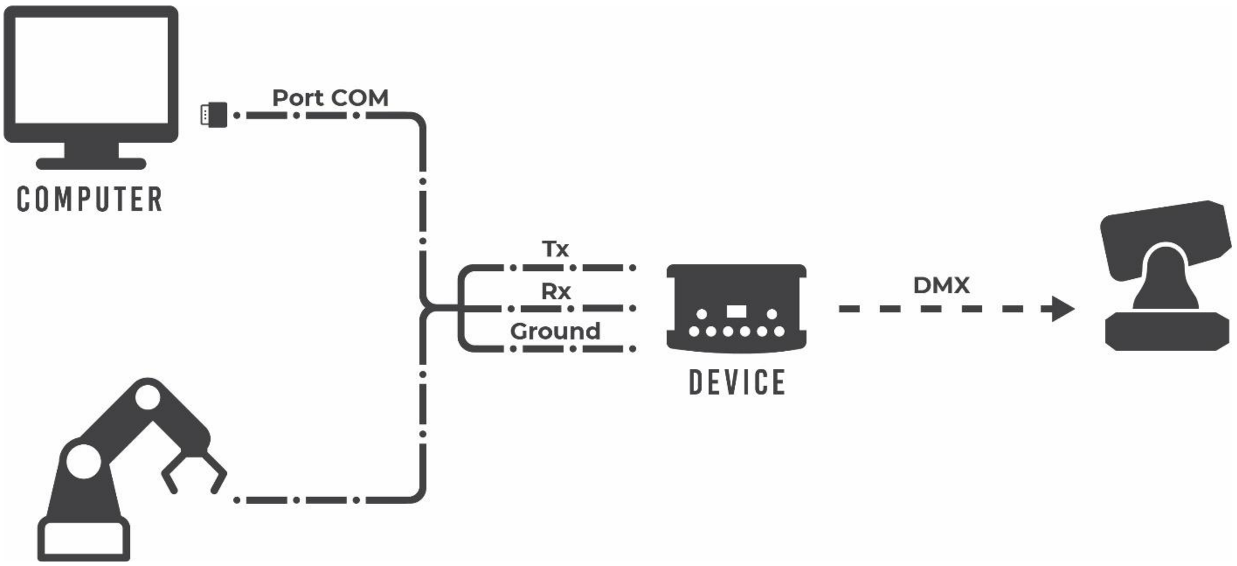
ASCII TABLE



ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[ENG OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x
25	19	11001	31	[END OF MEDIUM]	73	49	1001001	111	I	121	79	1111001	171	y
26	1A	11010	32	[SUBSTITUTE]	74	4A	1001010	112	J	122	7A	1111010	172	z
27	1B	11011	33	[ESCAPE]	75	4B	1001011	113	K	123	7B	1111011	173	{
28	1C	11100	34	[FILE SEPARATOR]	76	4C	1001100	114	L	124	7C	1111100	174	|
29	1D	11101	35	[GROUP SEPARATOR]	77	4D	1001101	115	M	125	7D	1111101	175	}
30	1E	11110	36	[RECORD SEPARATOR]	78	4E	1001110	116	N	126	7E	1111110	176	~
31	1F	11111	37	[UNIT SEPARATOR]	79	4F	1001111	117	O	127	7F	1111111	177	[DEL]
32	20	100000	40	[SPACE]	80	50	1010000	120	P					
33	21	100001	41	!	81	51	1010001	121	Q					
34	22	100010	42	"	82	52	1010010	122	R					
35	23	100011	43	#	83	53	1010011	123	S					
36	24	100100	44	$	84	54	1010100	124	T					
37	25	100101	45	%	85	55	1010101	125	U					
38	26	100110	46	&	86	56	1010110	126	V					
39	27	100111	47	'	87	57	1010111	127	W					
40	28	101000	50	(	88	58	1011000	130	X					
41	29	101001	51	)	89	59	1011001	131	Y					
42	2A	101010	52	*	90	5A	1011010	132	Z					
43	2B	101011	53	+	91	5B	1011011	133	[					
44	2C	101100	54	,	92	5C	1011100	134	\					
45	2D	101101	55	-	93	5D	1011101	135	]					
46	2E	101110	56	.	94	5E	1011110	136	^					
47	2F	101111	57	/	95	5F	1011111	137	_					

RS232 CONNECTION


Port COM
COMPUTER
Tx
Rx
Ground
DEVICE
DMX