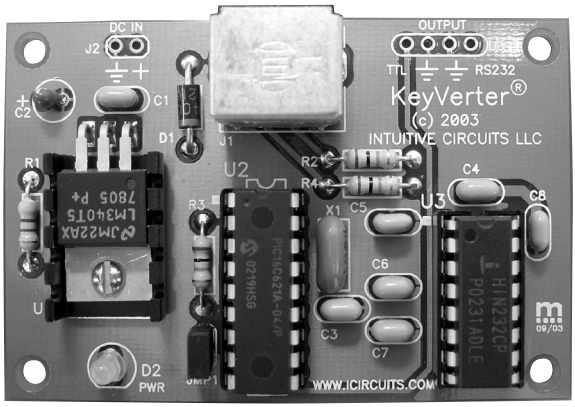
KeyVerter®

PS/2 Keyboard To RS-232/TTL Converter



Copyright © 2003 Intuitive Circuits, LLC

Description

KeyVerter® converts key stroke events from any PS/2 keyboard (small DIN connector) into an RS-232 and TTL serial output stream. Keyboard enable your favorite microcontroller like Parallax's BASIC Stamp 2 using only one 9600 baud serial input pin.

Specifications

Dimensions:	2.7" x 1.825" x .6"
Weight:	0.9 oz.
Input voltage:	7.5 to 12.0 volts DC (17 ma @ 9 volts without keyboard attached)
Operating temperature:	-10 C to +70 C
RS-232 output:	9600 baud, 8 data bits, 1 stop bit, no parity, inverted
TTL output:	9600 baud, 8 data bits, 1 stop bit, no parity, inverted

Connections

Pad/Connector	Hookup To
DC IN	Power supply +7 to +12 volts
GND	Power supply ground
RS-232 OUT	Microcontroller RS-232 serial input (Female DB-9 pin 2)
GND	Microcontroller RS-232 ground (Female DB-9 pin 5)
TTL OUT	Microcontroller TTL serial input
GND	Microcontroller TTL ground
J1	PS/2 keyboard

Configuration and Operation

KeyVerter® has two modes of reporting key stroke information. These modes are selected via the JMP1 jumper.

Jumper	State
JMP1	Removed: Normal mode Installed: Extended mode

Normal Mode - One byte is sent per key stroke (see figure 1.0)
For example when the user types "Yes" <enter> then 4 bytes: 59 hex, 65 hex, 73 hex, and 0D hex are sent out the TTL and RS-232 port.

Dec	Hex	Key	Dec	Hex	Key	Dec	Hex	Key	Dec	Hex	Key	Dec	Hex	Key
0	00		32	20	Space	64	40	@	96	60	`	128	80	Insert
1	01		33	21	!	65	41	A	97	61	a	129	81	Home
2	02		34	22	"	66	42	B	98	62	b	130	82	Page Up
3	03		35	23	#	67	43	C	99	63	c	131	83	Delete
4	04		36	24	$	68	44	D	100	64	d	132	84	End
5	05		37	25	%	69	45	E	101	65	e	133	85	Page Down
6	06		38	26	&	70	46	F	102	66	f	134	86	Up Arrow
7	07		39	27	'	71	47	G	103	67	g	135	87	Down Arrow
8	08	Backspace	40	28	(	72	48	H	104	68	h	136	88	Left Arrow
9	09	Tab	41	29	)	73	49	I	105	69	i	137	89	Right Arrow
10	0A		42	2A	*	74	4A	J	106	6A	j	138	8A	
11	0B		43	2B	+	75	4B	K	107	6B	k	139	8B	
12	0C		44	2C	,	76	4C	L	108	6C	l	140	8C	
13	0D	Enter	45	2D	-	77	4D	M	109	6D	m	141	8D	
14	0E		46	2E	.	78	4E	N	110	6E	n	142	8E	
15	0F		47	2F	/	79	4F	O	111	6F	o	143	8F	
16	10		48	30	0	80	50	P	112	70	p	144	90	
17	11		49	31	1	81	51	Q	113	71	q	145	91	F1
18	12		50	32	2	82	52	R	114	72	r	146	92	F2
19	13		51	33	3	83	53	S	115	73	s	147	93	F3
20	14		52	34	4	84	54	T	116	74	t	148	94	F4
21	15		53	35	5	85	55	U	117	75	u	149	95	F5
22	16		54	36	6	86	56	V	118	76	v	150	96	F6
23	17		55	37	7	87	57	W	119	77	w	151	97	F7
24	18		56	38	8	88	58	X	120	78	x	152	98	F8
25	19		57	39	9	89	59	Y	121	79	y	153	99	F9
26	1A		58	3A	:	90	5A	Z	122	7A	z	154	9A	F10
27	1B	Esc	59	3B	;	91	5B	[	123	7B	{	155	9B	F11
28	1C		60	3C	<	92	5C	\	124	7C	|	156	9C	F12
29	1D		61	3D	=	93	5D	]	125	7D	}	157	9D	
30	1E		62	3E	>	94	5E	^	126	7E	~	158	9E	
31	1F		63	3F	?	95	5F	_	127	7F		159	9F	

Fig 1.0 - KeyVerter® keyboard mapping chart

Configuration and Operation (cont.)

Extended Mode - Two bytes are sent per each key stroke, the Shift/Ctrl/Alt state (see figure 2.0) followed by the key stroke value (see figure 1.0)
For example when the user presses the left Control, left Alt, and Delete keys 2 bytes: D4 hex and 83 hex are sent out the TTL and RS-232 ports.

Bit	Hex	Key Press Status
0	01	Left Shift Key
1	02	Right Shift Key
2	04	Left Ctrl Key
3	08	Right Ctrl Key
4	10	Left Alt Key
5	20	Right Alt Key
6	40	Always 1
7	80	Always 1

Fig 2.0 - KeyVerter® keyboard extended key chart

Programming Sample

```
‘ keyvert.bs2 —- KeyVerter(r) PS/2 keyboard to RS-232/TTL converter BS2 sample
‘ (C) Copyright 2003, Intuitive Circuits, LLC
‘
‘ This program emulates a terminal by taking PS/2 keyboard input and displaying
‘ it on a 4 row x 20 column LCD.
‘
‘ Connect KeyVerter, BASIC Stamp 2 (BS2-IC), and 4 x 20 LCD as follows:
‘
‘       BASIC Stamp 2 pin 4 (GND)  ->  KeyVerter (TTL GND) -> LCD (GND)
‘       BASIC Stamp 2 pin 5 (I/O 0)  ->  LCD (SERIAL IN)
‘       BASIC Stamp 2 pin 6 (I/O 1)  <-  KeyVerter (TTL OUT)
‘
‘ * Also confirm KeyVerter’s JMP1 is removed for proper operation.

DIRS = %1111111111111101
      ‘ input pins are 0
      ‘ output pins are 1

‘ ——-[pin constants]————————————————————————————-
cSEROUT         CON   0       ‘ pin 0 - serial output connected to LCD (SERIAL IN)
cSERIN          CON   1       ‘ pin 1 - serial input connected to KeyVerter (TTL OUT)

‘ ——-[RS-232 constants]—————————————————————————————
cN96            CON   $4000+84    ‘ 9600 baud, 8 bit, no parity, inverted data
```

```
‘ ——-[lcd constants]————————————————————————————-
cLCD_I          CON   254     ‘ command prefix
cLCD_CLR        CON   1       ‘ lcd clear-screen command
cLCD_CURON      CON   13      ‘ turn on blinking block cursor

‘ ——-[variables]————————————————————————————-
x               VAR   BYTE    ‘ cursor x position (1-20)
y               VAR   BYTE    ‘ cursor y position (1-4)

key             VAR   BYTE    ‘ incoming key press

lcdRowBase      VAR   BYTE(4)

temp            VAR   BYTE

‘ ——-[code]—————————————————————————————————
main:
    pause 1000      ‘ wait for LCD to settle down

    ‘ lcd base position for each row
    lcdRowBase(0) = 128
    lcdRowBase(1) = 192
    lcdRowBase(2) = 148
    lcdRowBase(3) = 212

    serout cSEROUT,cN96,[cLCD_I,cLCD_CURON] ‘ turn on cursor

clear:
    x = 1           ‘ initialize cursor position
    y = 1
    serout cSEROUT,cN96,[cLCD_I,cLCD_CLR] ‘ clear lcd screen
    pause 1

loop:
    serin cSERIN,cN96,[key]

    if key = 8 then backspace
    if key = 13 then enter
    if key = 129 then homeKey
    if key = 132 then endKey
    if key = 145 then f1
    if key = 134 then upArrow
    if key = 135 then downArrow
    if key = 136 then leftArrow
    if key = 137 then rightArrow

    ‘ normal key press
    serout cSEROUT,cN96,[key]
    x = x + 1
    if x > 20 then incY
    goto loop
```

```
incY:
    x = 1
    y = y + 1
    if y <= 4 then no_wrap
    y = 1
no_wrap:
    gosub setCursor
    goto loop

backspace:
    goto leftArrow

enter:
    goto incY

homeKey:
    x = 1
    y = 1
    gosub setCursor
    goto loop

endKey:
    x = 20
    y = 4
    gosub setCursor
    goto loop

f1:
    goto clear  ‘ clear the screen

upArrow:
    y = y - 1
    if y > 0 then up_ok
    y = 4  ‘ wrapped
up_ok:
    gosub setCursor
    goto loop

downArrow:
    y = y + 1
    if y <= 4 then down_ok
    y = 1  ‘ wrapped
down_ok
    gosub setCursor
    goto loop
```

```
leftArrow:
    x = x - 1
    if x > 0 then left_ok
    x = 20 ‘ wrapped
    goto upArrow
left_ok:
    gosub setCursor
    goto loop

rightArrow:
    x = x + 1
    if x <= 20 then right_ok
    x = 1  ‘ wrapped
    goto downArrow
right_ok:
    gosub setCursor
    goto loop


‘———-[setCursor]————————————————————————————————————
‘
‘ set the lcd cursor position
‘
‘ Parameters: x = x position (1-20)
‘             y = y position (1-4)

setCursor
  temp = lcdRowBase(y-1) + (x-1)
  serout cSEROUT,cN96,[cLCD_I,temp] ‘ set cursor position
  return
```

Additional programming code samples are available at www.icircuits.com/download.htm. Any terminal program, such as HyperTerminal, can be used to demonstrate KeyVerter® functionality.

Warranty & Service

If the product fails to perform as described in our product description or specification, within 90 days from the date of shipment to the buyer, we will repair or replace the product and/or accessories originally supplied. Failure due to improper installation, misuse, abuse or accident is not covered by this warranty. Incidental and consequential damages are not covered by this warranty. The buyer must obtain a Return Material Authorization by calling (248) 524-1918, and shipping the defective product to Intuitive Circuits, 2275 Brinston, Troy, MI 48083, freight prepaid. After the warranty expires, we will promptly supply an estimate for the repair cost.

Intuitive Circuits, LLC

2275 Brinston
Troy, MI 48083
Voice: (248) 524-1918
Fax: (248) 524-3808
http://www.icircuits.com