



AN109 – EVD1000/1500 Serial Configuration Interface

INTRODUCTION

The EVD1000/1500 has a very flexible digital I/O structure, and in-circuit applications can be configured in a variety of fashions, utilizing different video data formats, levels of user control, etc. Most often, the EVD1000/1500 is configured through the use of an industry standard two-wire Serial Configuration Interface. In this Application Note, standard protocols and usage details for operation of the Serial Configuration Interface on these chips are given. For information about registers specific to the EVD1000/1500, please refer to the Data Sheet.

1 Serial Configuration Interface

The Serial Configuration bus interface is a standard slave transceiver, supporting 7-bit or 10-bit slave addresses and 400kbits/sec guaranteed transfer rate. It uses 8-bit sub-addressing with an auto-increment function.

The CLK_SYSTEM input pin is used as a clock for the Serial Configuration Interface, thus it must be running for the Serial Configuration Interface to function properly. The CLK_VIDEO input to the device is not used in the Serial Configuration Interface and thus need not be toggling for the device to be configured through the Serial Configuration Interface.

1.1 Slave Device Address

The table below shows the slave device address as a function of the SC_ADD_10_BIT_7_BIT_N_IN and the SC_ADD_CTRL_IN[6,3:0] input pins.

SC_ADD_10_BIT_7_BIT_N_IN	Slave Device address
0	{SC_ADD_CTRL_IN[6],01,SC_ADD_CTRL_IN[3:0]}
1	{000,SC_ADD_CTRL_IN[6],01,SC_ADD_CTRL_IN[3:0]}

10-bit addressing is compatible with, and can be combined with, 7-bit addressing. Using 10 bits for addressing exploits the reserved combination 1111XXX for the first seven bits of the first byte following a START(S) or repeated START(Sr) condition. The 10-bit slave address is formed from the first two bytes following a START condition(S) or a repeated START condition(Sr).

The first seven bits of the first byte are the combination 11110XX of which the last two bits(XX) are the two most-significant bits(MSBs) of the 10-bit address; the eighth bit of the first byte is the R/W_N bit that determines the direction of the message. If the R/W_N bit is 'zero', then the second byte contains the remaining 8 bits(XXXXXXXX) of the 10-bit address. If the R/W_N bit is 'one', then the next byte contains data transmitted from a slave to a master.



1.2 Write Format

7-bit address

S	Slave Address	R/W_N	A	Subaddress	A	Data0	A	...	Data N-1	A	P
0											

10-bit address

S	Slave Address 1 st 7 bits	R/W_N	A	Slave address 2 nd Byte	A	Subaddress	A	Data 0	A	...	Data N-1	A	P
11110XX 0 XXXX XXXX													

Where:

S	Start condition, 1 bit
Slave Address	7 Address bits
Slave Address 1 st 7 bits	Slave address 11110XX with XX being 2 MSBs
Slave Address 2 nd Byte	Second byte of slave address XXXX XXXX 8 LSBs
R/W_N	Read/Write Not
A	Acknowledge generated by EVD1000/1500, 1-bit
Subaddress	Subaddress of first register to write. 8-bits
Data0	First byte of data
...	N-2 {Data followed by A}
Data N-1	Nth byte of data
P	Stop condition

1.3 Read Format

7-bit address

S	Slave Address	R/W_N	A	Subaddress	A	
0						

10-bit address

S	Slave Address 1 st 7 bits	R/W_N	A	Slave address 2 nd byte	A	Subaddress	A	
11110XX 0 XXXX XXXX								

Then:

7-bit address

Sr	Slave Address	R/W_N	A	Data0	AM	...	Data(N-1)	NAM	P
1									

10-bit address

Sr	Slave Address	R/W_N	A	Data0	AM	...	Data(N-1)	NAM	P
11110XX 1									

Where:

S	Start condition, 1 bit
Slave Address	7 address bits
Slave Address 1 st 7 bits	Slave address 11110XX with XX being 2 MSBs
Slave Address 2 nd Byte	Second byte of slave address XXXX XXXX 8 LSBs
R/W_N	Read/Write Not
A	Acknowledge generated by EVD1000/1500, 1-bit
AM	Acknowledge, generated by a master, 1-bit
NAM	Not Acknowledge, generated by a master
Subaddress	Subaddress of the first register to read, 8-bits
Data(0)	First byte of the data read
Data(N-1)	Nth byte of the data read
...	N-2 {Data followed by AM}'s
P	Stop condition
Sr	Start repeat

**1.4 Slave Address Table (SC_ADD_10_BIT_7_BIT_N_IN=0, 7-bit)**

The following table shows the slave Address for the EVD1000/1500 as a function of the levels wired on the SC_ADD_CTRL_IN[6,3:0] inputs when in 7-bit address mode. The second column of the table shows the 7-bit Slave Address. The Device Address of the EVD1000/1500 may also be thought of as an 8-bit address, which would be the concatenation of the 7-bit Slave Address and the R/W_N bit. The third and fourth columns of the table show this 8-bit address as a function of the SC_ADD_CTRL_IN[6:3:0] inputs and the R/W_N bit..

{SC_ADD_CTRL_IN[6], SC_ADD_CTRL_IN[3:0]}	Slave Address 7-bits	{Slave_Address,R/W_N} 8-bits	
		R/W_N=0, Write	R/W_N=1, Read
0	10	20	21
1	11	22	23
2	12	24	25
3	13	26	27
4	14	28	29
5	15	2A	2B
6	16	2C	2D
7	17	2E	2F
8	18	30	31
9	19	32	33
A	1A	34	35
B	1B	36	37
C	1C	38	39
D	1D	3A	3B
E	1E	3C	3D
F	1F	3E	3F
10	50	A0	A1
11	51	A2	A3
12	52	A4	A5
13	53	A6	A7
14	54	A8	A9
15	55	AA	AB
16	56	AC	AD
17	57	AE	AF
18	58	B0	B1
19	59	B2	B2
1A	5A	B4	B3
1B	5B	B6	B4
1C	5C	B8	B5
1D	5D	BA	B7
1E	5E	BC	B9
1F	5F	BE	BF

For example, if SC_ADD_CTRL_IN[6] is wired to 1 and SC_ADD_CTRL_IN[3:0] are wired to value 5, then the Slave Address of the EVD1000/1500 would be 55 hex. The 8-bit address made from the concatenation of the Slave Address and the R/W_N bit would be AA hex for writes and AB hex for reads.

**1.5 Slave Address Table (SC_ADD_10_BIT_7_BIT_IN=1, 10-bit)**

The following table shows the slave Address for the EVD1000/1500 as a function of the levels wired on the SC_ADD_CTRL_IN[6,3:0] inputs when in 10-bit address mode. The second column of the table shows the 10-bit Slave Address. The Device Address of the EVD1000/1500 may also be thought of as an 11-bit address, which would be the concatenation of the 10-bit Slave Address and the R/W_N bit. The third and fourth columns of the table show this 11-bit address as a function of the SC_ADD_CTRL_IN[6:3:0] inputs and the R/W_N bit..

{SC_ADD_CTRL_IN[6], SC_ADD_CTRL_IN[3:0]}	Slave Address 10-bits	{Slave_Address,R/W_N} 11-bits	
		R/W_N=0, Write	R/W_N=1, Read
0	010	200	021
1	011	202	023
2	012	204	025
3	013	206	027
4	014	208	029
5	015	20A	20B
6	016	20C	20D
7	017	20E	20F
8	018	300	301
9	019	302	303
A	10A	304	305
B	10B	306	307
C	10C	308	309
D	10D	30A	30B
E	10E	30C	30D
F	10F	03E	30F
10	500	0A0	0A1
11	501	0A2	0A3
12	502	0A4	0A5
13	503	0A6	0A7
14	504	0A8	0A9
15	505	0AA	0AB
16	506	0AC	0AD
17	507	0AE	0AF
18	508	0B0	0B1
19	509	0B2	0B2
1A	50A	0B4	0B3
1B	50B	0B6	0B4
1C	50C	0B8	0B5
1D	50D	0BA	0B7
1E	50E	0BC	0B9
1F	50F	0BE	0BF

For example, if SC_ADD_CTRL_IN[6] is wired to 1 and SC_ADD_CTRL_IN[3:0] are wired to value 5, then the Slave Address of the EVD1000/1500 would be 055 hex. The 11-bit address made from the concatenation of the Slave Address and the R/W_N bit would be 0AA hex for writes and 0AB hex for reads.



For further questions or clarifications, contact your sales representative or the factory for additional support.

Enhanced Video Devices, Inc.
9830 Summers Ridge Road
San Diego, CA 92121
858-530-0100

www.enhancedvideodevices.com

Copyright © 2009 Enhanced Video Devices, Inc. (EVD). All rights reserved. The information contained herein is subject to change without notice in order to improve design and/or performance. EVD products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. EVD assumes no responsibility or liability for the use of any of its information, products, or services, and conveys no license or title under any patent, copyright, or mask work right to its products, unless otherwise expressly specified and agreed in writing. Furthermore, EVD's products are not designed for use as critical components in life support, life saving, critical control or safety applications where a malfunction or failure may reasonably be expected to result in significant injury or harm. The inclusion of EVD products in such applications implies that the manufacturer assumes all risk of such use and in doing so fully indemnifies EVD against all damages resulting from such application. Any applications that are described herein are for illustrative purposes only.