# XSV Parallel Port Interface

April 10, 2000 (Version 1.0) Application Note by D. Vanden Bout

## Summary

This application note describes the default parallel port interface circuit that is programmed into the XC95108 CPLD on the XSV Board. It also discusses how to change the parallel port interface to support other features.

## The Default Parallel Port Interface

Listing 1 shows the VHDL code for the default parallel port interface that is programmed into the XC95108 CPLD on the XSV Board. This interface provides two functions:

- It transfers configuration bitstreams from the PC to the Virtex FPGA.
- It lets the PC and the Virtex communicate through the parallel port after the FPGA is configured.

How the VHDL implements these functions is described below.

Lines 32–37 disable other chips and functions on the XSV Board so they cannot interfere with the configuration of the Virtex device. The Flash RAM is disabled by pulling its chip-enable pin high on line 34. The JTAG circuitry of the Virtex FPGA is kept quiescent by holding its clock pin low on line 35. And line 37 disables both the SAA7113 video decoder chip and the LXT970 Ethernet PHY chip. These chips are enabled again once the Virtex device is configured (as indicated when the Virtex DONE pin goes high).

The circuitry which actually controls the configuration of the Virtex device is described on lines 39–45. The three mode pins of the Virtex device are all pulled high on line 40 to set the device in the slave-serial configuration mode. The programming initiation pin and the configuration clock for the Virtex are connected to control lines C0 and C1 of the parallel port on lines 41 and 42, respectively. The configuration bitstream arrives serially over control line C3 and enters the least-significant bit of the Virtex FPGA programming data bus on line 45.

To configure the Virtex device, a program on the PC generates a low-going pulse on C0 to initiate the configuration process. Then it reads the configuration data from a .BIT file and sends it bit-by-bit through C3 while applying clock pulses on C1.

Once it has accepted an entire bitstream, the Virtex device will raise its DONE status pin to indicate it is configured. Line 45 causes the CPLD to disconnect C3 from the LSB of the programming data bus and connect data line D0 of the parallel port instead. The remaining seven parallel port data lines are connected to the Virtex programming data pins on line 48. The programming data pins act as general-purpose I/O pins after the Virtex device is configured. This provides a byte-wide path for data to come from the PC to the Virtex device.

Line 49 implements a channel for the Virtex device to pass data back to the PC through four status lines of the parallel port.

Finally, the logic levels on the Virtex DONE and INIT configuration status pins are displayed on segments 0 and 1 of the bargraph LED. This helps to detect problems that occur when configuring the Virtex FPGA.

## Changing the Parallel Port Interface

The parallel port interface is stored in the nonvolatile Flash of the XC95108 CPLD on the XSV Board. Any design you load into the CPLD will become active as soon as the XSV Board powers up. So it is possible to load a faulty interface design into the CPLD that makes it impossible to reprogram the CPLD even after you cycle the power. The only solution is to cut the CPLD off the board and replace it. So reprogramming the CPLD is a dangerous thing to do. This section will give you a couple of rules that may keep you out of trouble.

First of all, the CPLD gets its configuration data over the C1, C2, and C3 control lines of the parallel port. These lines connect to the JTAG inputs of the CPLD: TCK, TMS, and TDI, respectively. The CPLD also sends data back to the PC through its TDO pin that is attached to parallel port status line S7. The C1, C2, C3, and S7 parallel port lines are also connected to general-purpose I/O pins of the CPLD. If you program these general-purpose I/O pins as outputs, then it is impossible for the PC to override the CPLD outputs and the CPLD cannot be reprogrammed.

So how do you use these parallel port lines in an interface design while still allowing the CPLD to be reprogrammed? Just follow these rules:

- If any of the general-purpose I/O pins connected to C1, C2, C3, and S7 are only used as inputs, then no action needs to be taken for them.

- If any of the general-purpose I/O pins connected to C1, C2, C3, and S7 are used as outputs, they must enter a high-impedance state when the Virtex DONE pin is low (i.e., the Virtex FPGA is not configured).

By enabling any output drivers on the C1, C2, C3, and S7 lines with the Virtex DONE signal, you can always regain control of the CPLD by cycling the power or pulsing the Virtex PROGRAM pin. This places the Virtex FPGA in the unprogrammed state and its DONE signal goes low, which disables any drivers on C1, C2, C3, and S7.

Once you have removed the possibility of disabling the CPLD programming circuitry, you can begin adding new features to the interface circuit. One common alteration is to add circuitry that interfaces the Virtex FPGA to devices that are accessible from the CPLD.

For example, the CPLD connects to the serial port pins TXD (transmit data output), RXD (receive data input), CTS (clear to send input), and RTS (ready to send output). The Virtex FPGA can connect to these four pins through four lines connecting the FPGA and the CPLD. The CPLD and FPGA share 8 data lines, 21 address lines, and 3 control lines with the Flash RAM on the XSV Board. The CPLD and FPGA can communicate over 8+21+2=31 of these lines while holding the Flash chip-enable control input high to prevent it from interfering on the data lines. Two of these lines can be configured as inputs on the CPLD to take data from the Virtex FPGA and pass them to the TXD and RTS outputs. Another two lines can be configured as outputs on the CPLD to send data from the RXD and CTS outputs to the Virtex device. Then the Virtex FPGA can be configured to perform serial data communication through the serial port with the CPLD acting as the go-between.

**Listing 1: VHDL code for the default CPLD parallel port interface.**

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3
4   entity dwnldpar is
5         port(
6               -- parallel port data, control, and status pins
7               ppd: in std_logic_vector(7 downto 0);
8               ppc: in std_logic_vector(3 downto 0);
9               pps: out std_logic_vector(6 downto 3);
10
11              -- Virtex FPGA pins
12              V_a: in std_logic_vector(3 downto 0);   -- inputs from Virtex
13              V_tck: out std_logic;      -- driver to Virtex JTAG clock
14              V_cclk: out std_logic;     -- driver to Virtex config clock
15              V_progb: out std_logic;    -- driver to Virtex program pin
16              V_initb: in std_logic;     -- input from Virtex init pin
17              V_done: in std_logic;      -- input from Virtex done pin
18              V_d: out std_logic_vector(7 downto 0);  -- drivers to Virtex data pins
19              V_m: out std_logic_vector(2 downto 0);  -- Virtex config mode pins
20
21              ceb: out std_logic;        -- Flash chip-enable
22              resetb: out std_logic;     -- reset for video input and Ethernet chips
23              bar: out std_logic_vector(9 downto 0)   -- LED bargraph
24        );
25  end dwnldpar;
26
27  architecture dwnldpar_arch of dwnldpar is
28        constant LO: std_logic := '0';
29        constant HI: std_logic := '1';
30        constant SLAVE_SERIAL_MODE: std_logic_vector(2 downto 0) := "111";
31  begin
32        -- disable other chips on the XSV Board so they don't interfere
33        -- during the configuration of the Virtex FPGA
34        ceb          <= HI; -- disable Flash
35        V_tck  <= LO; -- deactivate Virtex JTAG circuit
36        -- disable the video input and Ethernet chips until config is done
37        resetb <= LO when V_done=LO else HI;
38
39        -- connect Virtex configuration pins
40        V_m    <= SLAVE_SERIAL_MODE;      -- set Virtex config mode pins
41        V_progb<= ppc(0);   -- Virtex programming pulse comes from parallel port
42        V_cclk <= ppc(1);   -- Virtex config clock comes from parallel port
43        -- config bitstream comes from parallel port control pin until
44        -- config is done and then gets driven by parallel port data pin
45        V_d(0) <= ppc(3) when V_done=LO else ppd(0);
46
47        -- connect the rest of the parallel port to the Virtex FPGA
48        V_d(7 downto 1) <= ppd(7 downto 1);     -- data from PC
49        pps(6 downto 3) <= V_a(3 downto 0);     -- status back to PC
50
51        -- display status of Virtex done and init pins on the bargraph LED
52        bar(0) <= V_done;
53        bar(1) <= V_initb;
54  end dwnldpar_arch;
```

**Listing 2: User-constraint file for CPLD pin assignments.**

```
#
# pin assignments for the XC95108 CPLD chip on the XSV Board
#

# Virtex FPGA
net V_tck           loc=p4;
# net V_dout        loc=p6;
# net V_din         loc=p32;
# net V_wrb         loc=p7;
# net V_csb         loc=p8;
net V_initb         loc=p9;
net V_done          loc=p10;
net V_progb         loc=p11;
net V_cclk          loc=p12;
net V_m<0>          loc=p13;
net V_m<1>          loc=p14;
net V_m<2>          loc=p15;
net V_d<0>          loc=p32;
net V_d<1>          loc=p33;
net V_d<2>          loc=p34;
net V_d<3>          loc=p35;
net V_d<4>          loc=p36;
net V_d<5>          loc=p37;
net V_d<6>          loc=p39;
net V_d<7>          loc=p40;
net V_a<0>          loc=p16;
net V_a<1>          loc=p17;
net V_a<2>          loc=p18;
net V_a<3>          loc=p19;
# net V_a<4>        loc=p20;
# net V_a<5>        loc=p23;
# net V_a<6>        loc=p24;
# net V_a<7>        loc=p25;
# net V_a<8>        loc=p27;
# net V_a<9>        loc=p28;
# net V_a<10>       loc=p29;
# net V_a<11>       loc=p30;
# net V_a<12>       loc=p49;
# net V_a<13>       loc=p50;
# net V_a<14>       loc=p52;
# net V_a<15>       loc=p53;
# net V_a<16>       loc=p54;
# net V_a<17>       loc=p55;
# net V_a<18>       loc=p56;
# net V_a<19>       loc=p58;
# net V_a<20>       loc=p59;
# net V_frdy        loc=p41;     # Flash ready status
# net V_c<0>        loc=p42;     # Flash output-enable
# net V_c<1>        loc=p43;     # Flash write-enable
# net V_c<2>        loc=p46;     # Flash chip-enable

# Flash RAM
# net resetb        loc=p3;
# net rdy           loc=p41;
net ceb             loc=P46;
# net oeb           loc=p42;
# net web           loc=p43;
# net d<0>          loc=p32;
# net d<1>          loc=p33;
# net d<2>          loc=p34;
# net d<3>          loc=P35;
# net d<4>          loc=P36;
```

```
63   # net d<5>          loc=P37;
64   # net d<6>          loc=P39;
65   # net d<7>          loc=P40;
66   # net a<0>          loc=p16;
67   # net a<1>          loc=p17;
68   # net a<2>          loc=p18;
69   # net a<3>          loc=p19;
70   # net a<4>          loc=p20;
71   # net a<5>          loc=p23;
72   # net a<6>          loc=p24;
73   # net a<7>          loc=p25;
74   # net a<8>          loc=p27;
75   # net a<9>          loc=p28;
76   # net a<10>         loc=p29;
77   # net a<11>         loc=p30;
78   # net a<12>         loc=p49;
79   # net a<13>         loc=p50;
80   # net a<14>         loc=p52;
81   # net a<15>         loc=p53;
82   # net a<16>         loc=p54;
83   # net a<17>         loc=p55;
84   # net a<18>         loc=p56;
85   # net a<19>         loc=p58;
86   # net a<20>         loc=p59;
87
88   # DIP and pushbutton switches
89   # net dipsw<1>      loc=p50;
90   # net dipsw<2>      loc=p52;
91   # net dipsw<3>      loc=p53;
92   # net dipsw<4>      loc=p54;
93   # net dipsw<5>      loc=p55;
94   # net dipsw<6>      loc=p56;
95   # net dipsw<7>      loc=p58;
96   # net dipsw<8>      loc=p59;
97   # net sw4           loc=p7;
98
99   # left, right, and bargraph LEDs
100  # net ls<0>         loc=p32;
101  # net ls<1>         loc=p33;
102  # net ls<2>         loc=p34;
103  # net ls<3>         loc=p35;
104  # net ls<4>         loc=p36;
105  # net ls<5>         loc=p37;
106  # net ls<6>         loc=p39;
107  # net rs<0>         loc=p40;
108  # net rs<1>         loc=p16;
109  # net rs<2>         loc=p17;
110  # net rs<3>         loc=p18;
111  # net rs<4>         loc=p19;
112  # net rs<5>         loc=p20;
113  # net rs<6>         loc=p23;
114  net bar<0>          loc=p24;
115  net bar<1>          loc=p25;
116  # net bar<2>        loc=p27;
117  # net bar<3>        loc=p28;
118  # net bar<4>        loc=p29;
119  # net bar<5>        loc=p30;
120  # net bar<6>        loc=p49;
121  # net bar<7>        loc=p42;
122  # net bar<8>        loc=p43;
123  # net bar<9>        loc=p41;
124
125  # Ethernet transceiver
126  # net ledsb         loc=p1;
127  # net ledrb         loc=p95;
```

```
128   # net ledtb         loc=p96;
129   # net ledlb         loc=p97;
130   # net ledcb         loc=p99;
131   # net cfg<0>        loc=p93;
132   # net cfg<1>        loc=p2;
133   net resetb          loc=p3;
134   # net mf<0>         loc=p91;
135   # net mf<1>         loc=p90;
136   # net mf<2>         loc=p89;
137   # net mf<3>         loc=p87;
138   # net mf<4>         loc=p86;
139   # net fde           loc=p92;
140   # net mddis         loc=p94;
141
142   # programmable oscillator
143   # net clk           loc=p22;
144
145   # video decoder
146   # net ce            loc=p3;
147
148   # parallel port
149   net ppd<0>          loc=p77;
150   net ppd<1>          loc=p74;
151   net ppd<2>          loc=p72;
152   net ppd<3>          loc=p70;
153   net ppd<4>          loc=p68;
154   net ppd<5>          loc=p67;
155   net ppd<6>          loc=p66;
156   net ppd<7>          loc=p65;
157   net ppc<0>          loc=p79;
158   net ppc<1>          loc=p78;
159   # net ppc<2>        loc=p73;
160   net ppc<3>          loc=p71;
161   net pps<3>          loc=p76;
162   net pps<4>          loc=p60;
163   net pps<5>          loc=p61;
164   net pps<6>          loc=p64;
165   # net pps<7>        loc=p63;
166
167   # serial port
168   # net rxd           loc=p80;
169   # net txd           loc=p81;
170   # net rts           loc=p82;
171   # net cts           loc=p85;
```