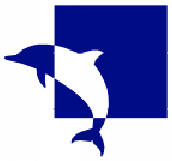
Electronics and Computer Science

University of Southampton



Xilinx FPGA Reconfiguration using JTAG

The Good News (Theory)

Following on from Xilinx 9500 series CPLDs, which were in-system programmed over their JTAG Test Access Ports (TAPs), Xilinx 4000 series FPGAs include a JTAG TAP for boundary scan testing, and for reconfiguration.

The Bad News (Reality)

Unlike the 9500 implementation, the 4000 series TAP was not hardwired. Instead, it was merely one way in which a set of multi-function I/O pins could be configured. Xilinx do not provide an application note or set of sample source code files which demonstrate reconfiguration via the 4000 series TAP.

The requirement

The Department of Electronics and Computer Science at the University of Southampton (ECS) had been using XESS Corporation XS-40 units for undergraduate FPGA experiments. A unit more suited to specific needs was required, which would also support PIC16F84 microcontroller exercises. The resultant specification for the Microsystem Experimenter Unit (MEU) required an XC4013 FPGA to power up as "SoftPIC", but to be reconfigurable to a user design.

Implementation – hardware

The best arrangement seemed to be PROM configuration and JTAG reconfiguration. Accordingly the MEU's FPGA MODE pins are strapped LO, and its multi-function TAP pins are connected to circuitry emulating the functions in the Xilinx parallel download cable (allowing a purely passive connection between the user's PC and the MEU).

Implementation – firmware

A great deal of experimentation was required to produce a functional system. Ultimately, this proved to have two key components:

- The power-up configuration in PROM must instantiate a functional TAP
- Each and every downloaded reconfiguration must also instantiate a functional TAP

Isn't hindsight wonderful? The above now seems blindingly obvious, but it took a while to prove.

Xilinx FPGA Reconfiguration using JTAG Solution – sample files

The following is a scrap of VHDL which contains all the essentials allowing a TAP to be implemented:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

package components is

attribute syn_black_box: boolean;
attribute black_box_pad_pin: string;
attribute syn_noprune: boolean;

attribute syn_black_box of components : package is true;

component TDI
   port( I : out STD_LOGIC);
end component;

component TCK
   port( I : out STD_LOGIC);
end component;

component TMS
   port( I : out STD_LOGIC);
end component;

component TDO
   port( O : in STD_LOGIC);
end component;

attribute syn_noprune of TDO: component is true;
attribute syn_black_box of others  : component is true;
attribute black_box_pad_pin of TCK: component is "I";
attribute black_box_pad_pin of TDI: component is "I";
attribute black_box_pad_pin of TMS: component is "I";
attribute black_box_pad_pin of TDO: component is "O";

component BSCAN
   port( TDO  : out STD_LOGIC;
         DRCK : out STD_LOGIC;
         IDLE : out STD_LOGIC;
         SEL1 : out STD_LOGIC;
         SEL2 : out STD_LOGIC;
         TDI  : in  STD_LOGIC;
         TMS  : in  STD_LOGIC;
         TCK  : in  STD_LOGIC;
         TDO1 : in  STD_LOGIC;
         TDO2 : in  STD_LOGIC);
end component;

end package components;
```

Xilinx FPGA Reconfiguration using JTAG Solution – sample files

The following code has a simple counter and a TAP:

```
-- Example of application (counter) with JTAG programming port

-- You must include the file jtag.vhd in both Synplify and Modelsim projects.

-- You will get the following warnings from Synplify, they can be ignored!
-- @W:"h:\vhdl\d4\exjtag.vhd":28:0:28:1|Port sel2 of entity work.bscan is unconnected
-- @W:"h:\vhdl\d4\exjtag.vhd":28:0:28:1|Port sel1 of entity work.bscan is unconnected
-- @W:"h:\vhdl\d4\exjtag.vhd":28:0:28:1|Port idle of entity work.bscan is unconnected
-- @W:"h:\vhdl\d4\exjtag.vhd":28:0:28:1|Port drck of entity work.bscan is unconnected
-- @W:"h:\vhdl\d4\exjtag.vhd":18:43:18:48|tdo1_p is not assigned a value (floating)
-- @W:"h:\vhdl\d4\exjtag.vhd":18:35:18:40|tdo2_p is not assigned a value (floating)

-- You will get the following warnings from Modelsim compilation, they can be ignored!
-- # WARNING[1]: H:/vhdl/d4/exjtag.vhd(42): No default binding for component: "bscan". (No entity named "bscan" was found)
-- # WARNING[1]: H:/vhdl/d4/exjtag.vhd(44): No default binding for component: "tdi". (No entity named "tdi" was found)
-- # WARNING[1]: H:/vhdl/d4/exjtag.vhd(46): No default binding for component: "tck". (No entity named "tck" was found)
-- # WARNING[1]: H:/vhdl/d4/exjtag.vhd(48): No default binding for component: "tms". (No entity named "tms" was found)
-- # WARNING[1]: H:/vhdl/d4/exjtag.vhd(50): No default binding for component: "tdo". (No entity named "tdo" was found)

-- You will get the following warnings from Modelsim, when you load the design for simulation.
-- They can be ignored!
-- # ** Warning: Component u0 is not bound.
-- #    Time: 0 ns  Iteration: 0  Region: /toplevel
-- # ** Warning: Component u1 is not bound.
-- #    Time: 0 ns  Iteration: 0  Region: /toplevel
-- # ** Warning: Component u2 is not bound.
-- #    Time: 0 ns  Iteration: 0  Region: /toplevel
-- # ** Warning: Component u3 is not bound.
-- #    Time: 0 ns  Iteration: 0  Region: /toplevel
-- # ** Warning: Component u4 is not bound.
-- #    Time: 0 ns  Iteration: 0  Region: /toplevel

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.ALL;

-- This package includes definitions of the JTAG TAP
use work.components.all;
-- End of JTAG package

entity toplevel is
    Port ( a : out std_logic_vector(15 downto 0);
           ck_20mhz : in std_logic;
           n_reset : in std_logic);
end toplevel;

architecture xilinx of toplevel is

-- These signals must be included for JTAG TAP
signal TCK_P, TDI_P, TMS_P, TDO_P, TDO2_P, TDO1_P : STD_LOGIC;
-- End of JTAG signals

-- This signal is for the example only. Include your own signals here, instead.
signal count : unsigned(15 downto 0);

begin

-- These 5 components are needed for the JTAG TAP
U0: BSCAN port map (TDO => TDO_P, TDI => TDI_P, TMS => TMS_P, TCK => TCK_P,
TDO2 => TDO2_P, TDO1 => TDO1_P);

U1: TDI port map (I => TDI_P);

U2: TCK port map (I => TCK_P);

U3: TMS port map (I => TMS_P);

U4: TDO port map (O => TDO_P);

-- End of JTAG components

-- This is the application - put your own application here.
     process (ck_20mhz, n_reset) is
          begin
            if (n_reset = '0') then
                count <= (others => '0');
                 elsif rising_edge(ck_20mhz) then
                     count <= count + 1;
                 end if;
          end process;

a <= std_logic_vector( count );

-- end of example

end xilinx;
```

Xilinx FPGA Reconfiguration using JTAG

Solution – operating procedures

(The following are the instructions provided to students)

Once a design has been placed and routed, the resultant .bit file can be downloaded into the XC4013 FPGA on the MEU via the latter's JTAG port. For this to work, the design must itself include a valid JTAG port correctly mapped onto the XC4013's pins TMS, TCK, TDI and TDO. A VHDL example along with a VHDL package defining the JTAG port is available for reference, along with a suitable constraints file.

When place and route is complete, select Tools → JTAG Programmer in Xilinx Design Manager. This should bring up a schematic showing TDI routing to TDO via a block representing the FPGA labelled with the FPGA part number and the design .bit filename. The first time this utility is used, select Output → Cable Setup and make sure that a "Parallel" cable on "lpt1" is selected. To run the download, Select Operations → Program.

If the JTAG Programmer starts up without the little schematic, select File → Initialise Chain (or press Ctrl-I). You will be prompted to browse for the .bit file to use.

To avoid having to go through Xilinx Design Manager every time you want to use a configuration established in a previous session, you can set up a Desktop shortcut to the JTAG Programmer – the relevant executable is jtagpgmr.exe. When running the programmer utility directly, you will always need to go through Initialise Chain to set up the port and downloadable bit file.

Warnings

The downloader does NOT verify that the download has been carried out correctly.

Running Output → Cable Setup should return the message "Communications established". This is misleading - all the software does is to confirm that the parallel port has a particular set of loop-back connections. No attempt is made to communicate with the JTAG TAP, and the software does not even check that the remote end of the cable is attached to a powered system. Unlike the Lattice isp system, any faulty connections in the download chain are detected only when a download is commanded.

Acknowledgements

The work showing how to instantiate a TAP and the need to include this in any reconfiguration was carried out by Iain McNally and Peter Troundle. Mark Zwolinski converted their Verilog code to VHDL, and developed the sample code and operating procedures for use by students.