

PicoBlaze™

for Spartan-3, Virtex-II, Virtex-IIPRO and Virtex-4 devices

JTAG Loader

Quick User Guide

Kris Chaplin and Ken Chapman

Limited Warranty and Disclaimer. These designs are provided to you “as is”. Xilinx and its licensors make and you receive no warranties or conditions, express, implied, statutory or otherwise, and Xilinx specifically disclaims any implied warranties of merchantability, non-infringement, or fitness for a particular purpose. Xilinx does not warrant that the functions contained in these designs will meet your requirements, or that the operation of these designs will be uninterrupted or error free, or that defects in the Designs will be corrected. Furthermore, Xilinx does not warrant or make any representations regarding use or the results of the use of the designs in terms of correctness, accuracy, reliability, or otherwise.

Limitation of Liability. In no event will Xilinx or its licensors be liable for any loss of data, lost profits, cost or procurement of substitute goods or services, or for any special, incidental, consequential, or indirect damages arising from the use or operation of the designs or accompanying documentation, however caused and on any theory of liability. This limitation will apply even if Xilinx has been advised of the possibility of such damage. This limitation shall apply notwithstanding the failure of the essential purpose of any limited remedies herein.

JTAG Loader prepared by
Kris Chaplin
Customer Applications Engineer
Xilinx Ltd
email:chaplin@xilinx.com

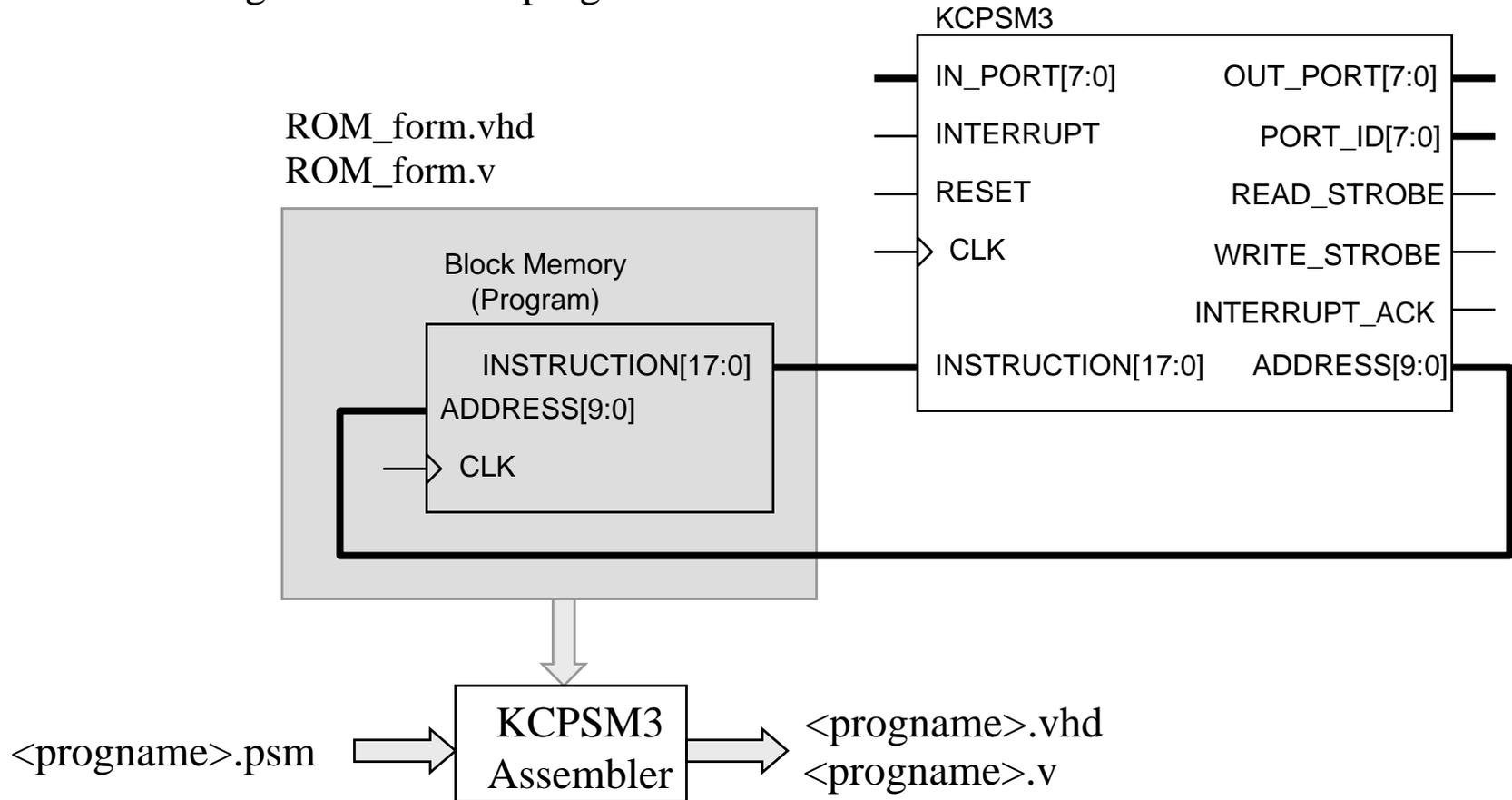
Please contact the author or Xilinx Technical support with any questions about this material.

This material should not be copied or circulated without permission of Xilinx Ltd.

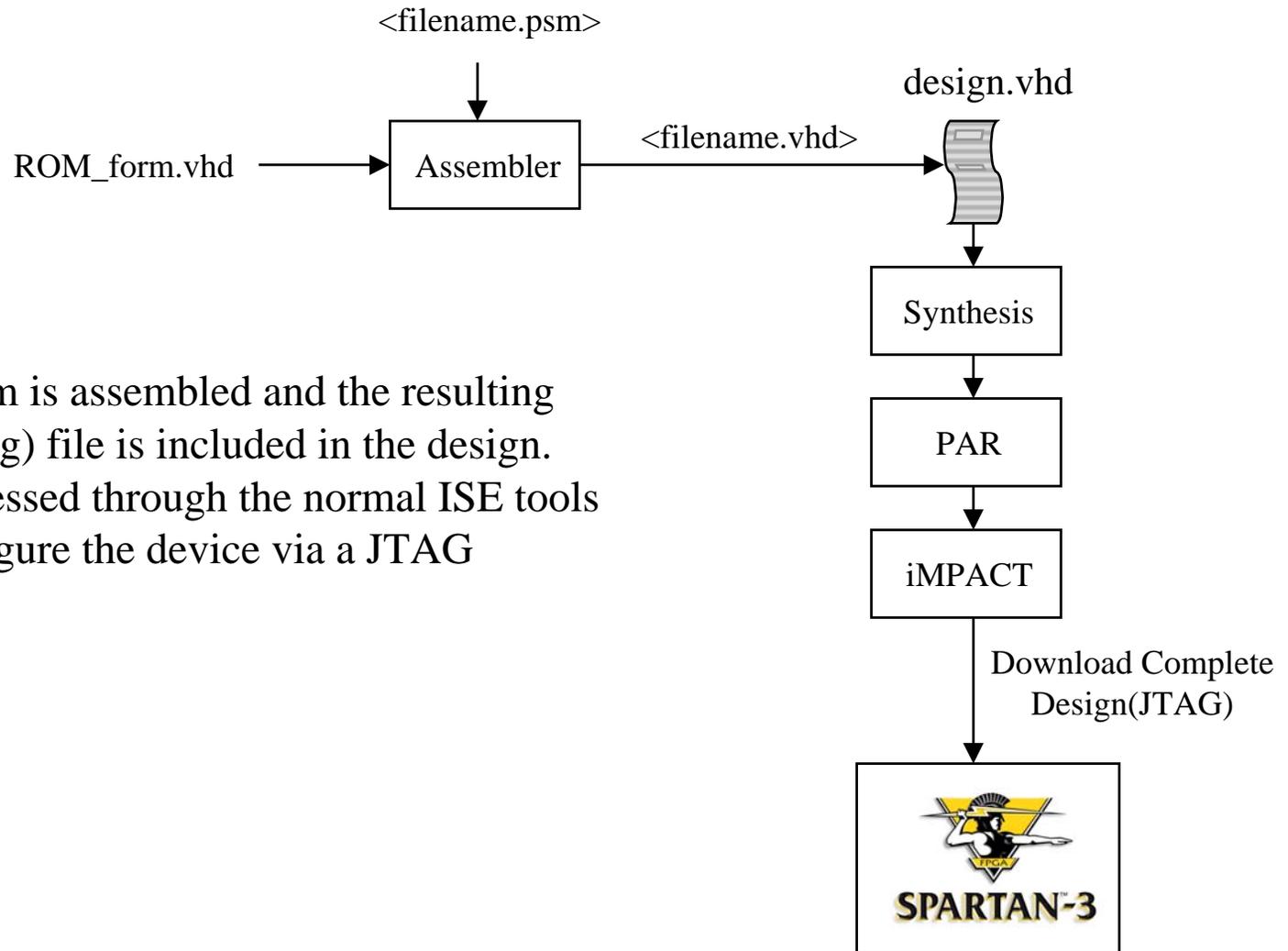


Normal PicoBlaze Design

A PicoBlaze (KCPSM3) program is stored in a BRAM configured as a ROM. The program is normally modified by a change to the configuration bit stream. The KCPSM3 assembler reads a VHDL or Verilog template describing the BRAM configuration and simply adds the initialization strings to define the program.



Normal Design Flow

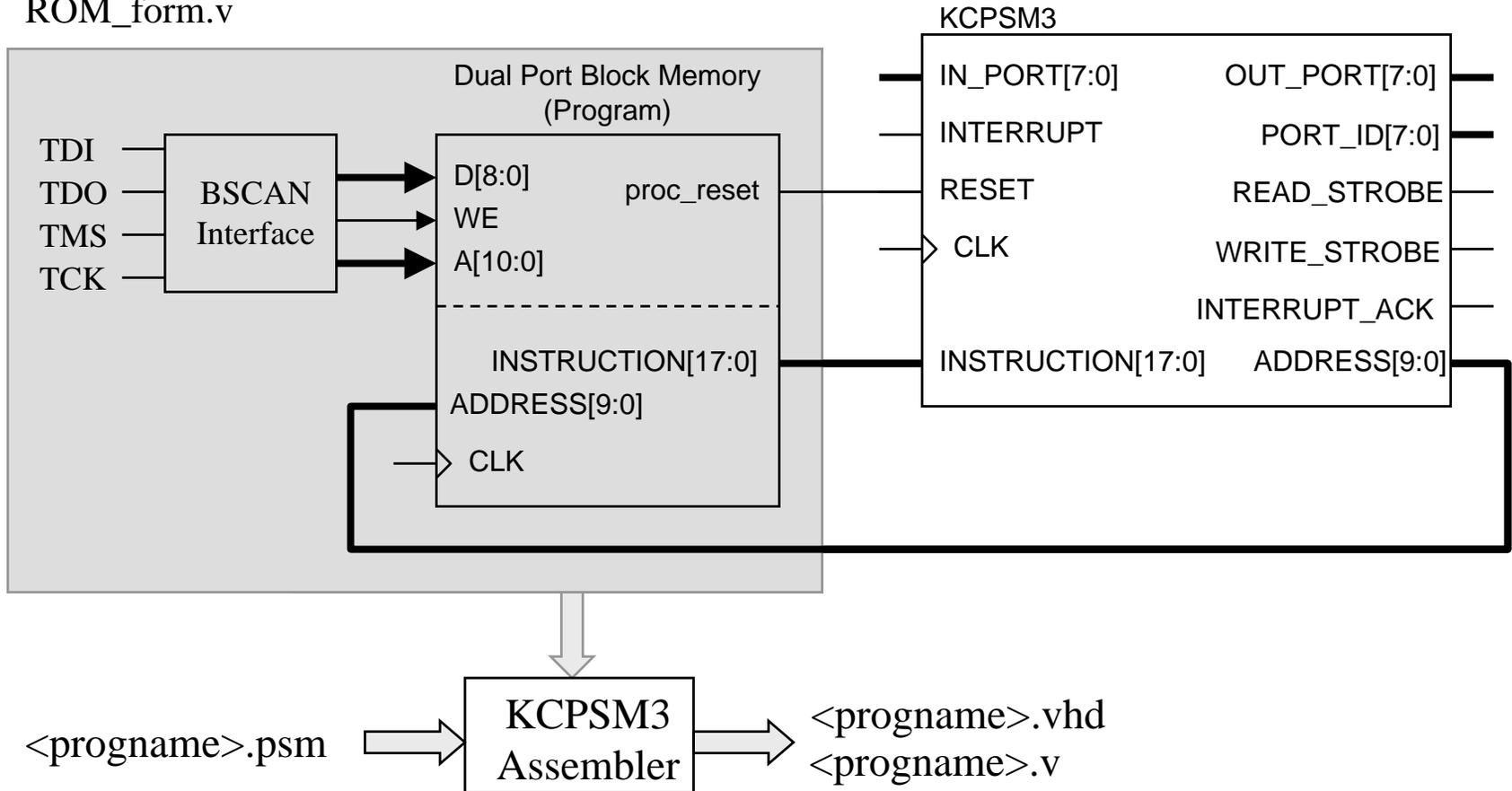


The PSM program is assembled and the resulting VHDL (or Verilog) file is included in the design. This is then processed through the normal ISE tools and used to configure the device via a JTAG download cable.

PicoBlaze JTAG Program Loader

The 'ROM_form' template is replaced. This adds a few slices of logic to connect the second port of the BRAM to the JTAG controller inside the FPGA. It also adds a reset control.

ROM_form.vhd
ROM_form.v



Insert JTAG Loader

- 1 - Replace the ROM_form.vhd (or ROM_form.v) file in your project directory.
- 2 - Assemble your program to create new VHDL (or Verilog) file.
- 3 - Add the 'reset' to the instantiation of the the program ROM **and connect to the PicoBlaze.**

```
component my_prog
  Port (
    address : in std_logic_vector(9 downto 0);
    instruction : out std_logic_vector(17 downto 0);
    proc_reset : out std_logic;
    clk : in std_logic);
end component;
```

```
processor: kcpsm3
  port map(
    address => address,
    instruction => instruction,
    port_id => port_id,
    write_strobe => write_strobe,
    out_port => out_port,
    read_strobe => read_strobe,
    in_port => in_port,
    interrupt => interrupt,
    interrupt_ack => interrupt_ack,
    reset => reset,
    clk => clk);

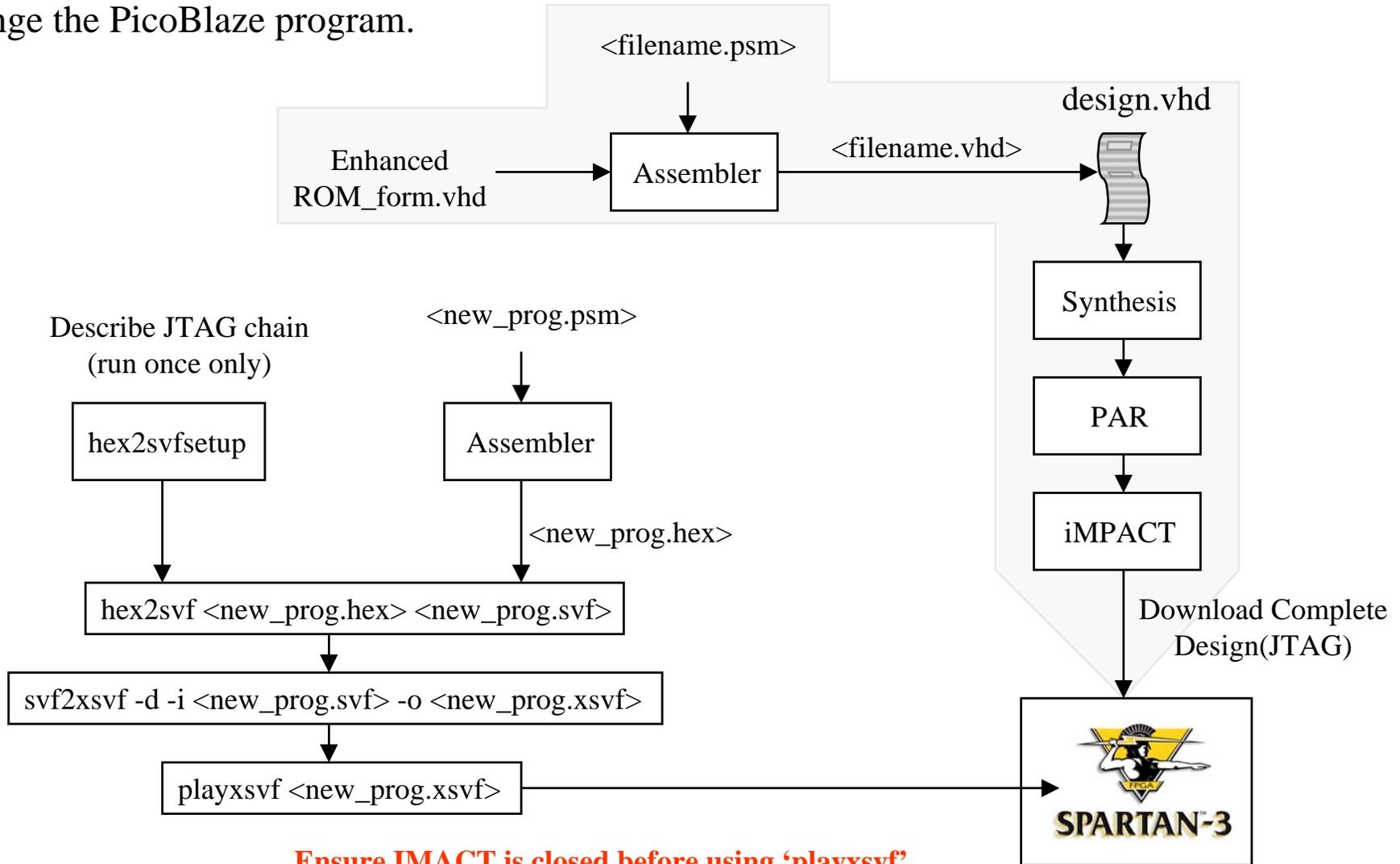
program_rom: my_prog
  port map(
    address => pv_address,
    instruction => pv_instruction,
    proc_reset => reset,
    clk => clk);
```

The reset will ensure the program is executed from the beginning following each new download.

- 4 - Synthesize and download the new design.

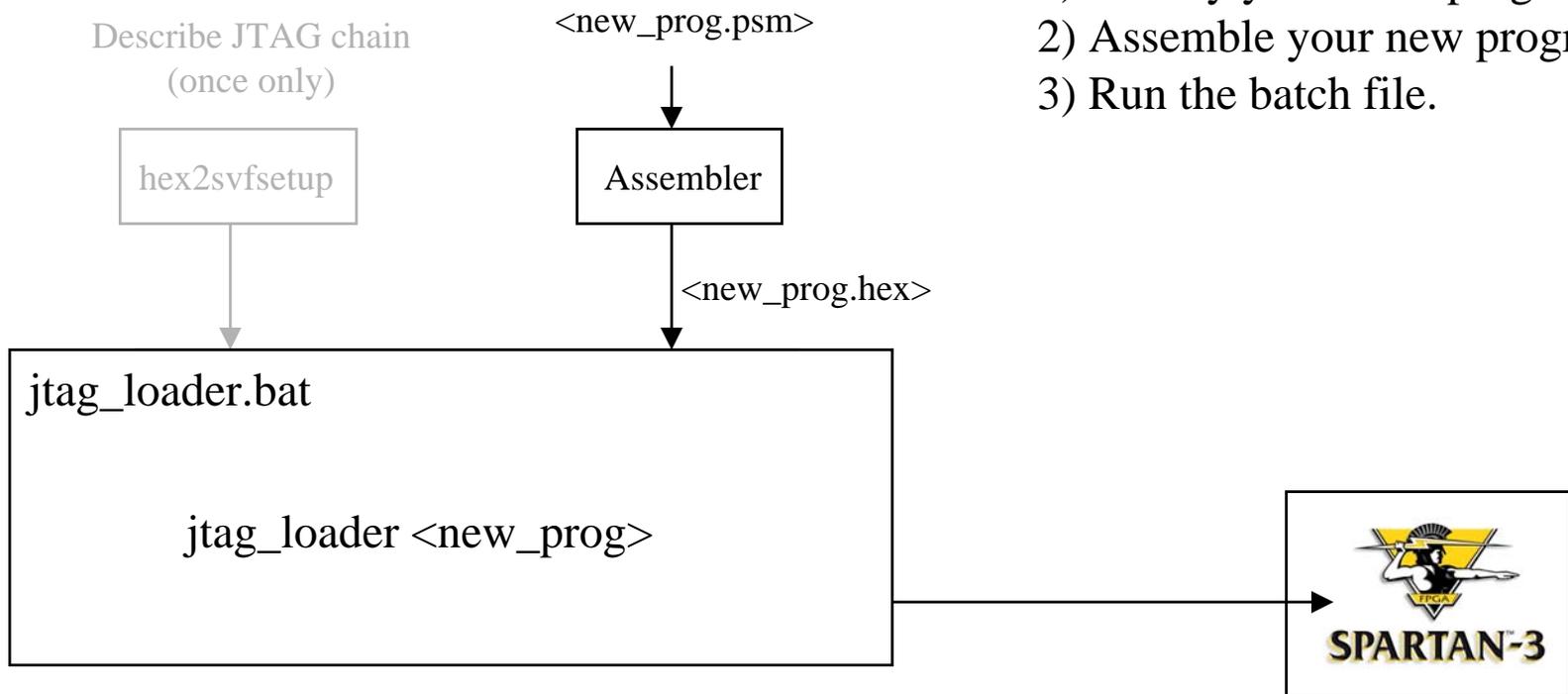
JTAG Loader Programs

Once the new design is configured into the device, a new set of programs can be used to rapidly change the PicoBlaze program.



The JTAG Loader “1-2-3”

Once the set up program has been used once, a batch file (provided) makes the execution of the remaining 3 programs much easier and faster.



- 1) Modify your PSM program.
- 2) Assemble your new program.
- 3) Run the batch file.

Working or Not Working?

When the 'playxsvf' part of the JTAG loader works the process completes in a few seconds and a few simple messages are displayed....

```
XSVF Player v5.01, Xilinx, Inc.  
XSVF file = led_lab.xsvf  
SUCCESS - Completed XSVF execution.  
Execution Time = 1.061 seconds
```

However, if the player can not access the JTAG cable, the process will appear to take a long time and the DOS window will be filled TCK, TMS and TDI values. These appear so fast that you will not notice the message about not being able to connect to the parallel cable.

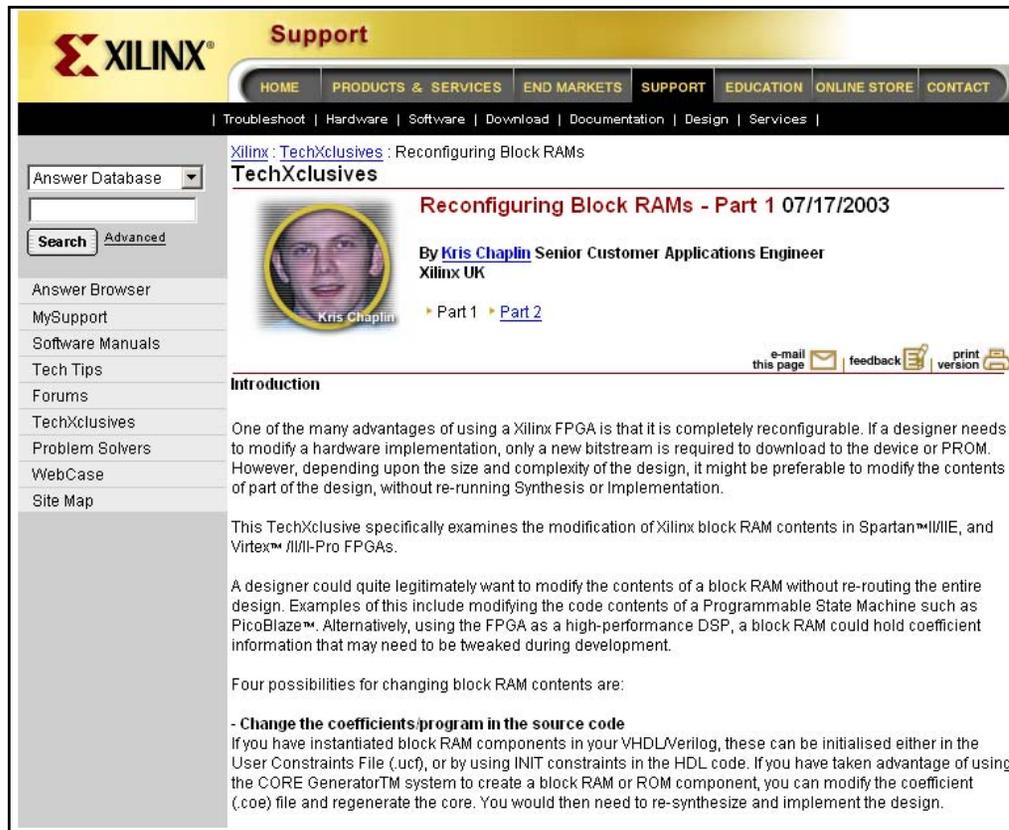
```
XSVF Player v5.01, Xilinx, Inc.  
INFO: XSVF file = blink.xsvf  
ERROR: Xilinx Parallel Cable is not connected to parallel port.  
TCK = 0; TMS = 1; TDI = 0  
TCK = 1; TMS = 1; TDI = 0  
TCK = 0; TMS = 1; TDI = 0  
TCK = 1; TMS = 1; TDI = 0  
TCK = 0; TMS = 1; TDI = 0  
TCK = 1; TMS = 1; TDI = 0  
TCK = 0; TMS = 1; TDI = 0  
TCK = 1; TMS = 1; TDI = 0  
TCK = 0; TMS = 1; TDI = 0  
TCK = 1; TMS = 1; TDI = 0
```

Make sure the parallel JTAG cable is connected and close all other programs which use the JTAG cable such as iMPACT as these may be preventing access.

Further Reading

To read more about the JTAG mechanism being used by this utility, please visit the TechXclusive web site.....

http://support.xilinx.com/xlnx/xweb/xil_tx_home.jsp



The screenshot displays the Xilinx Support website interface. At the top, the Xilinx logo is on the left, and the word 'Support' is centered. Below this is a navigation bar with links for HOME, PRODUCTS & SERVICES, END MARKETS, SUPPORT, EDUCATION, ONLINE STORE, and CONTACT. A secondary navigation bar lists Troubleshoot, Hardware, Software, Download, Documentation, Design, and Services. On the left side, there is a search box with 'Answer Database' selected, a search button, and a list of navigation links including Answer Browser, MySupport, Software Manuals, Tech Tips, Forums, TechXclusives, Problem Solvers, WebCase, and Site Map. The main content area features a breadcrumb trail: Xilinx > TechXclusives > Reconfiguring Block RAMs. Below this is the article title 'Reconfiguring Block RAMs - Part 1' dated 07/17/2003, written by Kris Chaplin, a Senior Customer Applications Engineer at Xilinx UK. A small portrait of Kris Chaplin is shown next to his name. To the right of the article title are icons for 'e-mail this page', 'feedback', and 'print version'. The article begins with an 'Introduction' section, stating that one of the advantages of using a Xilinx FPGA is its complete reconfigurability. It explains that while a new bitstream is typically required for hardware changes, it's preferable to modify the design's contents without re-running synthesis or implementation. The article then discusses the specific modification of Xilinx block RAM contents in Spartan-3 and Virtex-5 Pro FPGAs. It notes that a designer might want to modify block RAM contents without re-routing the entire design, providing examples like modifying a Programmable State Machine or using a block RAM for coefficient storage in a DSP. Finally, it lists four possibilities for changing block RAM contents, with the first being 'Change the coefficients program in the source code', which involves instantiating block RAM components in VHDL/Verilog and using the CORE Generator system to create and regenerate the core.

