How To Use UJTAG

Introduction

UJTAG is an embedded macro for the ProASIC$^{PLUS®}$ and ProASIC®3 device families. It is implemented in unused I/O tiles and used as the interface between external JTAG ports and internal logic. This macro can be used to shift in data or OPCODEs through JTAG ports for the purpose of uploading some internal logic or controlling the internal functionality. For example, the UJTAG macro can be used to preload the ProASIC^{PLUS} RAM, using RAM to emulate ROM or using UJTAG to control the PLL for dynamic configuration. This macro can also be used to shift out data from internal logic to an external JTAG port for the purpose of driving LEDs, or it can be monitored by an oscilloscope as a test point. UJTAG is an extension of standard JTAG ports, passing data in or out with the help of a TAP (Test Access Port) controller. There are multiple Application Notes demonstrating specific applications using UJTAG. Links to these can be found in the reference section.

This document references a sample application to help illustrate the use of the UJTAG macro to shift in and shift out data as well as the Libero® Integrated Development Environment (IDE) design flow. The Design source files and a Libero IDE project are provided.

Sample Design

The sample design used in this application note is a flip-flop array in a ProASIC^{PLUS} device. The inputs of these three flip-flops are loaded by the TDI pin of the JTAG circuit through the UJTAG and UJTAG_Interface macros. The UJTAG_Interface macro contains a 3-bit Serial-In-Parallel-Out (SIPO) shift register, which starts to load the flip-flop series when a user defined UJTAG OPCODE is applied. The outputs of the flip-flops are routed to external pins to drive LEDs separately. When all three flip-flop outputs are '1', the AND3 gate will be turned on to drive another LED by the TDO pin via the UJTAG macro. Figure 1 shows the sample design.

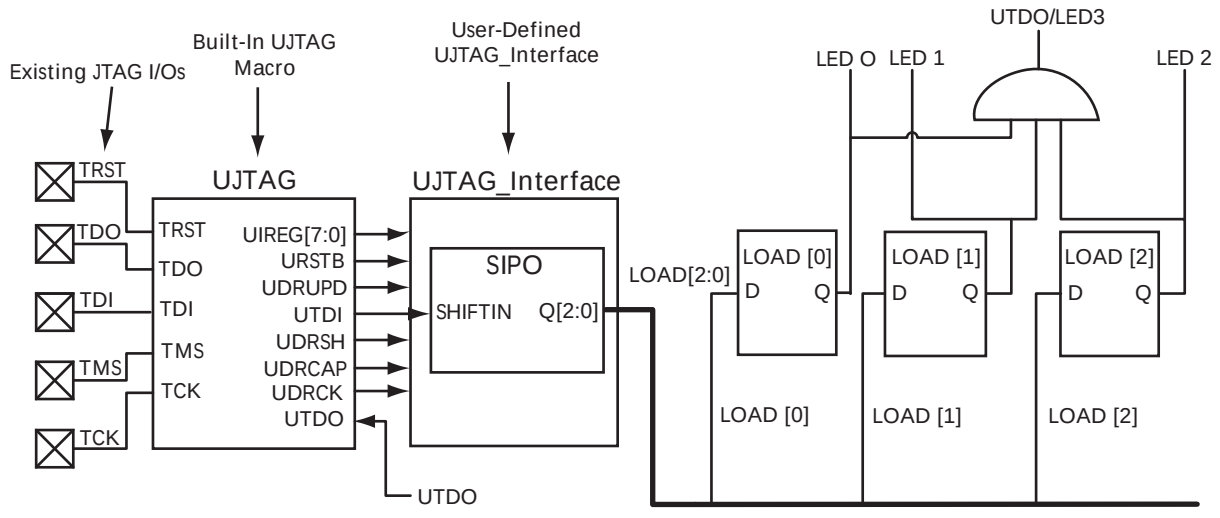

***Figure 1* • Top-Level Diagram of Sample Design**

© 2015 Microsemi Corporation

Design Implementation

As a unique macro, UJTAG can be instantiated in HDL design entry (as the sample code shows in "Appendix B" on page 7) or can be found within the ViewDraw™ actelcells library and connected in the schematic design.

For the sample design, instantiate the UJTAG macro in the top-level schematic as shown in Figure 2.

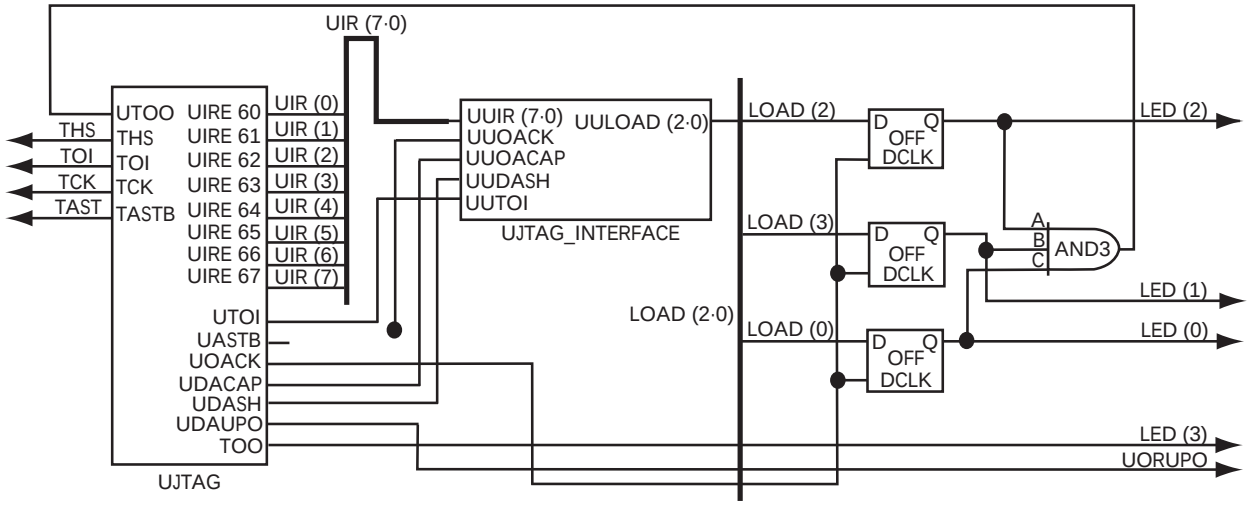

***Figure 2* • Libero IDE Top-Level Schematic Design**

Stimulus

Figure 3 demonstrates the TAP controller state machine.

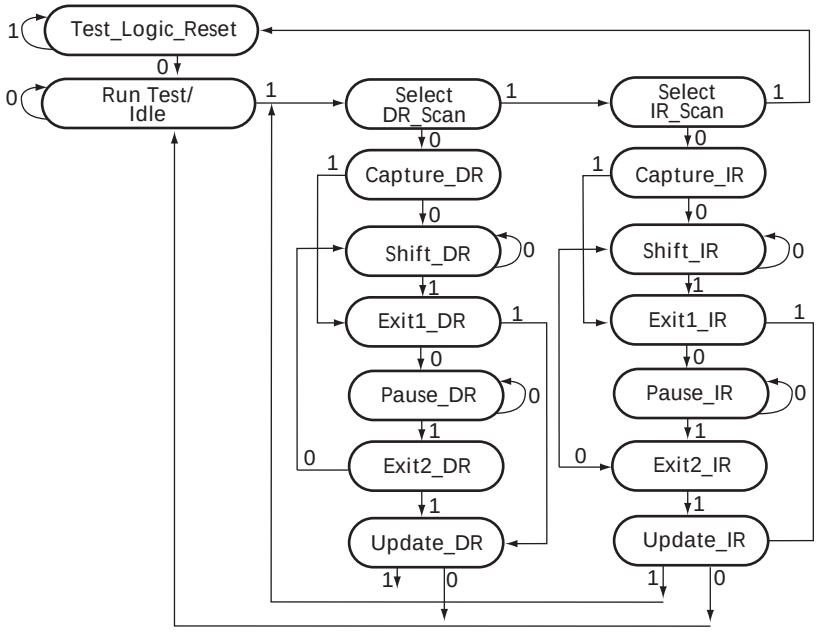

***Figure 3* • TAP Controller State Machine**

In order to run the simulation, an appropriate stimulus file must be generated to control the TAP controller state machine. Figure 4 shows the stimulus file used to run simulation of the sample design. The combination of TCK and TMS toggles can move the TAP controller into different states.

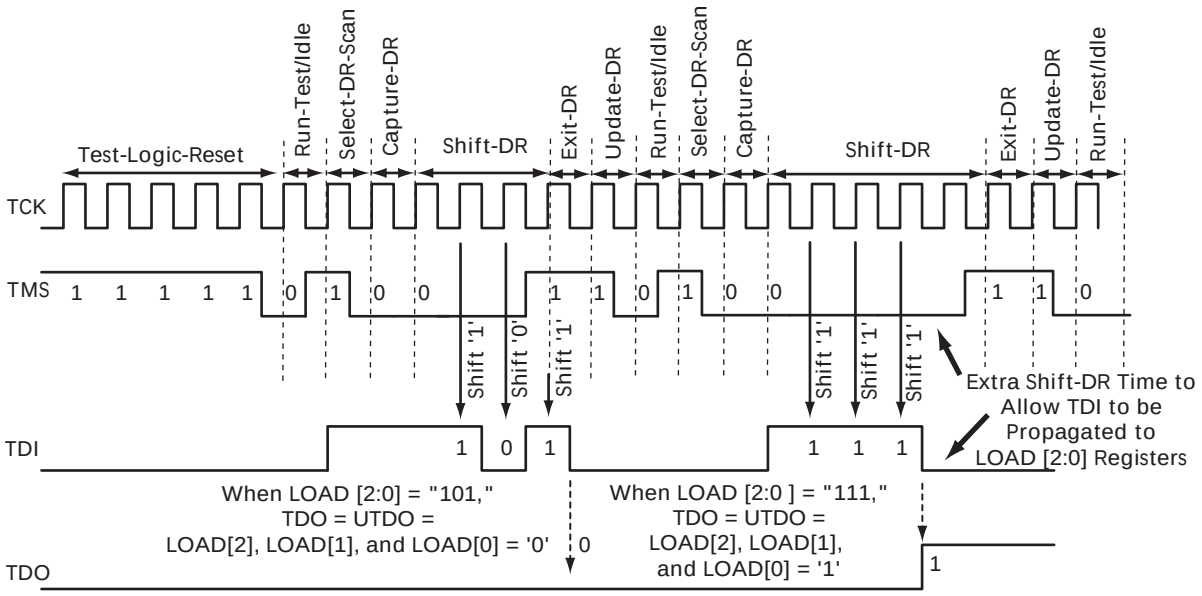

***Figure 4* • Stimulus for Sample Design**

The TCK and TMS signals generated in Figure 3 on page 2 are used to manipulate the TAP Controller State Machine (SM) to achieve the following functions:

1. The first five assertions of '1' on TMS are used to reset the SM to Test-Logic-Reset state.
2. The following '0' moves SM to Run-Test-Idle state.
3. An additional '1' sets SM to Select-DR-Scan state.
4. Followed by '00', SM is in Shift-DR state.
5. Shift in of n bits of TDI so TMS is kept at '0' for 'n-1' TCK cycles.
6. The following '1' on TMS, clocked by the rising edge of TCK, starts to transfer SM to Exit-DR state.
7. The following '10' puts SM back to the Run-Test-Idle state and there it waits for the next instruction.

According to the logic design of the sample design, if 111 is shifted in through TDI, then all 3 inputs of the AND3 gate will be driven to '1', and the output '1' will be shifted out through TDO.

The TCK and TMS signals are also used to control the SM to shift in OPCODEs to the Instruction Registers (IR) through the UJTAG shift register SHREG[7:0]. OPCODEs 16–127 are reserved for user-defined applications. In the sample design, OPCODE 18 (decimal) or 00010010 is used to start loading data from TDI. Figure 5 on page 4 shows the TCK/TMS/TDI signals used to shift in OPCODE 00010010 to initiate loading data into the flip-flop array. Since SHREG[7:0] = TDI & SHREG[7:1] (TDI concatenated with SHREG[7:1]), TDI should be shifted in LSB first.

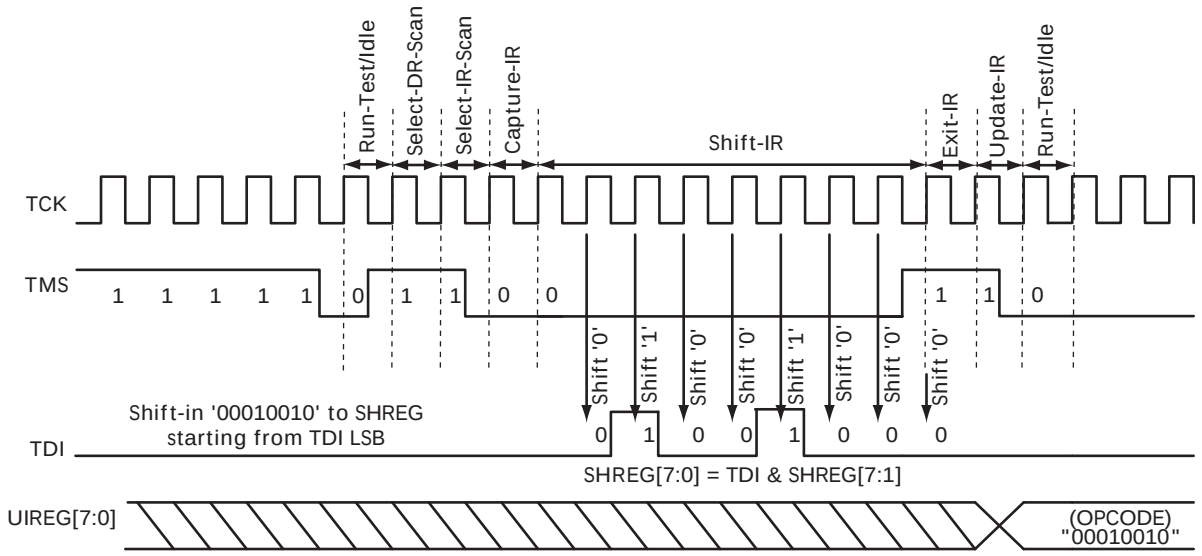

***Figure 5* • Use TCK/TMS/TDI Signals to Shift in OPCODE**

Simulation

Invoking Model*Sim* from the Libero IDE project will automatically execute the simulation. The simulation result appears, as shown in Figure 6.

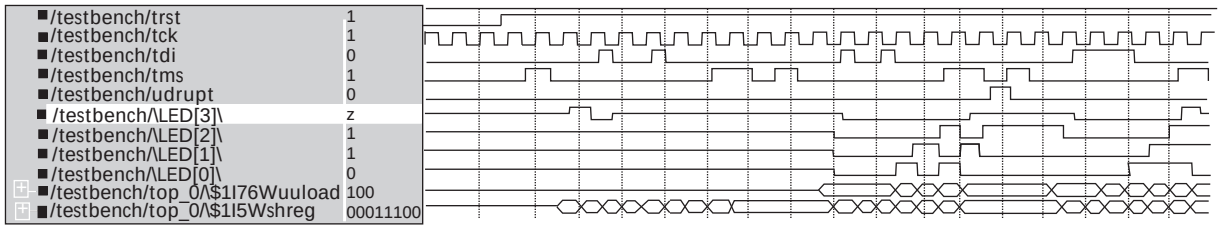

***Figure 6* • Simulation Result of Sample Application**

Synthesis and Layout

To complete the project, follow the Libero synthesis and layout design flow. For more details on the Libero design flow, refer to the *Libero IDE User's Guide*. The ADB file included with this project targets the APA evaluation board, so specific I/O constraints are applied. Refer to "Appendix C" on page 10 for more information. The TDO (LED[3]) signal is connected to pin 91 or DS3 on the eval board through a jumper wire.

On-Board Testing

This design targets the Microsemi APA Eval Board. After completing layout, generate the STAPL programming files to program the eval board with FlashPro Lite. FlashPro Lite can also be used to control the TAP controller state machine of the APA device. Another STAPL file, UJTAG_Eval.stp (used to control the TAP controller), can be found in the project, in the \how2useUJTAG\designer\impl1 folder. It contains instructions on TCK/TMS/TDI signals that are similar to those in the simulation stimulus file. The DRSCAN value inside the STAPL file can also be changed to achieve a different display on the LEDs. For more details, refer to "Appendix D" on page 11.

Summary

The UJTAG macro is an extension to the external JTAG port of the ProASIC^{PLUS} and ProASIC3 device families, controlled by the TAP controller. It can be used to shift in and shift out data/OPCODEs to and from the internal logic, PLL, and RAM block. Using the UJTAG macro in a design enables real-time updating and monitoring of the internal behavior of a Flash device.

Related Documents

ProASIC3 FPGA Fabric User's Guide, "UJTAG Applications in Microsemi's Low Power Flash *Devices" chapter*

ProASIC3 FPGA Fabric User's Guide

ProASICPLUS PLL Dynamic Reconfiguration Using JTAG

RAM Initialization and ROM Emulation in ProASICPLUS Devices

Appendix A

Libero IDE Project Design Hierarchy

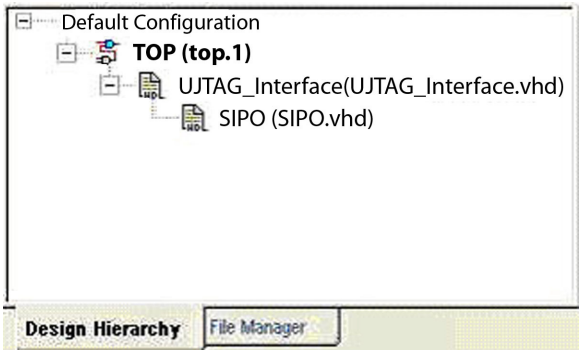

***Figure 7* • Libero IDE Design Hierarchy of the Sample Application**

Design Files Directory Hierarchy

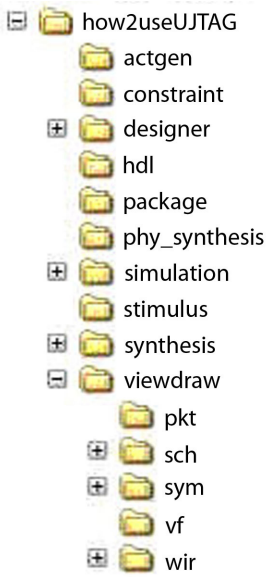

***Figure 8* • Design Files Directory Hierarchy**

Appendix B

Top Level HDL Code for Sample Application

```
-- Version: 6.1 SP1 6.1.1.108

library ieee;
use ieee.std_logic_1164.all;
library APA;

entity TOP is

port(\LED[2]\, \LED[1]\, \LED[0]\, UDRUPD, \LED[3]\ : out
         std_logic; TRST, TCK, TDI, TMS : in std_logic; SEL_ON :
        out std_logic);

end TOP;

architecture DEF_ARCH of TOP is

  component ujtag_interface
    port(UUIR : in std_logic_vector(7 downto 0); UUDRCK, UUDRCAP,
        UUDRSH, UUTDI : in std_logic; UULOAD :
        out std_logic_vector(2 downto 0); SEL_ON : out std_logic);
  end component;

  component DFF
    port(CLK, D : in std_logic; Q : out std_logic);
  end component;

  component UJTAG
    port(URSTB : out std_logic; UTDO : in std_logic; UDRCK,
        UDRCAP, UDRSH, UDRUPD, UTDI, UIREG0, UIREG1, UIREG2,
        UIREG3, UIREG4, UIREG5, UIREG6, UIREG7 : out std_logic;
        TDO, TMS, TDI, TCK, TRSTB : inout std_logic);
  end component;

  component AND3
    port(A, B, C : in std_logic; Y : out std_logic);
  end component;

    signal \$1N29\, \$1N60\, \$1N63\, \$1N66\, \$1N79\,
        \LED[0]_net_1\, \LED[1]_net_1\, \LED[2]_net_1\,
        \LED[3]_net_1\ : std_logic;
    signal LOAD : std_logic_vector(2 downto 0);
    signal \LOAD\, LOAD_net_1, \TCK\, \TDI\, \TMS\, \TRST\
         : std_logic;
    signal UIR : std_logic_vector(7 downto 0);
    signal \UIR\, UIR_net_1, UIR_net_2, UIR_net_3, UIR_net_4,
        UIR_net_5, UIR_net_6 : std_logic;

begin

    \LED[2]\ <= \LED[2]_net_1\;
    \LED[1]\ <= \LED[1]_net_1\;
    \LED[0]\ <= \LED[0]_net_1\;
    \LED[3]\ <= \LED[3]_net_1\;
    \TRST\ <= TRST;
    \TCK\ <= TCK;
    \TDI\ <= TDI;
    \TMS\ <= TMS;

    \$1I91\ : ujtag_interface
```

```
      port map(UUIR(7) => UIR_net_6, UUIR(6) => UIR(6), UUIR(5)
         => \UIR\, UUIR(4) => UIR_net_1, UUIR(3) => UIR_net_2,
        UUIR(2) => UIR_net_3, UUIR(1) => UIR_net_4, UUIR(0) =>
        UIR_net_5, UUDRCK => \$1N79\, UUDRCAP => \$1N63\, UUDRSH
         => \$1N66\, UUTDI => \$1N60\, UULOAD(2) => LOAD_net_1,
        UULOAD(1) => LOAD(1), UULOAD(0) => \LOAD\, SEL_ON =>
        SEL_ON);

    \$1I85\ : DFF
      port map(CLK => \$1N79\, D => \LOAD\, Q => \LED[0]_net_1\);

    \$1I5\ : UJTAG
      port map(URSTB => OPEN, UTDO => \$1N29\, UDRCK => \$1N79\,
        UDRCAP => \$1N63\, UDRSH => \$1N66\, UDRUPD => UDRUPD,
        UTDI => \$1N60\, UIREG0 => UIR_net_5, UIREG1 => UIR_net_4,
        UIREG2 => UIR_net_3, UIREG3 => UIR_net_2, UIREG4 =>
        UIR_net_1, UIREG5 => \UIR\, UIREG6 => UIR(6), UIREG7 =>
        UIR_net_6, TDO => \LED[3]_net_1\, TMS => \TMS\, TDI =>
        \TDI\, TCK => \TCK\, TRSTB => \TRST\);

    \$1I84\ : DFF
      port map(CLK => \$1N79\, D => LOAD(1), Q => \LED[1]_net_1\);

    \$1I6\ : AND3
      port map(A => \LED[2]_net_1\, B => \LED[1]_net_1\, C =>
        \LED[0]_net_1\, Y => \$1N29\);

    \$1I83\ : DFF
      port map(CLK => \$1N79\, D => LOAD_net_1, Q =>
        \LED[2]_net_1\);


end DEF_ARCH;
```

HDL Code for UJTAG_Interface

```
-- UJTAG_Interface.vhd
library ieee;
use ieee.std_logic_1164.all;
library apa;

entity UJTAG_Interface is
port (  uuir:        in std_logic_vector(7 downto 0);
        uudrck:     in std_logic;
        uudrcap:    in std_logic;
        uudrsh:     in std_logic;
        uutdi:      in std_logic;
        uuload:     out std_logic_vector(2 downto 0);
        sel_on:     out std_logic
        );
end UJTAG_Interface;

architecture Behav of UJTAG_Interface is

component SIPO
port (  Shiften : in std_logic;
        Shiftin : in std_logic;
        Aclr    : in std_logic;
        Clock   : in std_logic;
        Q       : out std_logic_vector(2 downto 0)) ;
end component;
```

```
signal uudrsh_in: std_logic;
signal uuir_in: std_logic_vector(7 downto 0);

signal uusrsel      : std_logic;
signal uuclr        : std_logic;
signal uushiftin       : std_logic;


begin

uuir_in  <= uuir;
uudrsh_in<= uudrsh;


inst_SIPO: SIPO
port map (  Shiften=>uudrsh_in,
            Shiftin=>uushiftin,
            Aclr=>uuclr,
            Clock=>uudrck,
            Q=>uuload(2 downto 0)) ;



uusrsel<= '1' when uuir_in="00010010" else '0'; -- Start loading when OPCODE=18;
sel_on <= uusrsel;

process (uudrcap) is
begin
    if (uudrcap='1' and uusrsel='1') then
            uuclr<='1';
    else
            uuclr<='0';
    end if;
end process;


process (uudrck) is
begin
    if (uudrck'event and uudrck='1') then
        if (uudrsh_in='1' and uusrsel='1') then
        uushiftin<=uutdi;
        else
        uushiftin<='0';
        end if;
    end if;
end process;



end Behav;
```

Appendix C

I/O Constraints

```
// Version: 6.1 SP1 6.1.1.108

//
// I/O constraints
//
set_io "96" "LED[0]";
set_io "94" "LED[2]";
set_io "95" "LED[1]";
set_io "90" "UDRUPD";
set_io "87" "SEL_ON";
```

Appendix D

STAPL Code for Sample Application

```
NOTE "CREATOR" "map2bitstream 5_2_0.1";
NOTE "DEVICE" "APA075";
NOTE "PACKAGE" "APA075-PQ208";
NOTE "DATE" "2005/04/22";
NOTE "STAPL_VERSION" "JESD71";
NOTE "IDCODE" "01A081CF";
NOTE "DESIGN" "UJTAG";
NOTE "CHECKSUM" "1111";
NOTE "SAVE_DATA" "BITSTREAM";
NOTE "AMHOME" "D:\Tools\Libero60\Designer/am";
NOTE "SECURITY" "DISABLE";
NOTE "ALG_VERSION" "16";
NOTE "MAX_FREQ" "10000000";
NOTE "TRACKING_SAR" "34881";

ACTION LOADTDI  = LOADZB;

DATA PARAMETERS;
BOOLEAN ULOP=1;
INTEGER freq = 4;   ' RCK frequency 1-4 MHz
BOOLEAN USE_RCK=0;
ENDDATA;

PROCEDURE LOADZB USES PARAMETERS;

WAIT RESET, 5 CYCLES;
IRSCAN 8,$12;  fl shift IR, SHREG=00010010
DRSCAN 8, #00000001;fl turn on LED[2]

WAIT 1000000 USEC;
DRSCAN 8, #00000010;fl turn on LED[1]

WAIT 1000000 USEC;
DRSCAN 8, #00000011;fl turn on LED[2],LED[1]

WAIT 1000000 USEC;
DRSCAN 8, #00000100;fl turn on LED[0]

WAIT 1000000 USEC;
DRSCAN 8, #00000101;fl turn on LED[2],LED[0]

WAIT 1000000 USEC;
DRSCAN 8, #00000110;fl turn on LED[1],LED[0]

WAIT 1000000 USEC;
DRSCAN 8, #00000111;fl turn on LED[2],LED[1],LED[0]. LED[3] is also turned on under this
condition.
ENDPROC;

CRC 0000;
```

List of Changes

The following table shows important changes made in this document for each revision.

Revision*	Changes	Page
Revision 1 (April 2015)	Non-technical Updates.	N/A
Revision 0 (July 2005)	Initial Release.	N/A

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

© 2015 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

51900101-01/04.15