



KC705 MultiBoot Design

July 2012

XTP104

Revision History

Date	Version	Description
07/25/12	3.0	Recompiled for 14.2. Added AR50886.
05/08/12	2.0	Recompiled for 14.1.
01/18/12	1.0	Initial version for 13.4.

© Copyright 2012 Xilinx, Inc. All Rights Reserved.

XILINX, the Xilinx logo, the Brand Window and other designated brands included herein are trademarks of Xilinx, Inc. All other trademarks are the property of their respective owners.

NOTICE OF DISCLAIMER: The information disclosed to you hereunder (the “Information”) is provided “AS-IS” with no warranty of any kind, express or implied. Xilinx does not assume any liability arising from your use of the Information. You are responsible for obtaining any rights you may require for your use of this Information. Xilinx reserves the right to make changes, at any time, to the Information without notice and at its sole discretion. Xilinx assumes no obligation to correct any errors contained in the Information or to advise you of any corrections or updates. Xilinx expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE INFORMATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD-PARTY RIGHTS.

Overview

- Kintex-7 MultiBoot Capability
- Overview of 7 Series Fallback MultiBoot
- Xilinx KC705 Board
- Software Requirements
- KC705 Setup
- Compiling the MultiBoot Design
- Run MultiBoot Design
- Kintex-7 MultiBoot Details
- References

Kintex-7 MultiBoot Capability

► What is MultiBoot?

- The MultiBoot feature allows the FPGA application to load two or more FPGA bitstreams under the control of the FPGA application. The FPGA application triggers a MultiBoot operation, causing the FPGA to reconfigure from a different configuration bitstream. After a MultiBoot operation is triggered, the FPGA restarts its configuration process as usual.
- More details can be found in [UG470](#)

► MultiBoot Capability

- FPGA application controlled configuration
- Bitstream selection of multiple applications

► Safe Update

- Golden bitstream
- Upgradeable bitstream
- Failure recovery
 - Possible Triggers (CRC error, IDCODE error, WDT timeout)

Kintex-7 MultiBoot Capability

► Overview of 7 Series Fallback MultiBoot

- The 7 series FPGAs MultiBoot and fallback features support updating systems in the field. Bitstream images can be upgraded dynamically in the field. The FPGA MultiBoot feature enables switching between images on the fly. When an error is detected during the MultiBoot configuration process, the FPGA can trigger a fallback feature that ensures a known good design can be loaded into the device.
- The MultiBoot feature allows the FPGA application to load two or more FPGA bitstreams under the control of the FPGA application. The FPGA application triggers a MultiBoot operation, causing the FPGA to reconfigure from a different configuration bitstream. After a MultiBoot operation is triggered, the FPGA restarts its configuration process as usual.

► Reference Design Source and Application

- rdf0104.zip
- Available through <http://www.xilinx.com/kc705>

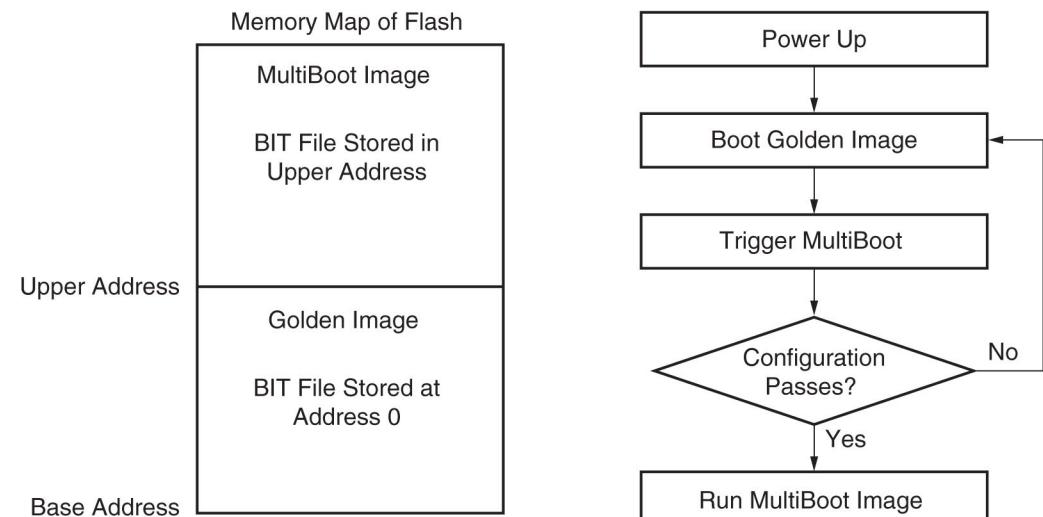
KC705 MultiBoot Design Description

► Description

- Design consists of three bitstreams, golden_iprog_spi.bit, multiboot.bit, and corrupted.bit loaded in the PROM at 0x0, 0x400000, and 0x800000
- The golden_iprog_spi.bit has software to allow a reboot to the second or third bitstreams with certain switch settings
- The multiboot.bit is compiled with “-g ConfigFallback:Enable” bitgen option
- The corrupted.bit is the multiboot.bit with one byte changed to force a CRC error when loaded

► This diagram shows the general process

- Since the corrupted.bit doesn't pass configuration, the Golden Image is booted



Upgrade Example

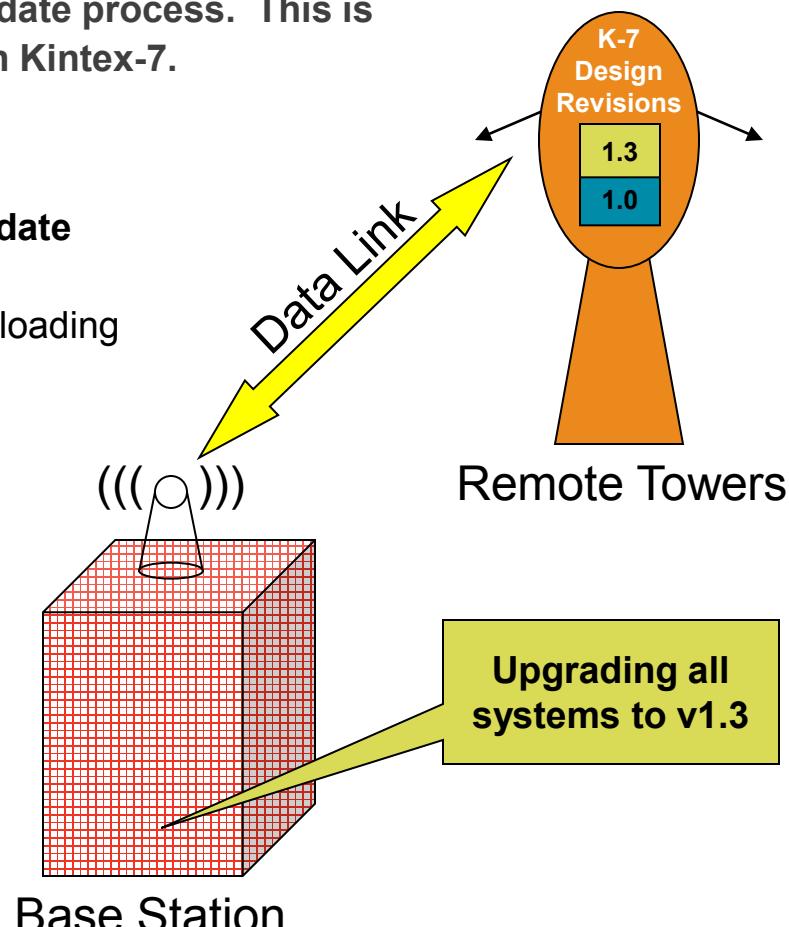
- **MultiBoot:** Process by which the FPGA selectively reprograms and reloads its bitstream from an attached external memory.
- **Safe update:** Field updating bitstream storage with a new bitstream in such a manner to prevent any failure due to a failure in the update process. This is accomplished with the enhanced MultiBoot available in Kintex-7.

Can a multi-boot system be implemented without safe update design considerations?

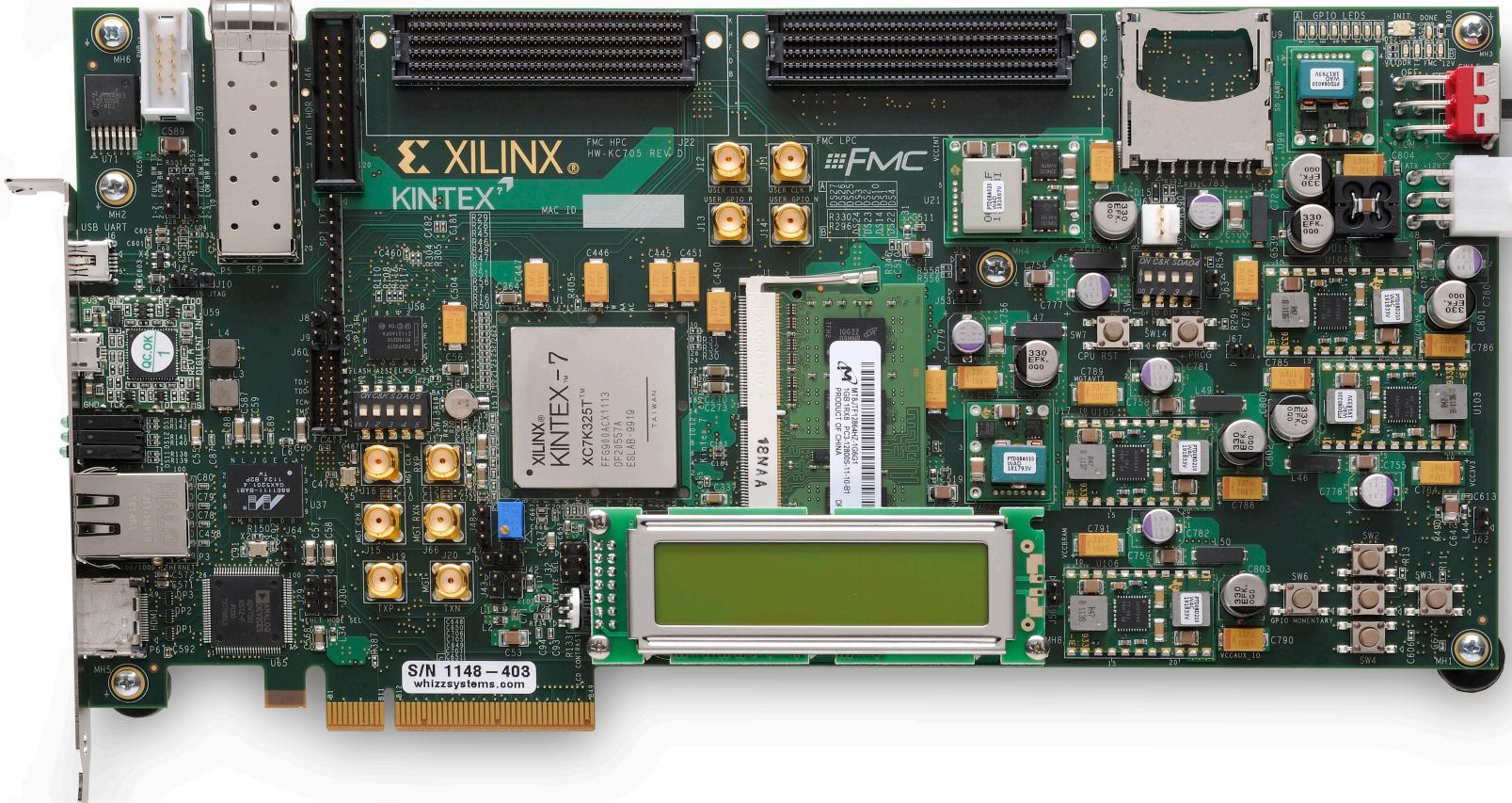
Yes – If there are no potential for disruptions during flash loading

How is a system upgraded?

1. New multi-boot image is created
2. System setup to receive the new image
3. User application erases section of Flash
4. The new image is delivered into the system's Flash
5. User application resets system



Xilinx KC705 Board



Note: Presentation applies to the KC705

XILINX

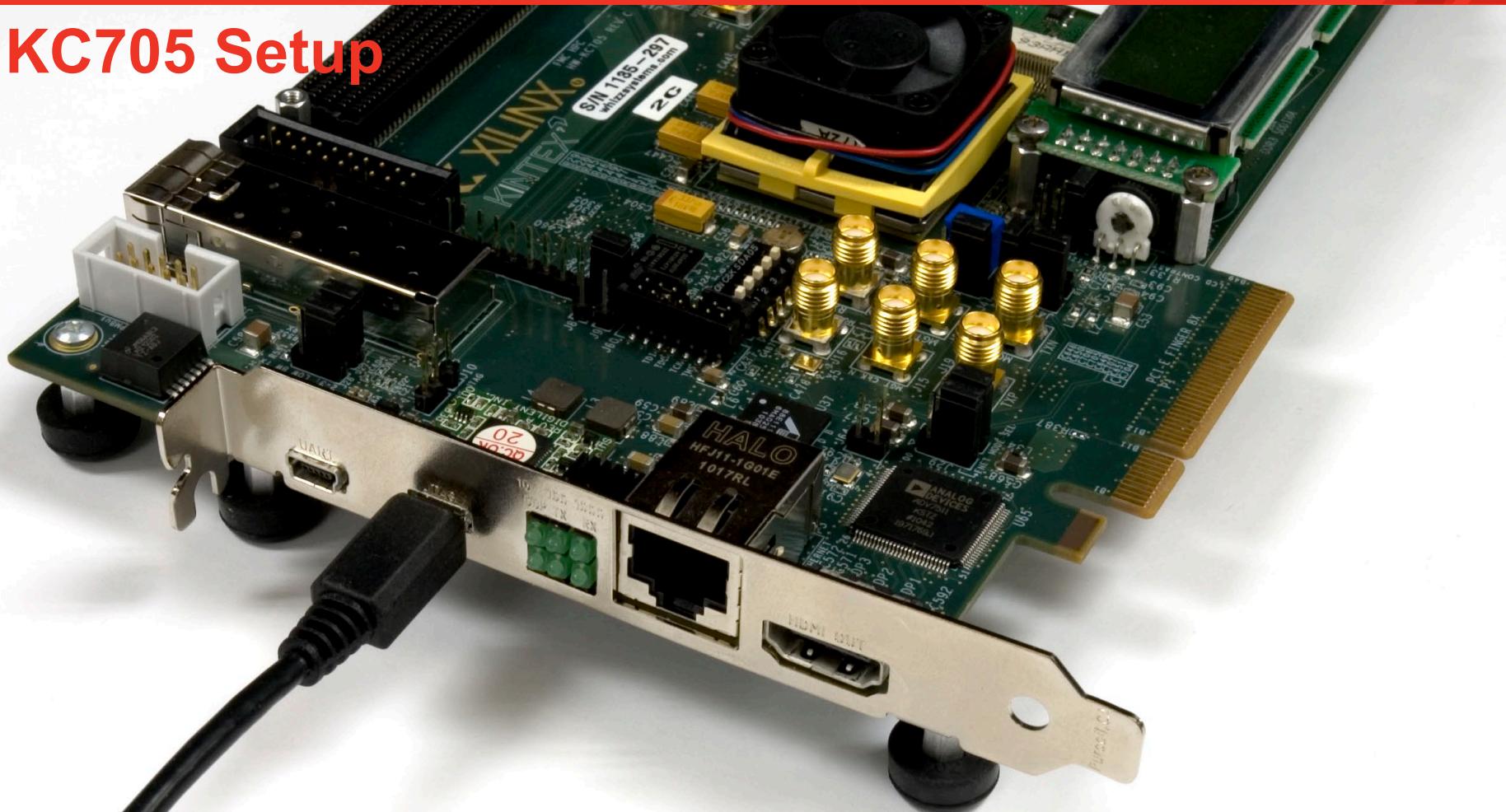
ISE Software Requirement

► Xilinx ISE 14.2 software

- Apply [AR50886](#)



KC705 Setup



- **Connect a USB Type-A to Micro-B cable to the USB JTAG (Digilent) connector on the KC705 board**
 - Connect this cable to your PC
 - Power on the KC705 board

Hardware Setup

► Set S13 to 00001 (1 = on, Position 1 → Position 5)

- This enables Master SPI configuration from the QSPI Flash
 - Flash A25, A24 = 00
 - FPGA mode pins M[2:0] = 001

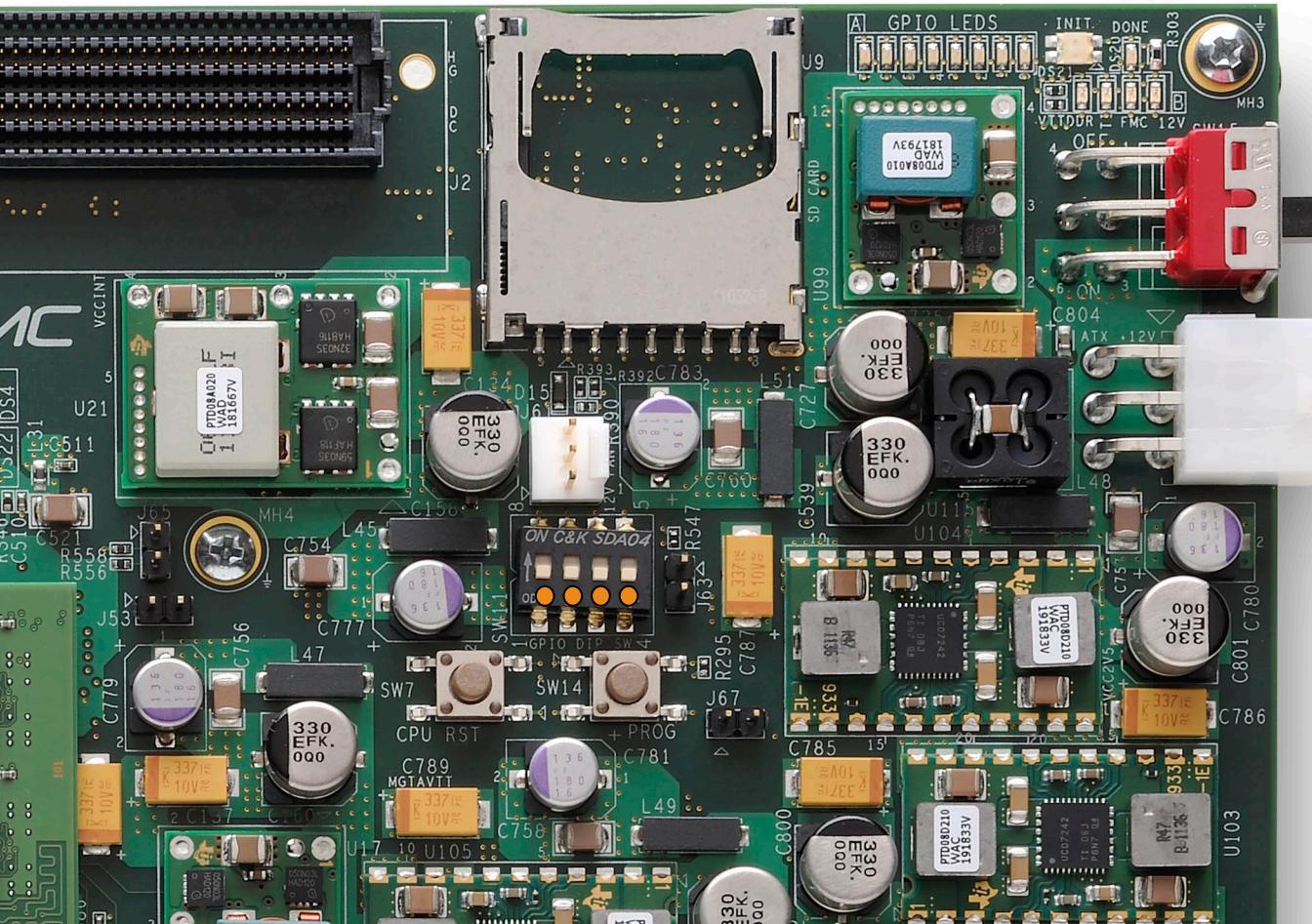


Note: Presentation applies to the KC705

 XILINX ➤ ALL PROGRAMMABLE™

Hardware Setup

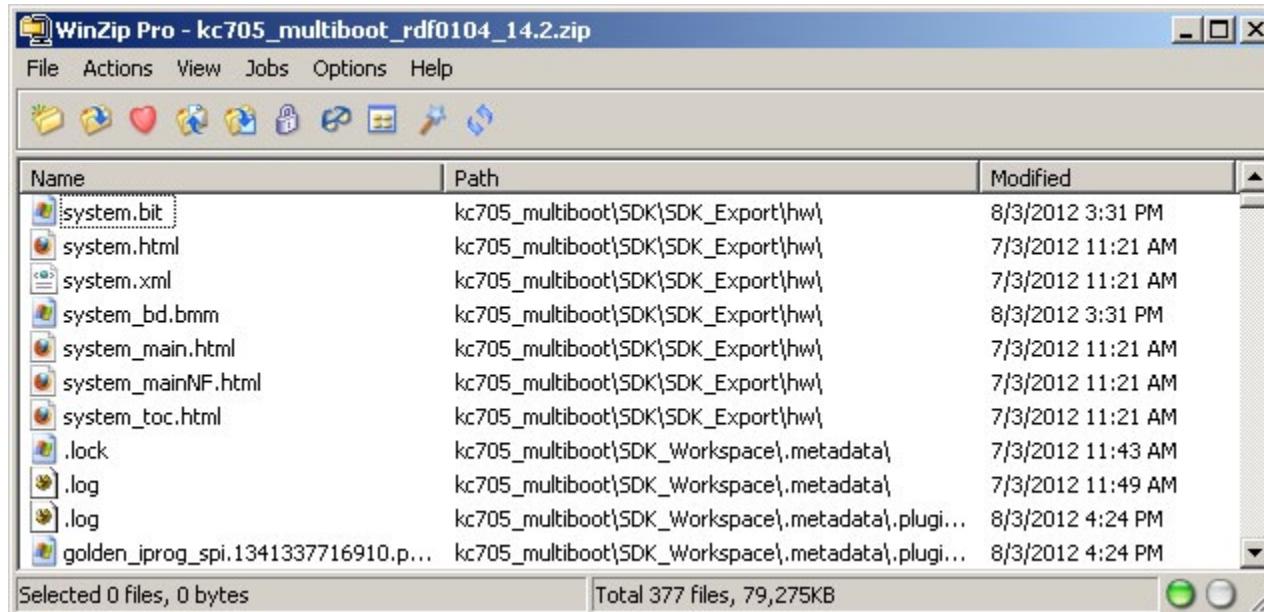
- ▶ Set S11 to 0000 (1 = on, Position 1 → Position 4)
 - This sets the MultiBoot functions to off



Program SPI MultiBoot Design

► Unzip the KC705 Multiboot Design Files (14.2 CES)

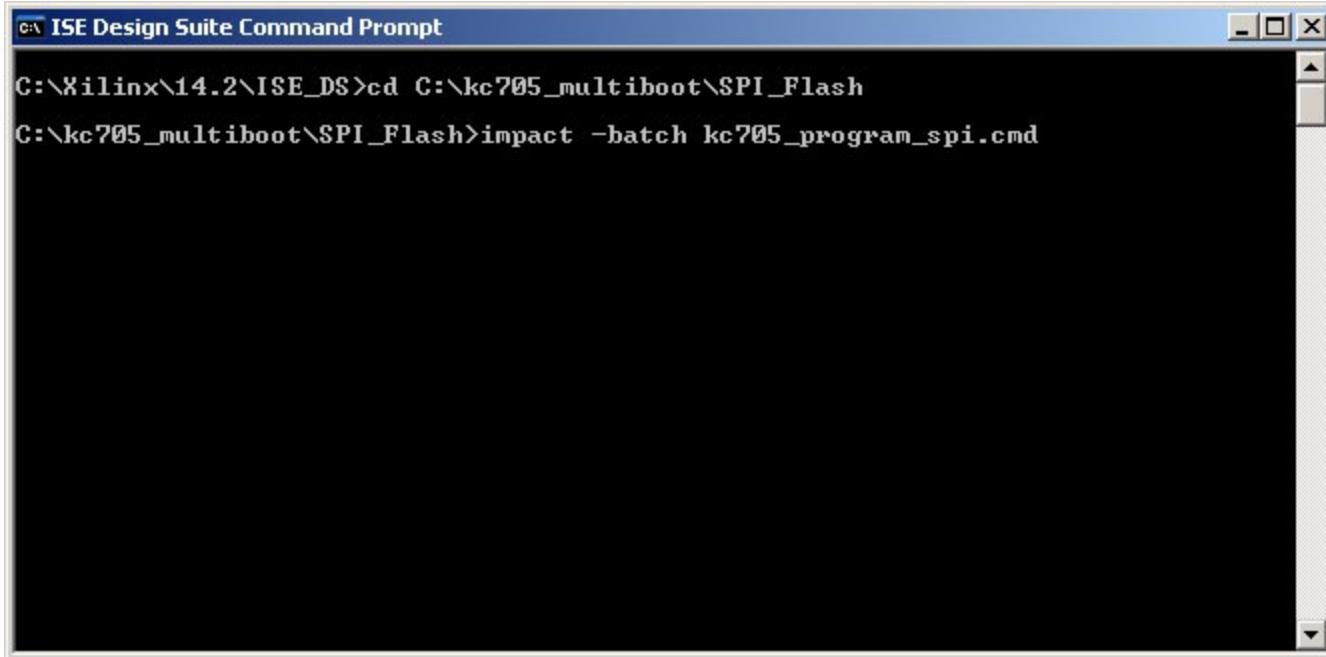
- Available through <http://www.xilinx.com/kc705>



Program SPI MultiBoot Design

- Open an ISE Design Suite Command Prompt and type:

```
cd C:\kc705_multiboot\SPI_Flash  
impact -batch kc705_program_spi.cmd
```

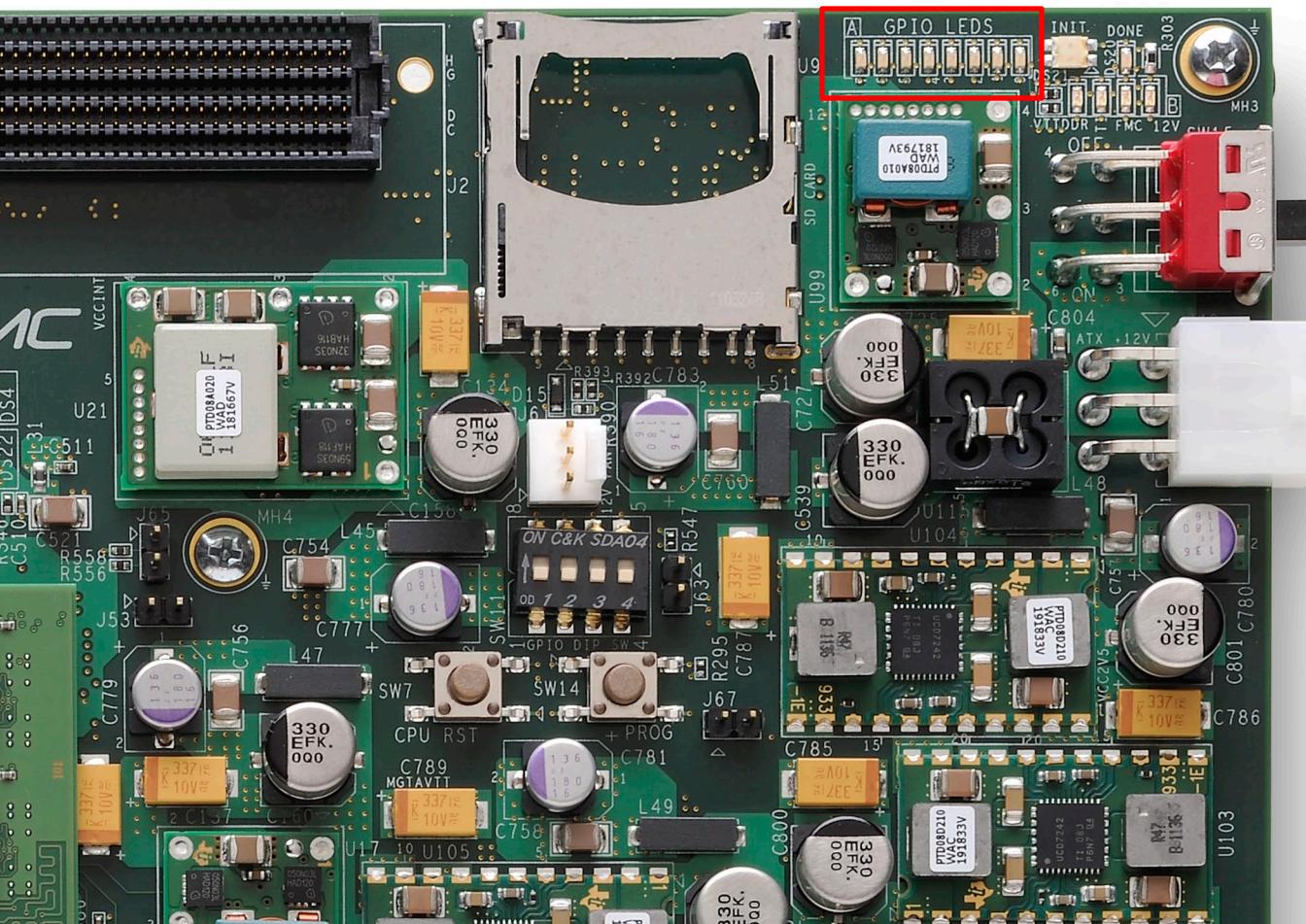


The screenshot shows a Windows Command Prompt window titled "ISE Design Suite Command Prompt". The window contains the following text:

```
C:\Xilinx\14.2\ISE_DS>cd C:\kc705_multiboot\SPI_Flash  
C:\kc705_multiboot\SPI_Flash>impact -batch kc705_program_spi.cmd
```

Run MultiBoot Design

- The initial design, “golden_iprog_spi.bit”, shows a cycling LED pattern on the GPIO LEDs



Run MultiBoot Design

► Run iMPACT:

impact



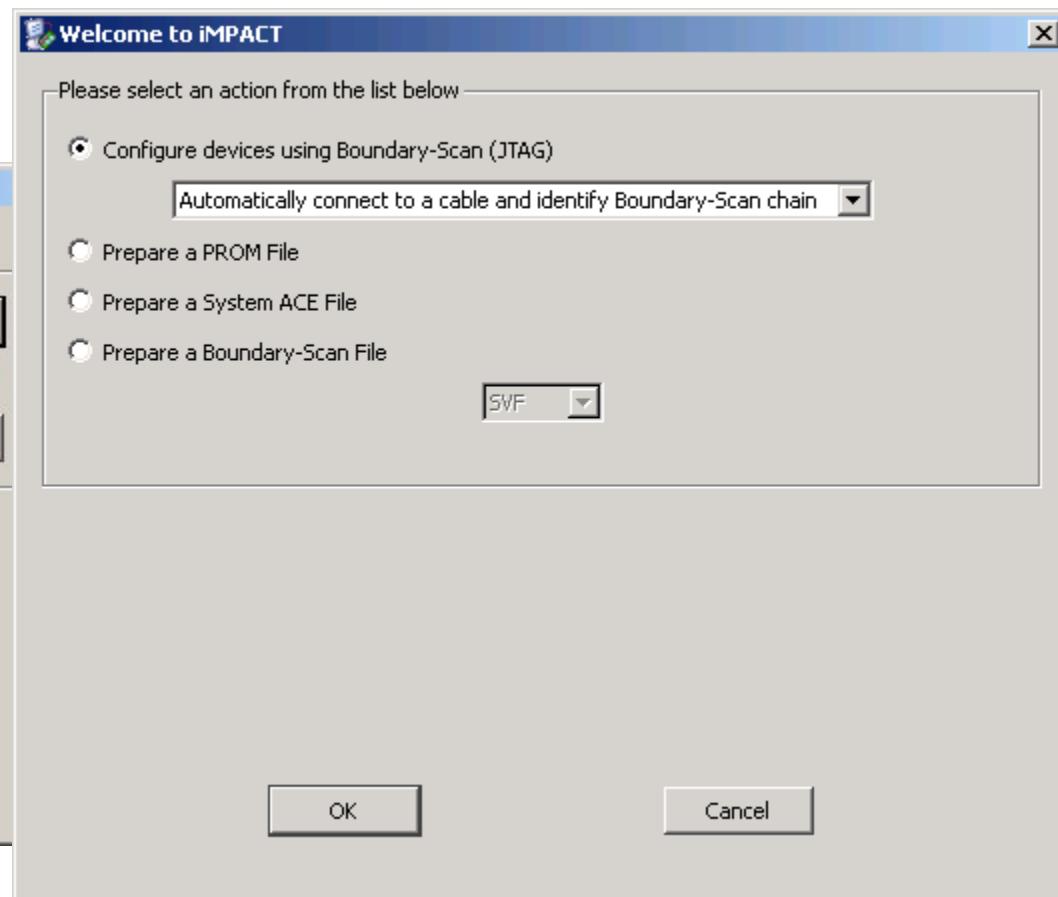
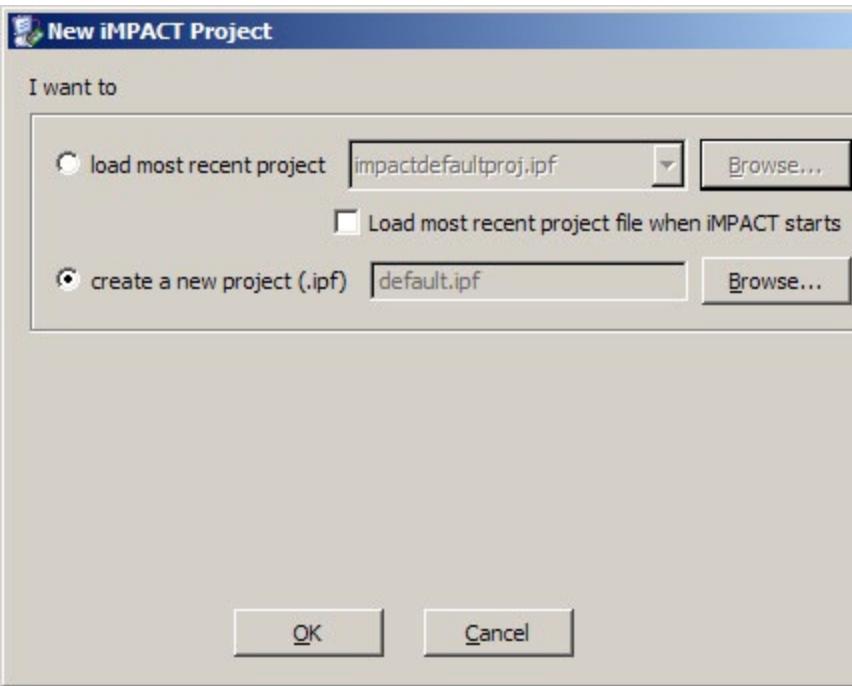
```
C:\ISE Design Suite Command Prompt
Match cycle: NoWait
LCK_cycle = NoWait.
LCK cycle: NoWait
done.
INFO:Cse - Status register values:
INFO:Cse - 0011 1111 1001 1110 0000 1000 0000 0010
INFO:Cse - '1': Completed downloading bit file to device.
INFO:Cse - '1': Programming completed successfully.
key: period_frc, value: 0
key: dclk_has_reset, value: 0
key: period_int, value: 10
Found Slave on Bus Index.
Found Slave on Bus Index.
SPI core clock speed value = 0xA801.
'1': IDCODE is '20ba18' (in hex).
'1': ID Check passed.
'1': Erasing Device.
'1': Using Sector Erase.
'1': Programming Flash.
'1': Programming in x1 mode.
INFO:iMPACT - '1': Checking done pin....done.
'1': Programmed successfully.
Elapsed time =      630 sec.

C:\kc705_multiboot\SPI_Flash>impact
```

Run MultiBoot Design

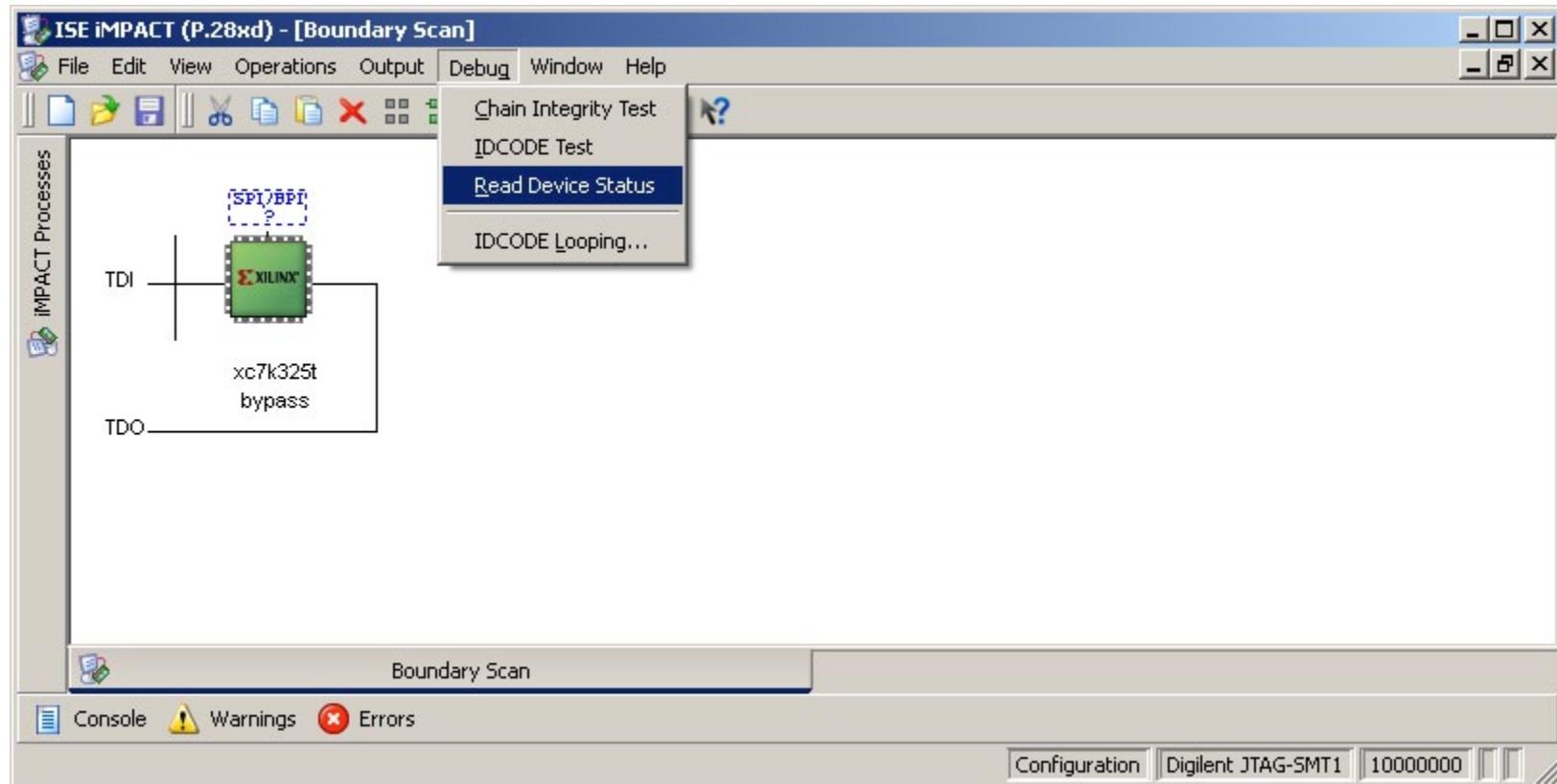
► Select

- Create a new project
- Configure devices using Boundary-Scan (JTAG)



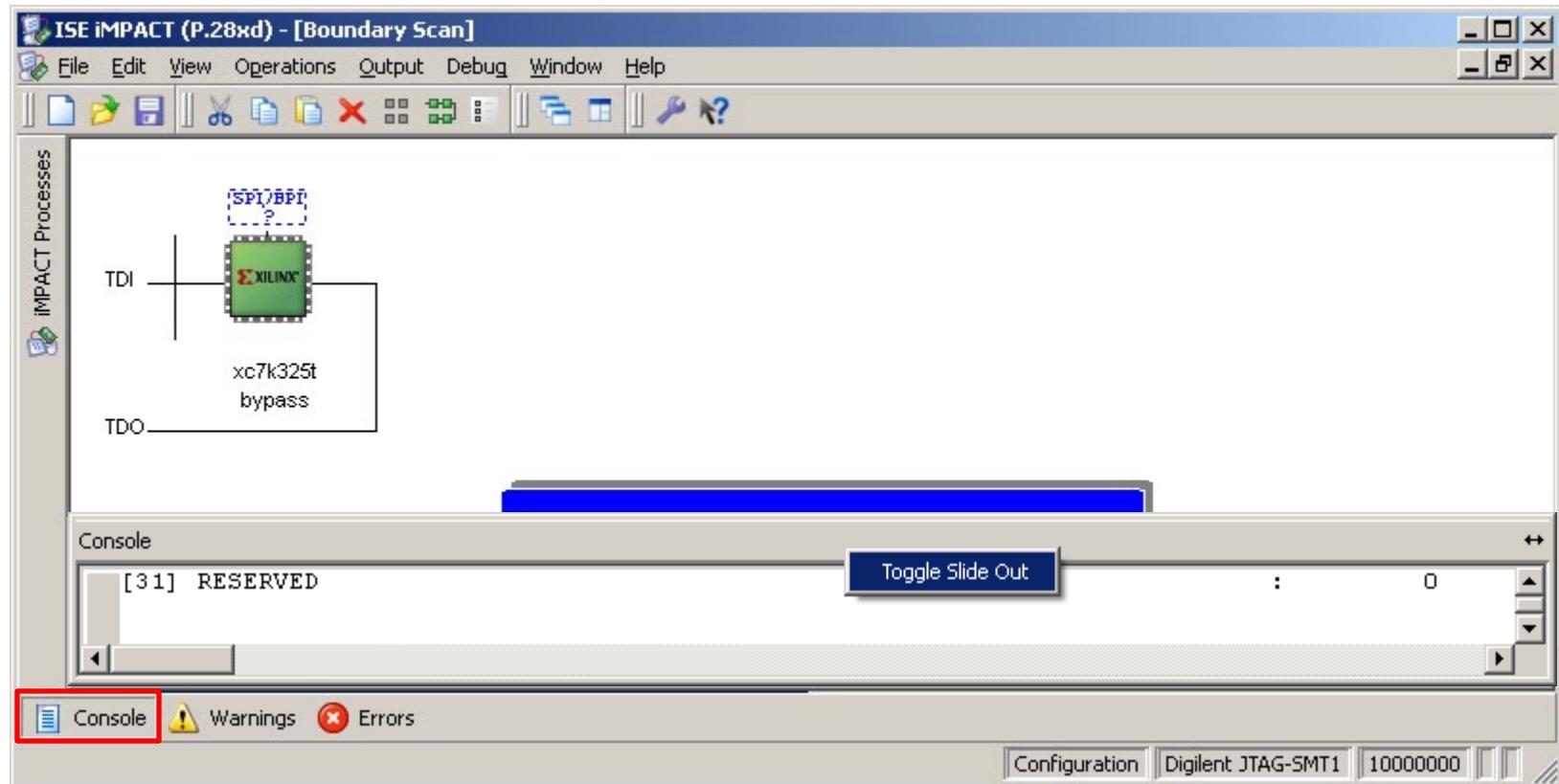
Run MultiBoot Design

- Select the FPGA
- Select the menu item Debug → Read Device Status



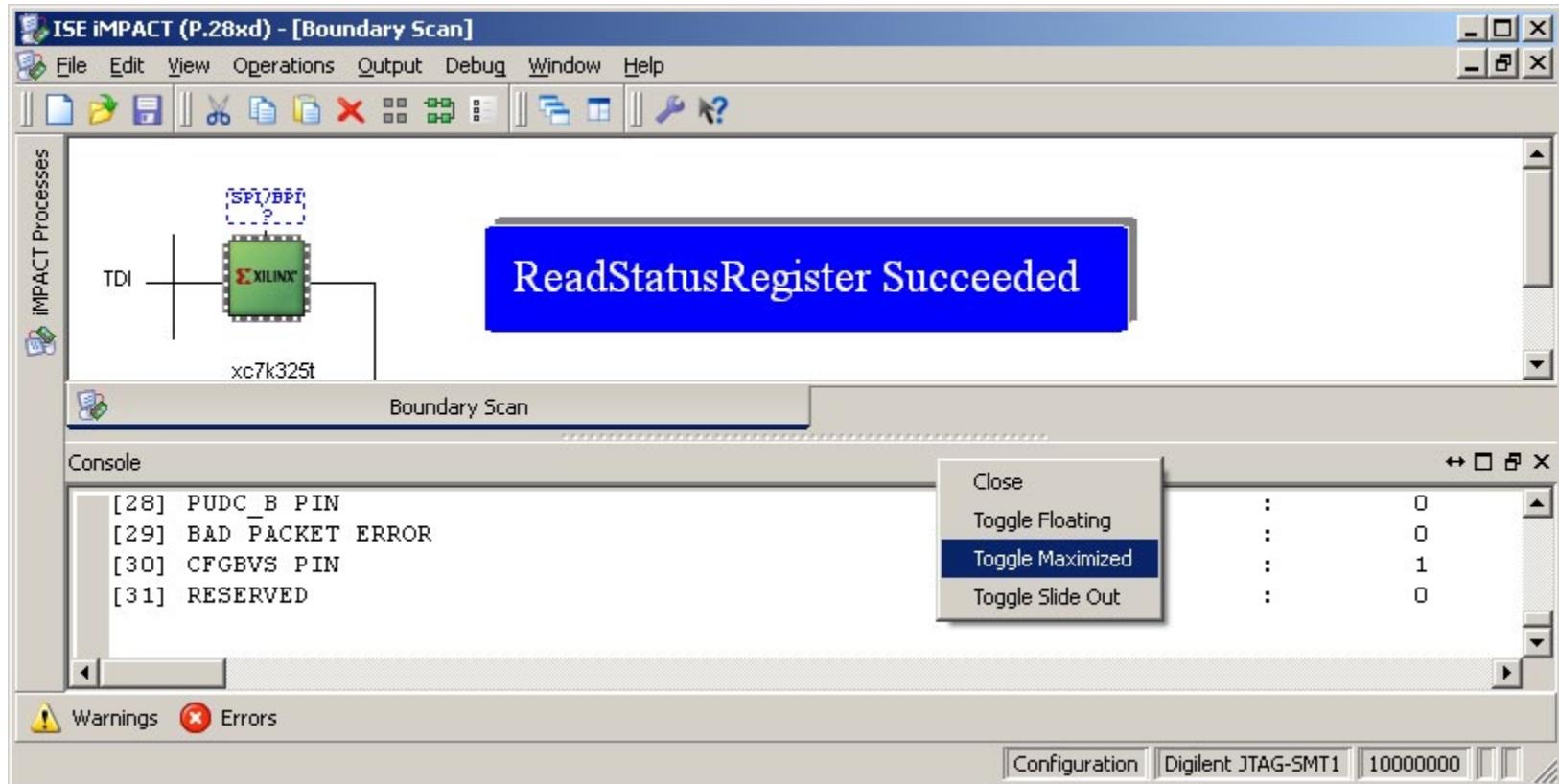
Run MultiBoot Design

- Click console tab and then right click on the console panel and select “Toggle Slide Out”



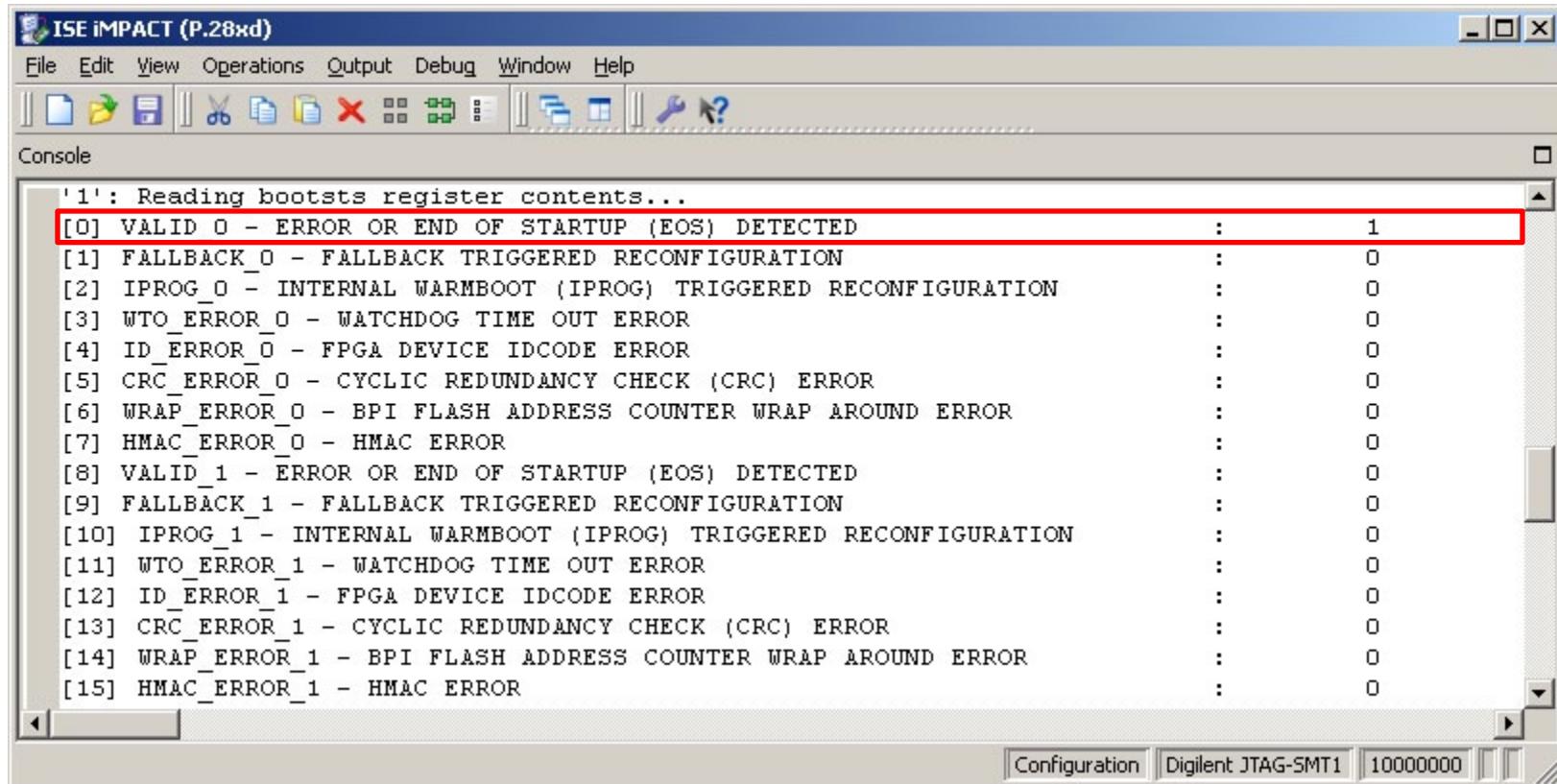
Run MultiBoot Design

- Right click on the Console panel and select Toggle Maximized



Run MultiBoot Design

- Scroll to the “Reading bootsts register contents...” message
- Since this was a regular boot, only “VALID_0” is set



The screenshot shows the ISE iMPACT software interface with a "Console" tab selected. The window title is "ISE iMPACT (P.28xd)". The menu bar includes File, Edit, View, Operations, Output, Debug, Window, and Help. The toolbar below the menu has icons for file operations like Open, Save, and Close, as well as configuration and debug tools. The main console area displays the following text:

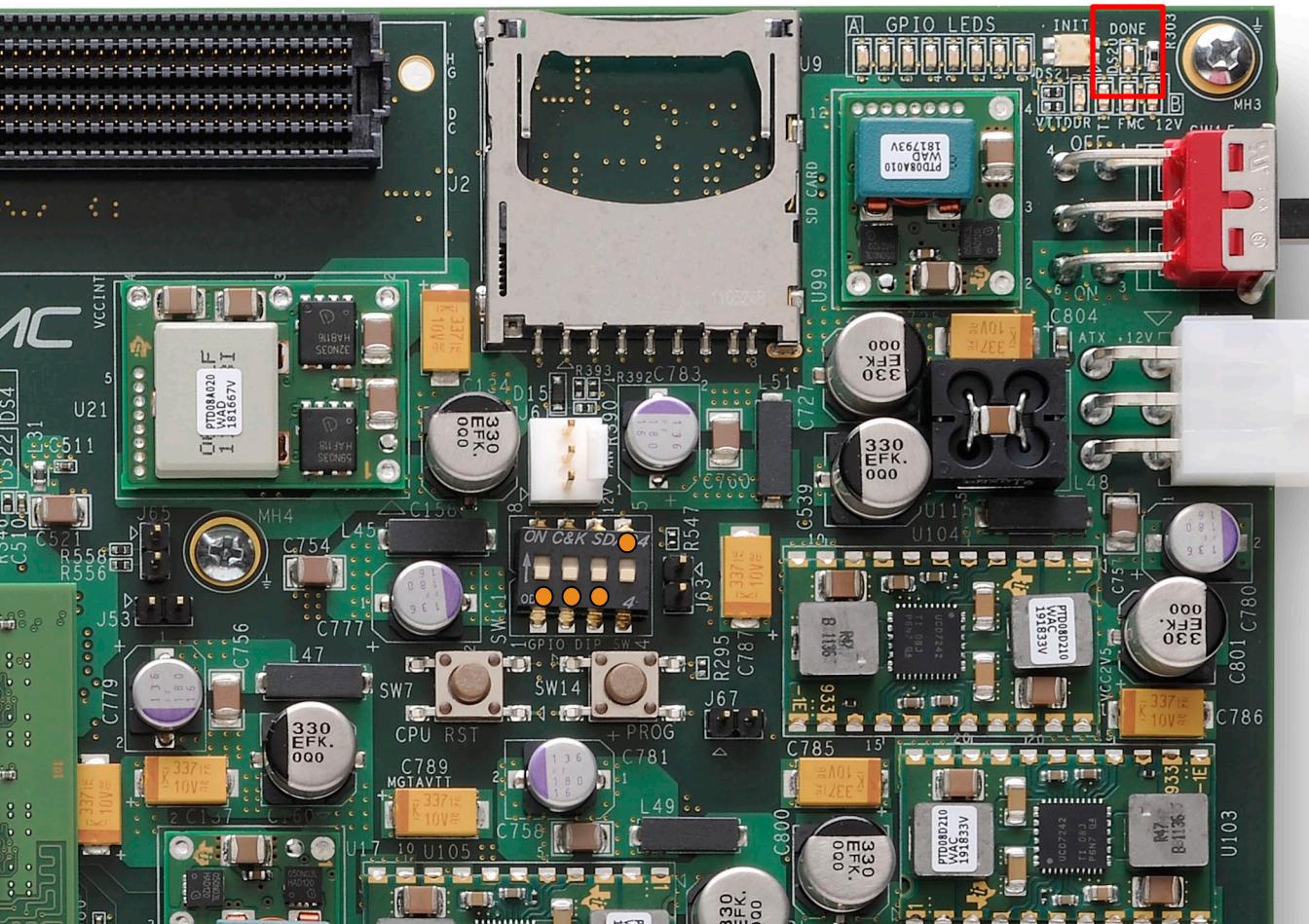
```
'1': Reading bootsts register contents...
[0] VALID_0 - ERROR OR END OF STARTUP (EOS) DETECTED : 1
[1] FALBACK_0 - FALBACK TRIGGERED RECONFIGURATION : 0
[2] IPROG_0 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 0
[3] WTO_ERROR_0 - WATCHDOG TIME OUT ERROR : 0
[4] ID_ERROR_0 - FPGA DEVICE IDCODE ERROR : 0
[5] CRC_ERROR_0 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 0
[6] WRAP_ERROR_0 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[7] HMAC_ERROR_0 - HMAC ERROR : 0
[8] VALID_1 - ERROR OR END OF STARTUP (EOS) DETECTED : 0
[9] FALBACK_1 - FALBACK TRIGGERED RECONFIGURATION : 0
[10] IPROG_1 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 0
[11] WTO_ERROR_1 - WATCHDOG TIME OUT ERROR : 0
[12] ID_ERROR_1 - FPGA DEVICE IDCODE ERROR : 0
[13] CRC_ERROR_1 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 0
[14] WRAP_ERROR_1 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[15] HMAC_ERROR_1 - HMAC ERROR : 0
```

At the bottom of the console window, there are tabs for Configuration, Digilent JTAG-SMT1, and a clock frequency of 10000000 Hz.

Run MultiBoot Design

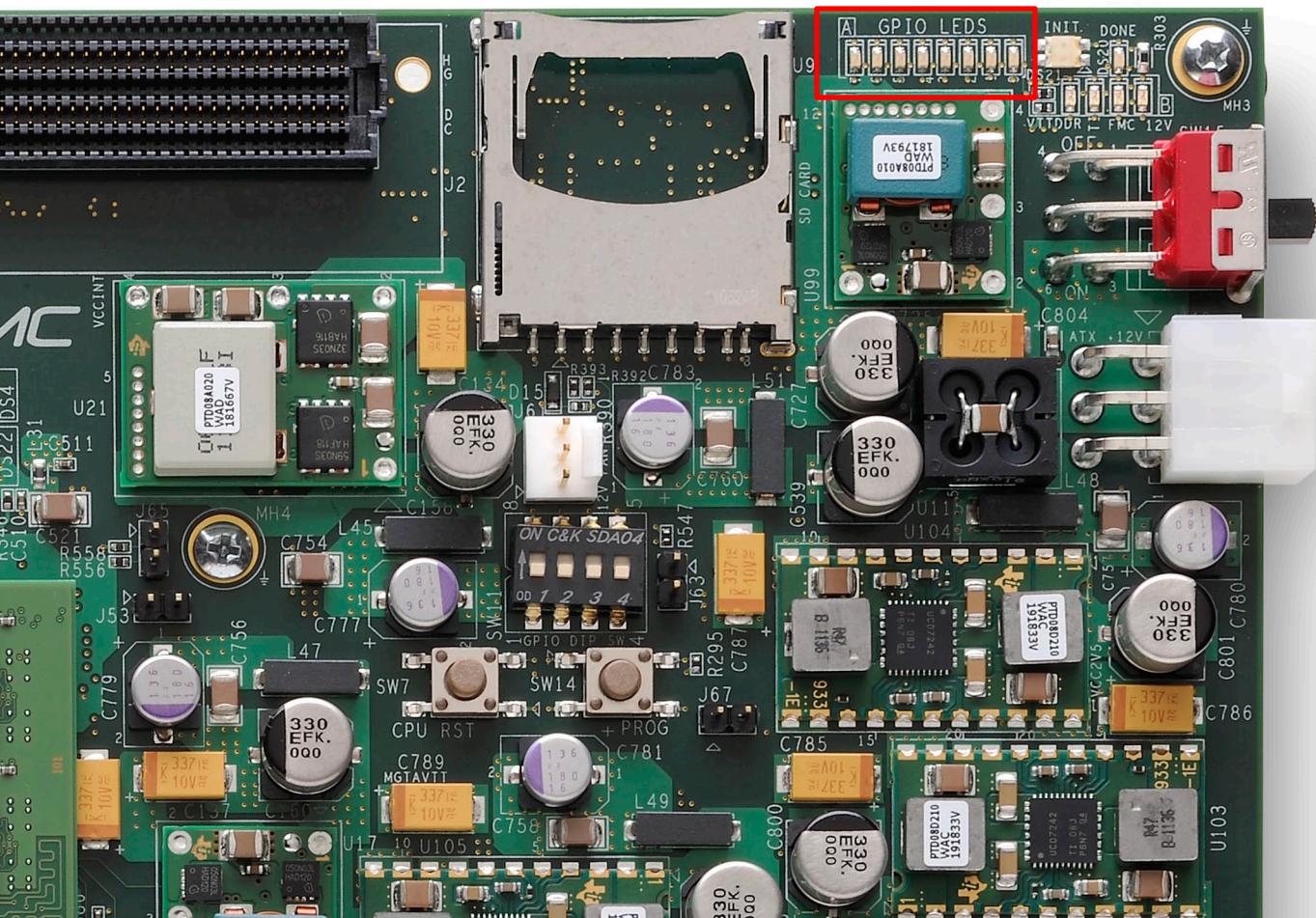
► Set S11 to 0001 (1 = on, Position 1 → Position 4)

- Issues an IPROG command to re-configure from a good MultiBoot bitstream
 - Set it back to 0000 when the DONE LED goes out



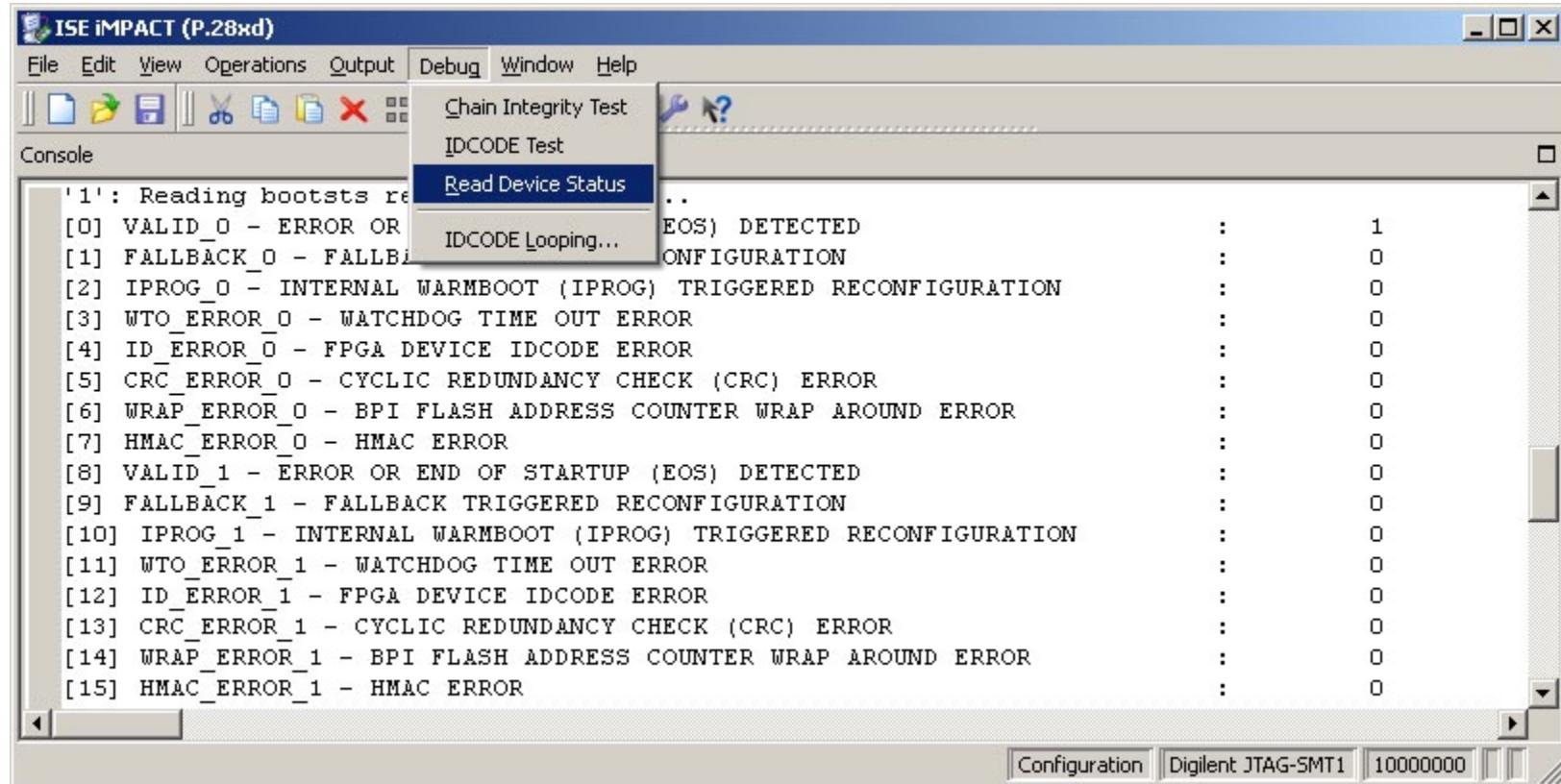
Run MultiBoot Design

- The MultiBoot design, “multiboot.bit”, shows a rapid blinking LED pattern on the GPIO LEDs



Run MultiBoot Design

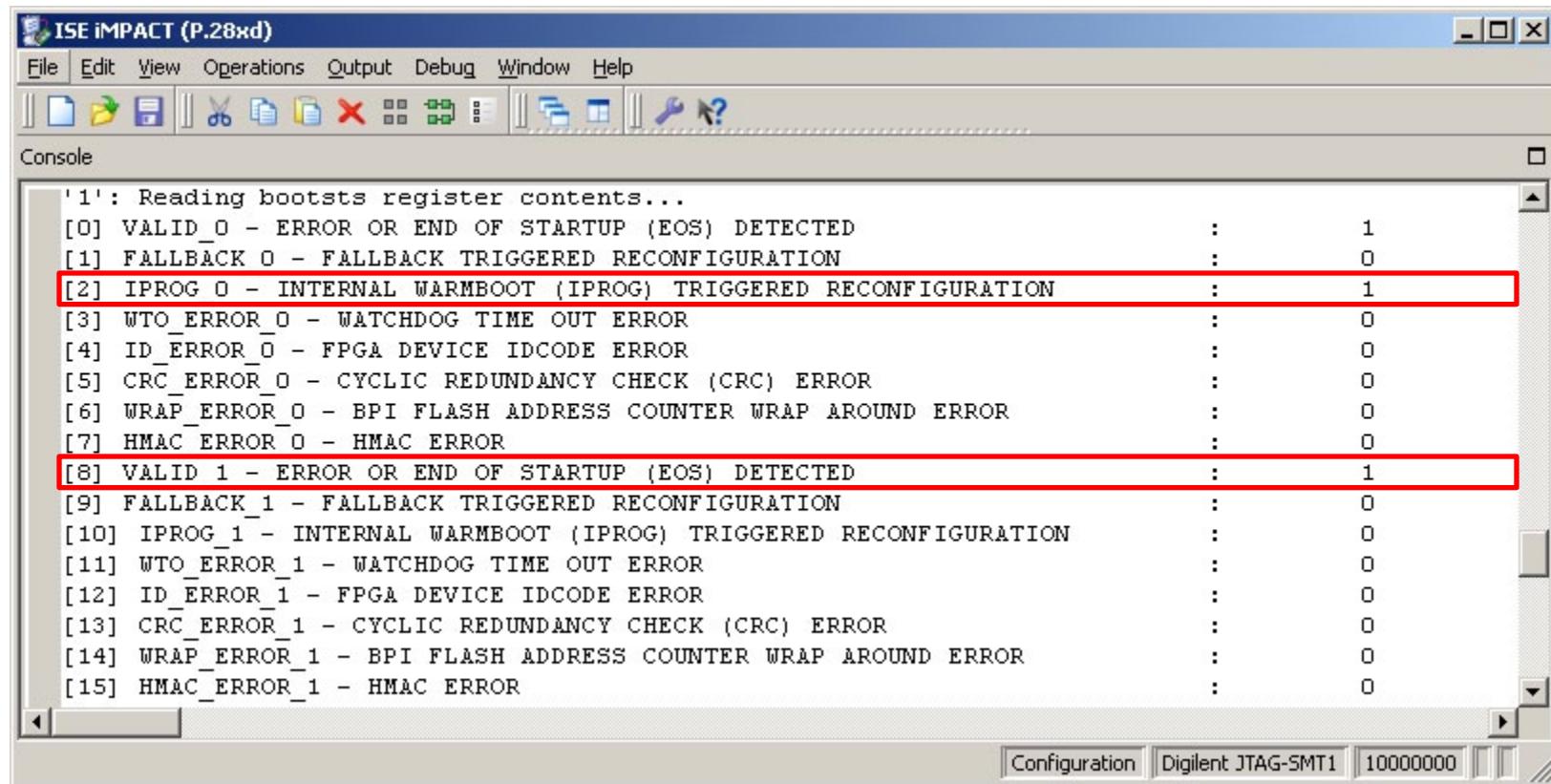
- Select the menu item Debug → Read Device Status
- Scroll down to the “Reading bootsts register contents...” message



Run MultiBoot Design

► The following status register messages are now set:

- IPROG_0
- VALID_1



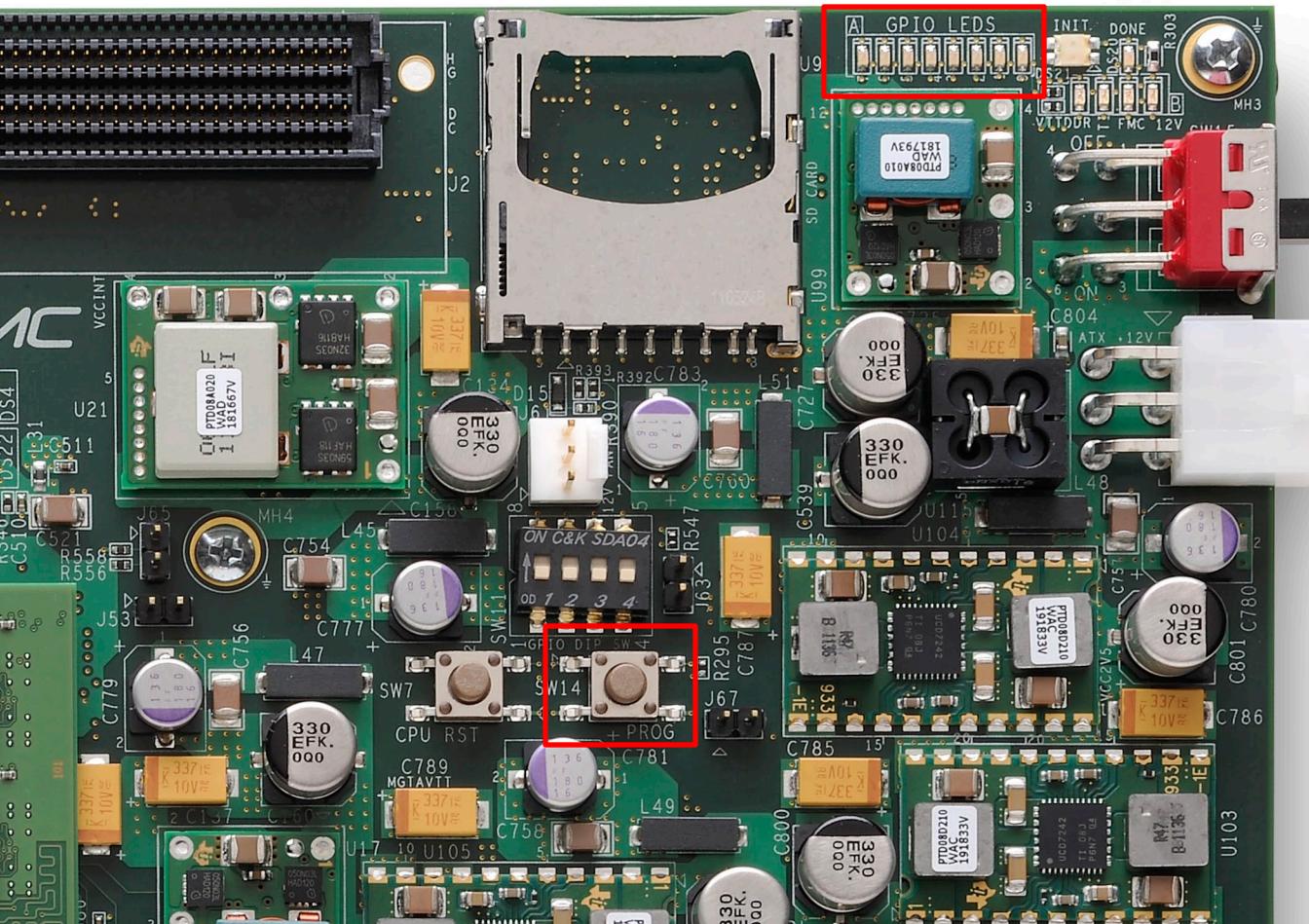
The screenshot shows the ISE iMPACT software interface with the title bar "ISE iMPACT (P.28xd)". The menu bar includes File, Edit, View, Operations, Output, Debug, Window, and Help. The toolbar below has icons for file operations like Open, Save, and Close. The main window is titled "Console". The output window displays the following text:

```
'1': Reading bootssts register contents...
[0] VALID_0 - ERROR OR END OF STARTUP (EOS) DETECTED : 1
[1] FALLBACK_0 - FALLBACK TRIGGERED RECONFIGURATION : 0
[2] IPROG_0 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 1
[3] WTO_ERROR_0 - WATCHDOG TIME OUT ERROR : 0
[4] ID_ERROR_0 - FPGA DEVICE IDCODE ERROR : 0
[5] CRC_ERROR_0 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 0
[6] WRAP_ERROR_0 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[7] HMAC_ERROR_0 - HMAC ERROR : 0
[8] VALID_1 - ERROR OR END OF STARTUP (EOS) DETECTED : 1
[9] FALLBACK_1 - FALLBACK TRIGGERED RECONFIGURATION : 0
[10] IPROG_1 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 0
[11] WTO_ERROR_1 - WATCHDOG TIME OUT ERROR : 0
[12] ID_ERROR_1 - FPGA DEVICE IDCODE ERROR : 0
[13] CRC_ERROR_1 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 0
[14] WRAP_ERROR_1 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[15] HMAC_ERROR_1 - HMAC ERROR : 0
```

The lines for [2], [8], and [9] are highlighted with a red border.

Run MultiBoot Design

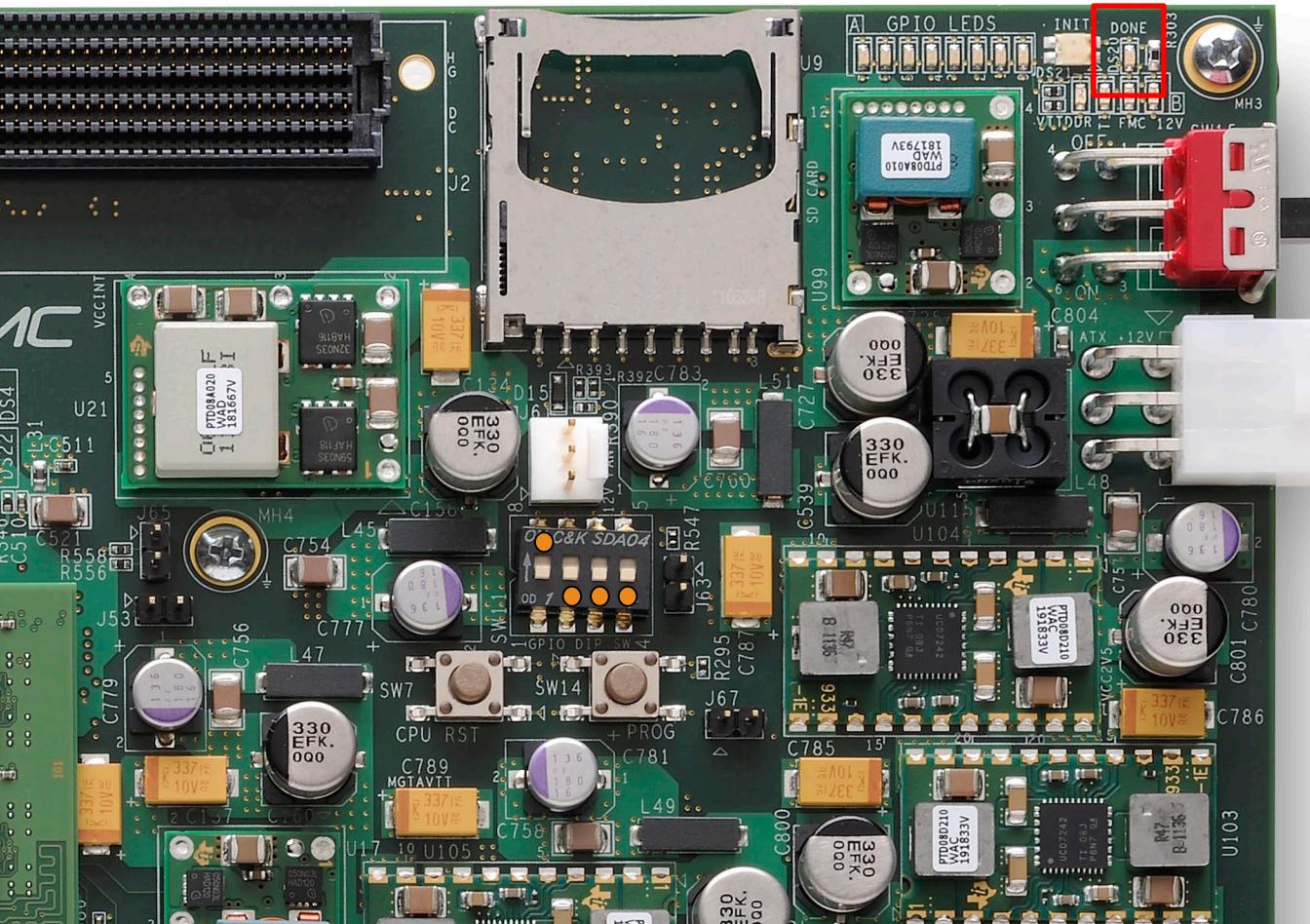
- Press PROG to return to the golden bitstream
 - Wait until you see the cycling pattern on the LEDs



Run MultiBoot Design

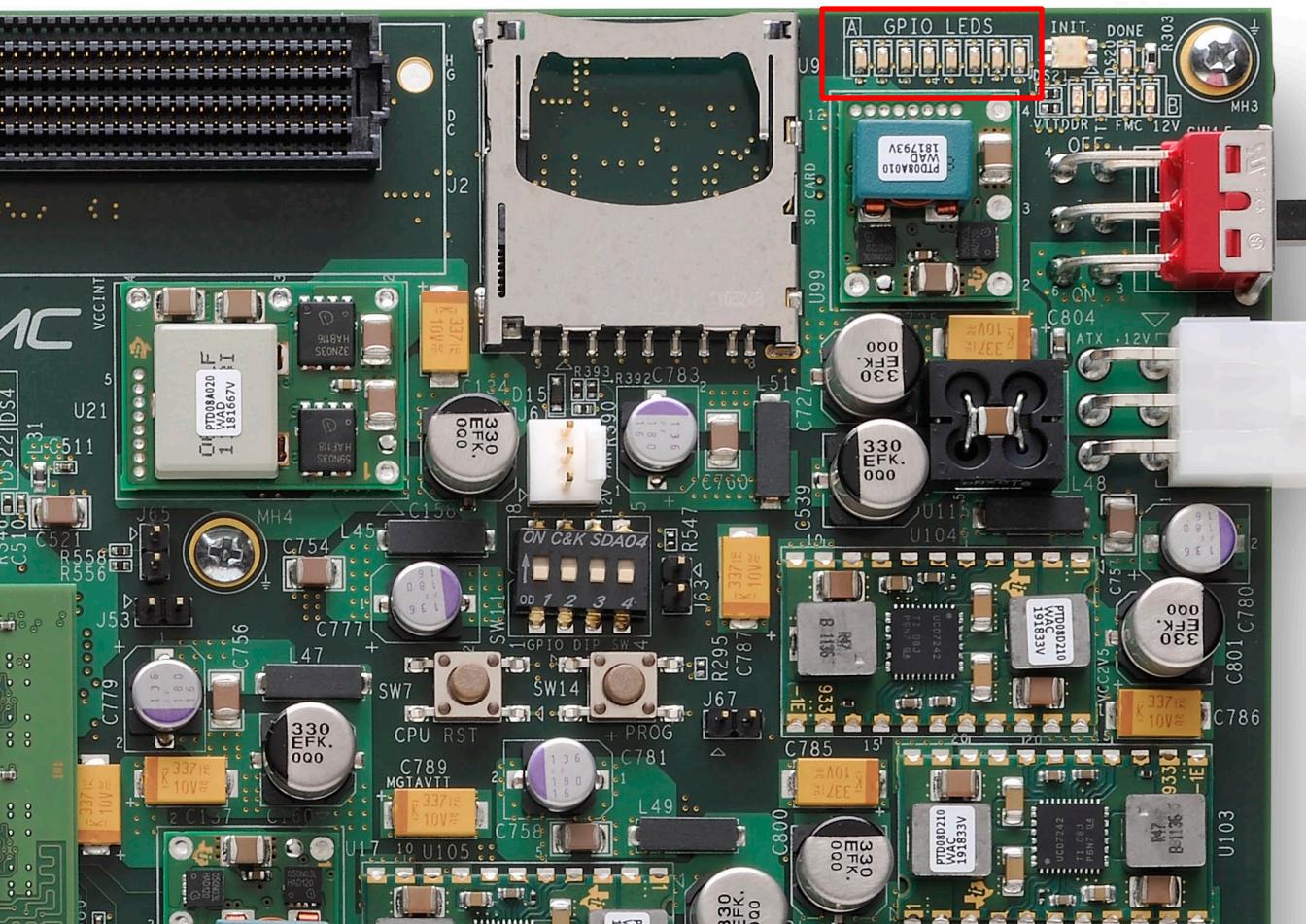
► Set S11 to 1000 (1 = on, Position 1 → Position 4)

- Issues an IPROG command to re-configure from a corrupted MultiBoot bitstream
 - Set it back to 0000 when the DONE LED goes out



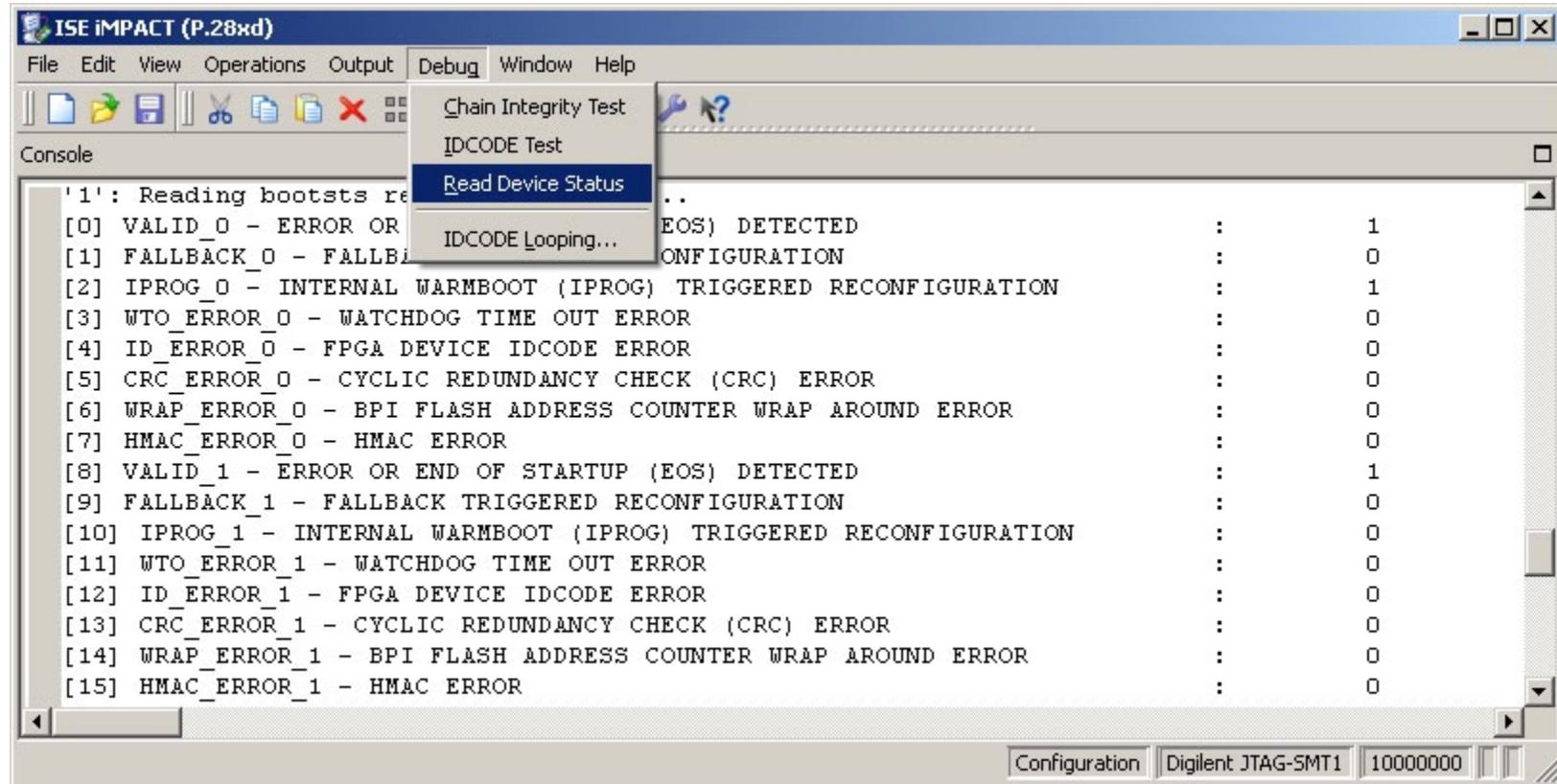
Run MultiBoot Design

- The corrupted.bit generates a CRC error and cannot load the FPGA
- The FPGA falls back to the golden bitstream with a cycling LED pattern



Run MultiBoot Design

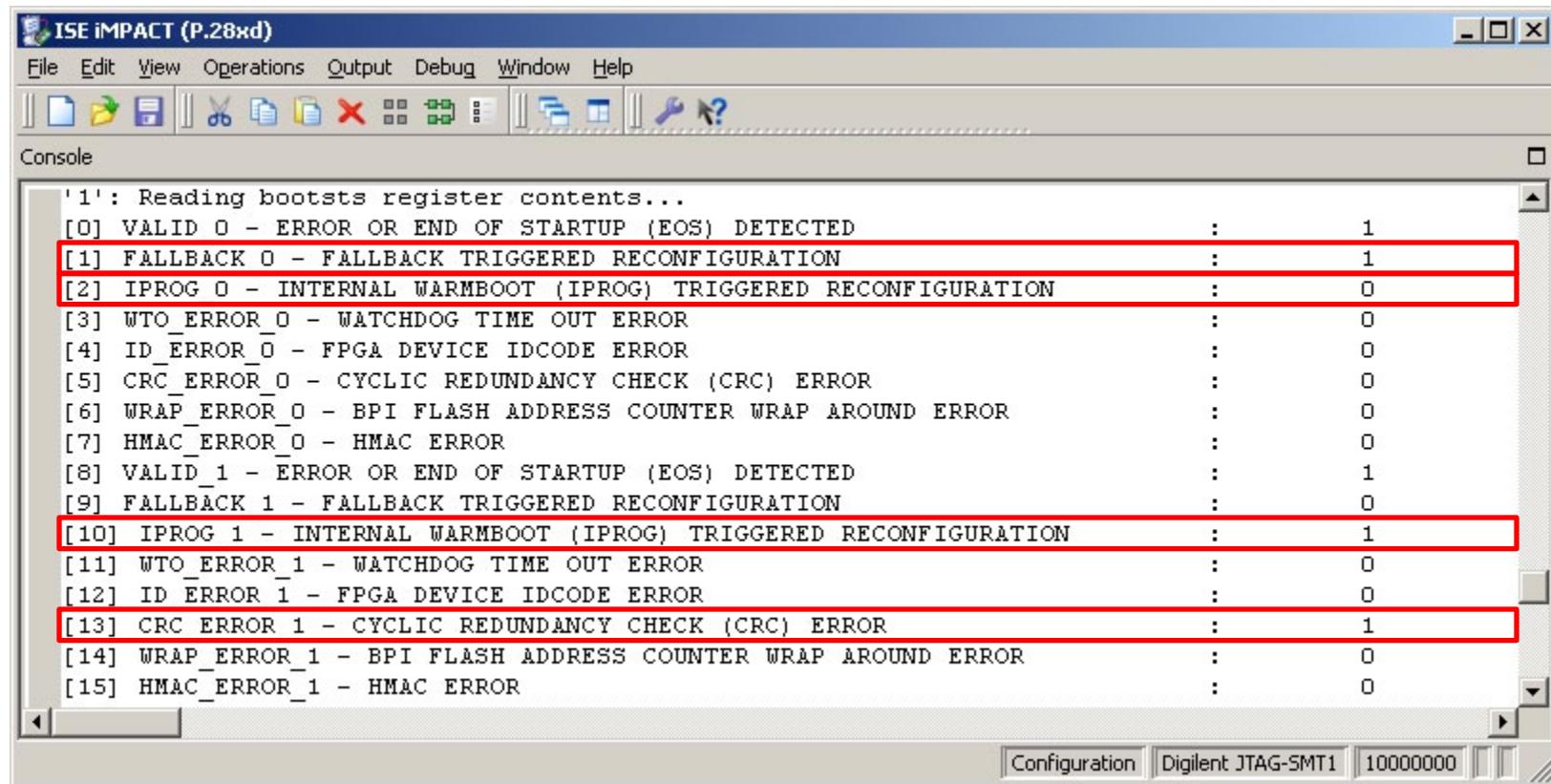
- Select the menu item Debug → Read Device Status
- Scroll down to the “Reading bootsts register contents...” message



Run MultiBoot Design

► The following status register messages are now seen:

- IPROG_0 is no longer active
- FALBACK_0, IPROG_1, and CRC_ERROR_1 are now active



The screenshot shows the ISE iMPACT software interface with a "Console" tab selected. The output window displays the following text:

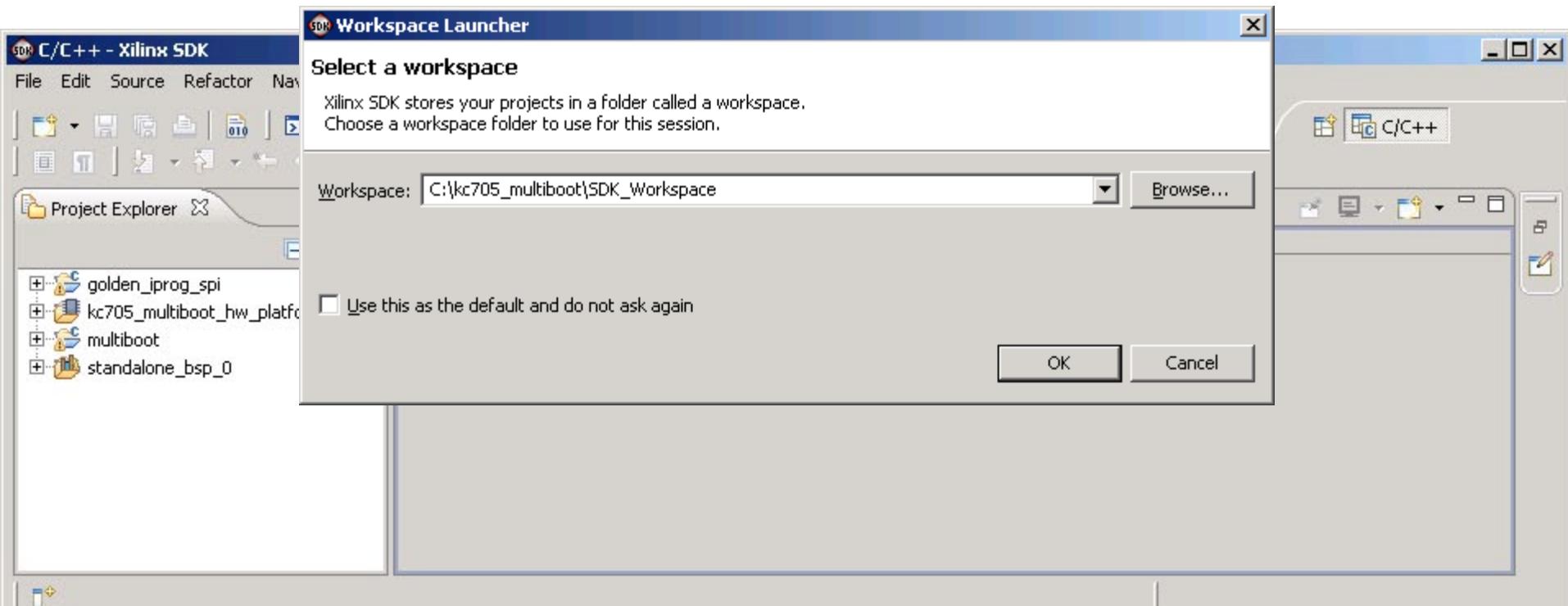
```
'1': Reading bootsr register contents...
[0] VALID_0 - ERROR OR END OF STARTUP (EOS) DETECTED : 1
[1] FALBACK_0 - FALBACK TRIGGERED RECONFIGURATION : 1
[2] IPROG_0 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 0
[3] WTO_ERROR_0 - WATCHDOG TIME OUT ERROR : 0
[4] ID_ERROR_0 - FPGA DEVICE IDCODE ERROR : 0
[5] CRC_ERROR_0 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 0
[6] WRAP_ERROR_0 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[7] HMAC_ERROR_0 - HMAC ERROR : 0
[8] VALID_1 - ERROR OR END OF STARTUP (EOS) DETECTED : 1
[9] FALBACK_1 - FALBACK TRIGGERED RECONFIGURATION : 0
[10] IPROG_1 - INTERNAL WARMBOOT (IPROG) TRIGGERED RECONFIGURATION : 1
[11] WTO_ERROR_1 - WATCHDOG TIME OUT ERROR : 0
[12] ID_ERROR_1 - FPGA DEVICE IDCODE ERROR : 0
[13] CRC_ERROR_1 - CYCLIC REDUNDANCY CHECK (CRC) ERROR : 1
[14] WRAP_ERROR_1 - BPI FLASH ADDRESS COUNTER WRAP AROUND ERROR : 0
[15] HMAC_ERROR_1 - HMAC ERROR : 0
```

The lines corresponding to FALBACK_0, IPROG_1, and CRC_ERROR_1 are highlighted with a red border.

Compile KC705 MultiBoot Design

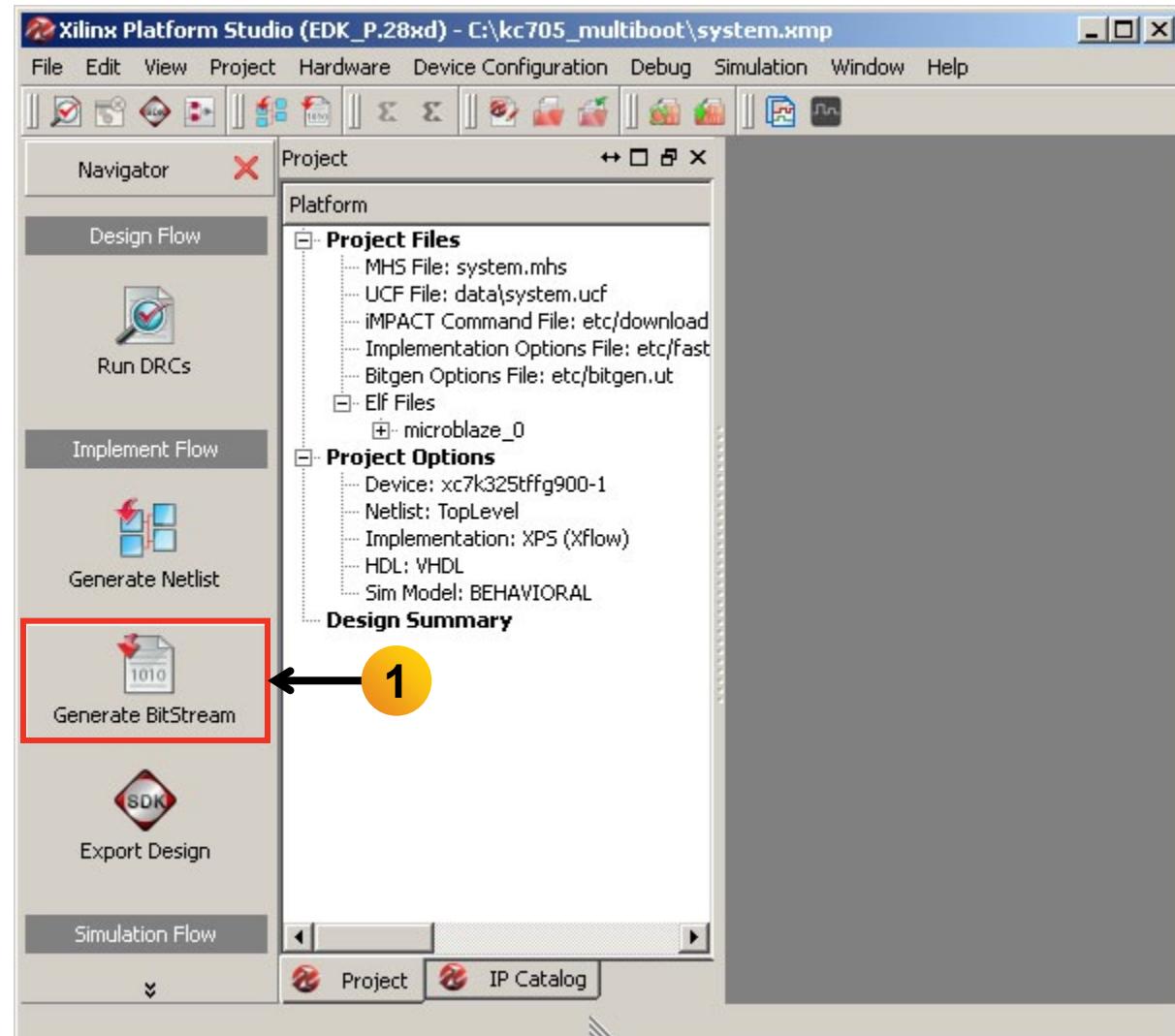
Compile KC705 MultiBoot Design

- If desired, FPGA compile can be skipped by opening SDK directly:
Start → All Programs → Xilinx Design Tools → ISE Design Suite 14.2 → EDK → Xilinx Software Development Kit
- Select the workspace: <design files>\SW\SDK
- Go to SDK Software Compile



Compile KC705 MultiBoot Design

- Open XPS project
<project directory>\
system.xmp
- Create the hardware
design, system.bit,
located in
<project directory>
/implementation
 - Click the Generate
Bitstream button (1)
 - Or from the menu, select
Hardware → Generate
Bitstream



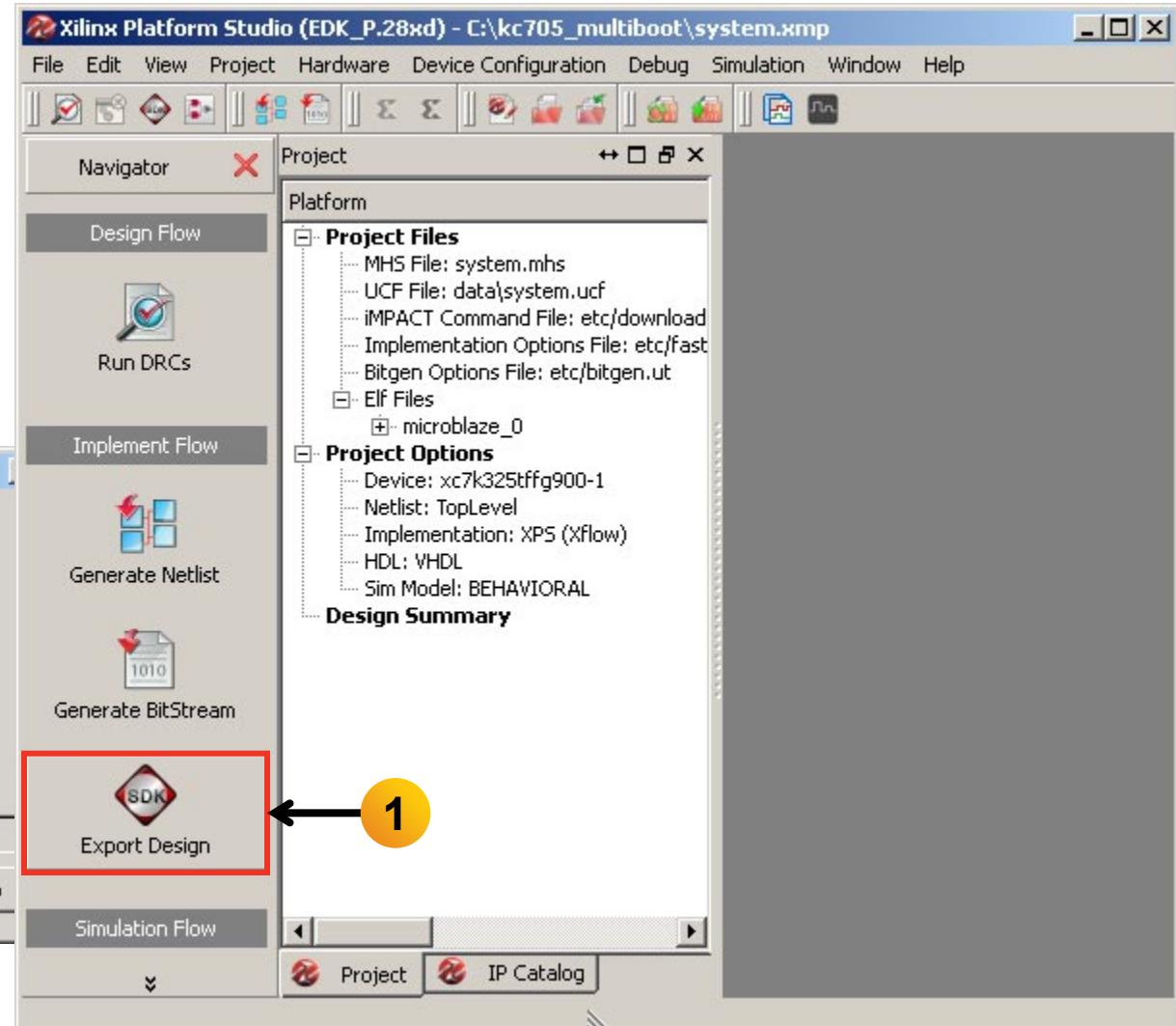
Launch KC705 Design in SDK

► Open SDK

- Click the Export Design button (1)
- Click Export & Launch SDK (2)



2

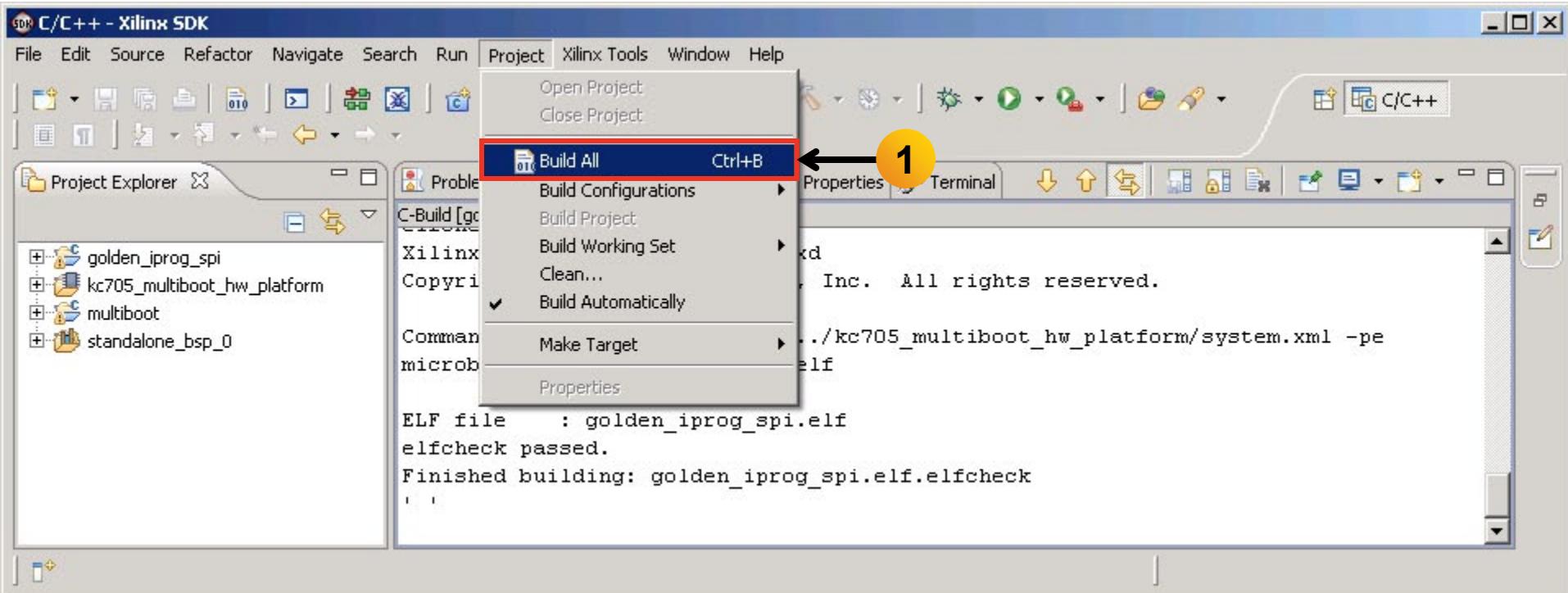


1

Compile KC705 Software in SDK

► SDK Software Compile - Build ELF files in SDK

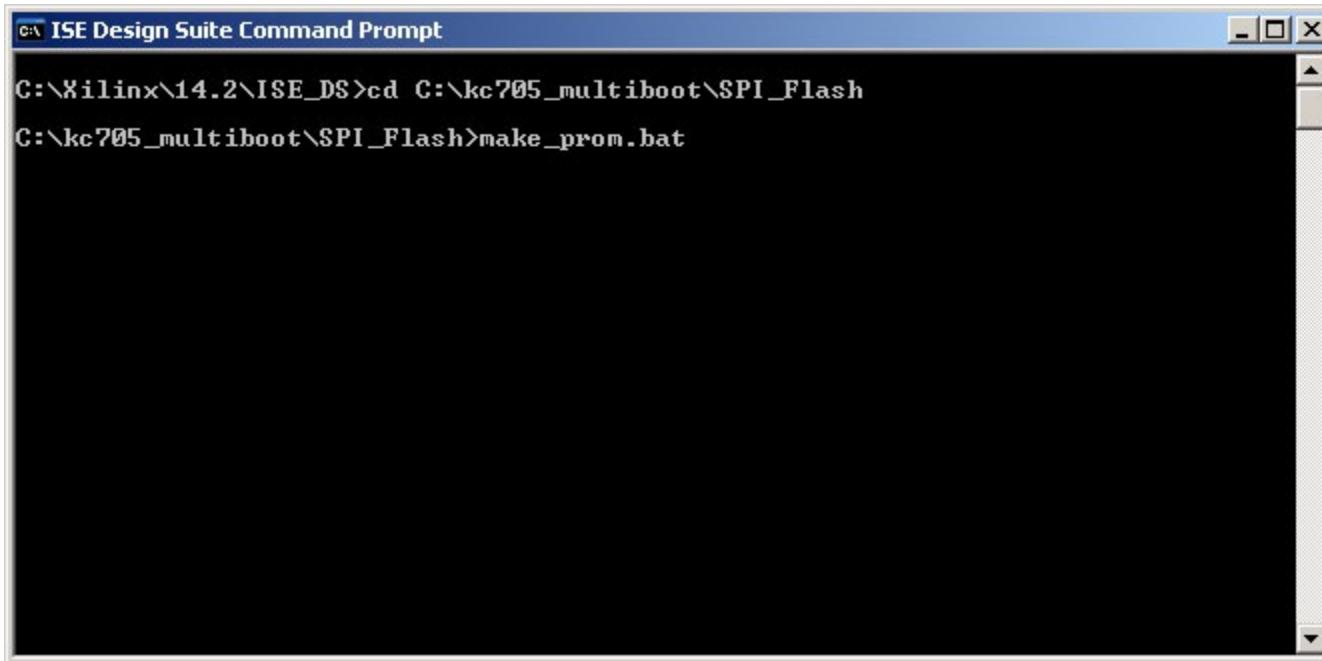
- Project should build automatically; if not, select Project → Build All (1)
- Note: If bypassing the FPGA compile, the ELF files are already built; if desired, the ELF files can be re-built by selecting Clean... followed by Build All



Generate KC705 PROM File

- Start a ISE Design Suite Command Prompt and enter these commands:

```
cd C:\kc705_multiboot\SPI_Flash  
make_prom.bat
```

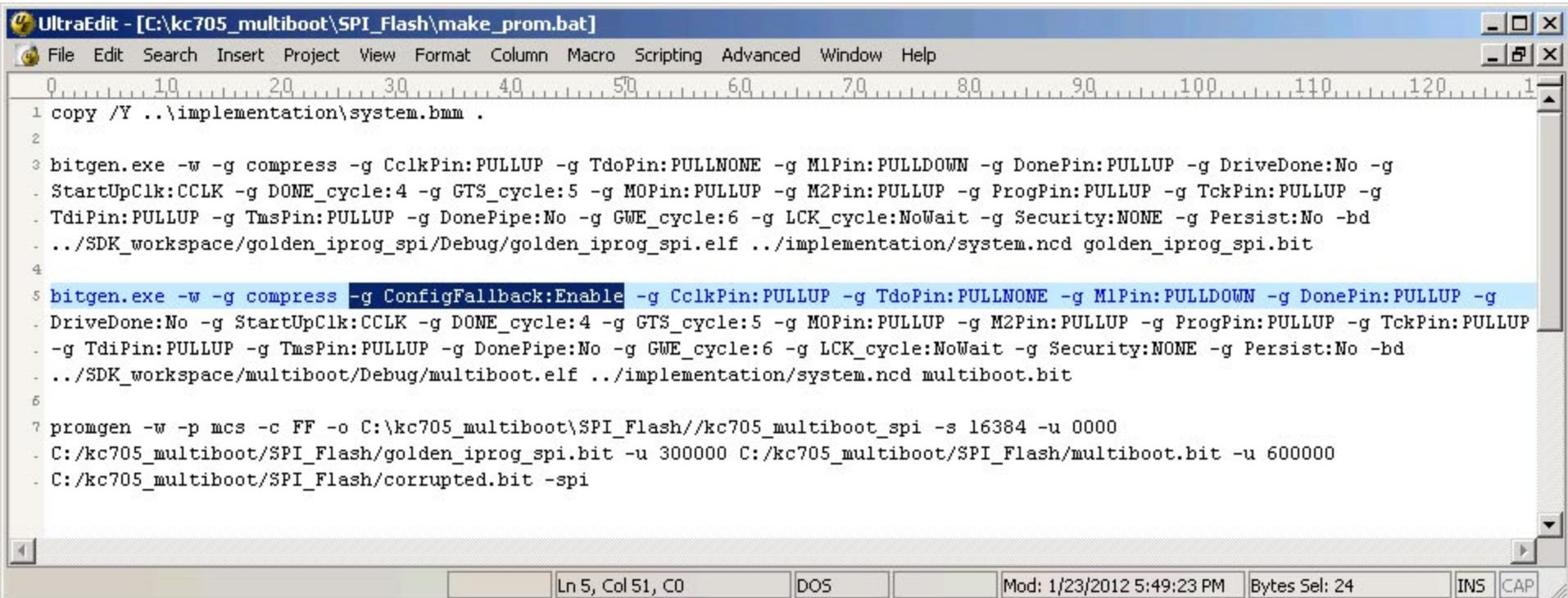


The screenshot shows a Windows command prompt window titled "ISE Design Suite Command Prompt". The window contains the following text:

```
C:\Xilinx\14.2\ISE_DS>cd C:\kc705_multiboot\SPI_Flash  
C:\kc705_multiboot\SPI_Flash>make_prom.bat
```

Generate KC705 PROM File

- The make_prom.bat generates compressed bitstreams with the ELF files in BRAM
- The MultiBoot bitstream uses the -g ConfigFallback:Enable option
- “corrupted.bit” is a manually corrupted bitstream



The screenshot shows a window titled "UltraEdit - [C:\kc705_multiboot\SPI_Flash\make_prom.bat]" containing a batch script. The script performs the following steps:

- 1 copy /Y ..\implementation\system.bmm .
- 2
- 3 bitgen.exe -w -g compress -g CclkPin:PULLUP -g TdoPin:PULLNONE -g M1Pin:PULLDOWN -g DonePin:PULLUP -g DriveDone:No -g StartUpClk:CCLK -g DONE_cycle:4 -g GTS_cycle:5 -g MOPin:PULLUP -g M2Pin:PULLUP -g ProgPin:PULLUP -g TckPin:PULLUP -g TdiPin:PULLUP -g TmsPin:PULLUP -g DonePipe:No -g GWE_cycle:6 -g LCK_cycle:NoWait -g Security:NONE -g Persist:No -bd ..\SDK_workspace/golden_iprog_spi/Debug/golden_iprog_spi.elf ..\implementation/system.ncd golden_iprog_spi.bit
- 4
- 5 bitgen.exe -w -g compress -g ConfigFallback:Enable -g CclkPin:PULLUP -g TdoPin:PULLNONE -g M1Pin:PULLDOWN -g DonePin:PULLUP -g DriveDone:No -g StartUpClk:CCLK -g DONE_cycle:4 -g GTS_cycle:5 -g MOPin:PULLUP -g M2Pin:PULLUP -g ProgPin:PULLUP -g TckPin:PULLUP -g TdiPin:PULLUP -g TmsPin:PULLUP -g DonePipe:No -g GWE_cycle:6 -g LCK_cycle:NoWait -g Security:NONE -g Persist:No -bd ..\SDK_workspace/multiboot/Debug/multiboot.elf ..\implementation/system.ncd multiboot.bit
- 6
- 7 promgen -w -p mcs -c FF -o C:\kc705_multiboot\SPI_Flash\kc705_multiboot_spi -s 16384 -u 0000 C:/kc705_multiboot/SPI_Flash/golden_iprog_spi.bit -u 300000 C:/kc705_multiboot/SPI_Flash/multiboot.bit -u 600000 C:/kc705_multiboot/SPI_Flash/corrupted.bit -spi

Kintex-7 MultiBoot Details

MultiBoot Register Setup & Command

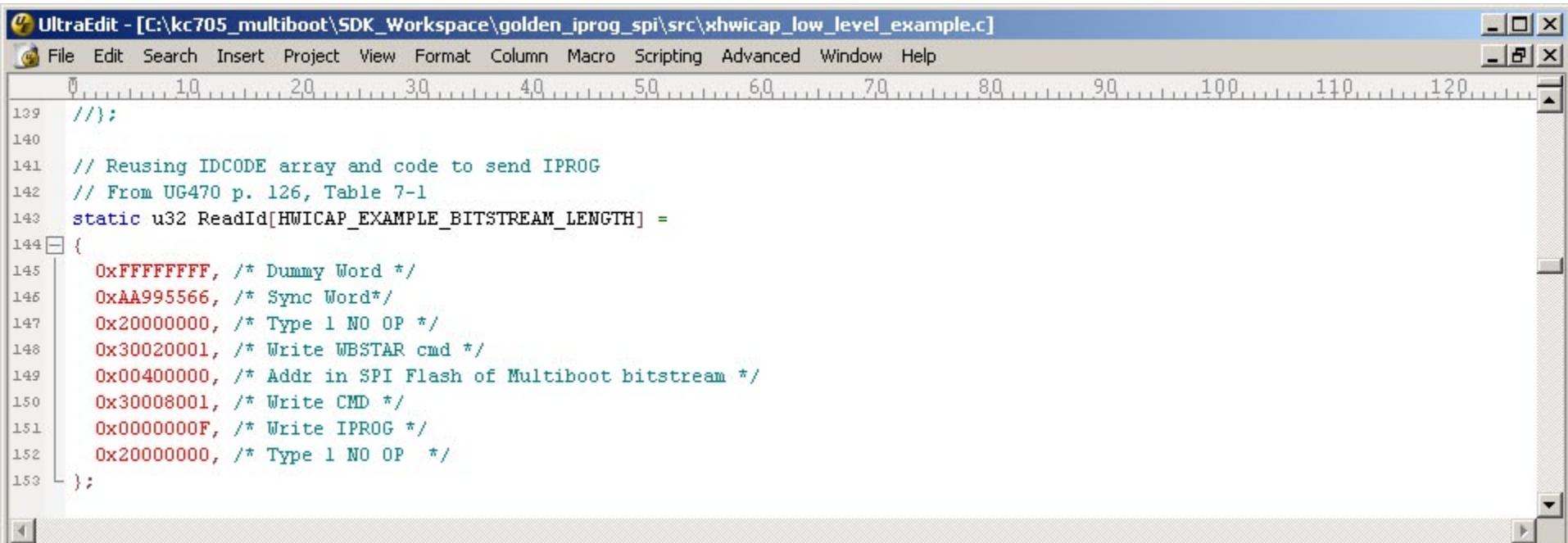
- Below are the values programmed into the ICAP via IPROG
 - This table is from UG470, 7 Series FPGAs Configuration User Guide

Table 7-1: Example Bitstream for IPROG through ICAP

Configuration Data (hex)	Explanation
FFFFFFFF	Dummy Word
AA995566	Sync Word
20000000	Type 1 NO OP
30020001	Type 1 Write 1 Words to WBSTAR
00000000	Warm Boot Start Address (Load the Desired Address)
30008001	Type 1 Write 1 Words to CMD
0000000F	IPROG Command
20000000	Type 1 NO OP

MultiBoot Register Setup & Command

- IPROG command issued via the software in xhwicap_low_level_example.c



The screenshot shows a window titled "UltraEdit - [C:\kc705_multiboot\SDK_Workspace\golden_iprog_spi\src\xhwicap_low_level_example.c]" containing C code. The code defines a static array 'ReadId' of type u32 with a length of HWICAP_EXAMPLE_BITSTREAM_LENGTH. The array contains 144 elements, starting with a dummy word (0xFFFFFFFF) and followed by various SPI commands and addresses. The code is annotated with comments explaining each byte's purpose.

```
139 //};  
140  
141 // Reusing IDCODE array and code to send IPROG  
142 // From UG470 p. 126, Table 7-1  
143 static u32 ReadId[HWICAP_EXAMPLE_BITSTREAM_LENGTH] =  
144 {  
145     0xFFFFFFFF, /* Dummy Word */  
146     0xAA995566, /* Sync Word*/  
147     0x20000000, /* Type 1 NO OP */  
148     0x30020001, /* Write WBSTAR cmd */  
149     0x00400000, /* Addr in SPI Flash of Multiboot bitstream */  
150     0x30008001, /* Write CMD */  
151     0x0000000F, /* Write IPROG */  
152     0x20000000, /* Type 1 NO OP */  
153 };
```

References

References

► 7 Series Configuration

- 7 Series FPGAs Configuration User Guide
 - http://www.xilinx.com/support/documentation/user_guides/ug470_7Series_Config.pdf

Documentation

Documentation

➤ Kintex-7

- Kintex-7 FPGA Family
 - <http://www.xilinx.com/products/silicon-devices/fpga/kintex-7/index.htm>

➤ KC705 Documentation

- Kintex-7 FPGA KC705 Evaluation Kit
 - <http://www.xilinx.com/products/boards-and-kits/EK-K7-KC705-G.htm>
- KC705 Getting Started Guide
 - http://www.xilinx.com/support/documentation/boards_and_kits/ug883_K7_KC705_Eval_Kit.pdf
- KC705 User Guide
 - http://www.xilinx.com/support/documentation/boards_and_kits/ug810_KC705_Eval_Bd.pdf
- KC705 Reference Design User Guide
 - http://www.xilinx.com/support/documentation/boards_and_kits/ug845_Ref_Design.pdf