Eliminate Production Programming Frustrations

Use the Right Tool for the right job: Vivado for Development, ScanWorks for Production

BY Larry Osborn


ASSET

Larry Osborn – Senior Product Manager

Larry has more than 30 years of years of experience in embedded software product development and delivery. Roles included product management, marketing, sales support, field applications engineering, and software developer.

Product Manager for ASSET for 16 years. Focus on software centric solutions for embedded test, using both processor and FPGAs.

Prior to ASSET worked for Lockheed Martin, Hewlett-Packard, and Wind River.

Larry earned a Bachelor of Science degree in computer science from University of Kansas, and has been awarded patents with Intel and Hewlett Packard and numerous achievement awards from Wind River and Lockheed Martin.

Table of Contents

Table of Figures

ASSET is a member of the Xilinx partner program, and one value ASSET provides is production test and programming solutions complementing Xilinx tools and supporting Xilinx silicon. As part of the partnership, we jointly support customers, so they are successful. As part of that activity, I have noticed many customers posting on the Xilinx support community struggles they are having trying to use Vivado/SDK/Platform Cable for production in-system programming. Xilinx clearly states those products are not intended for production use.

Due to seeing this issue again and again on the Xilinx community, I was motivated to understand the challenges engineers were having and show a better way to solve the issue. This eBook discusses the challenges and shows the better way to do in-system programming in production.

This eBook is to address what ASSET is seeing with our customer base that are trying to use Xilinx tools for device programming in a **production** setting. In 2021 I did a webinar In-System Fast Flash Programming Technologies on the need/impact of ***fast flash programming*** in system to address beat rate issues and other process management issues. In this webinar I programmed an QSPI device that was connected to the Zynq-7020PS and another QSPI device that was connected to the Zynq-7020PL, another QSPI device connected to a PMOD header using a direct connect method and finally a SD card. After speaking with a few customers, I got curious how the silicon vendors were programming devices in production. So, I embarked on a project to use the Zedboard from Avent and the tool chain from Xilinx (Vivado and SDK) with the Platform Cable to experience for myself what customers experience.

QSPI-PS Programming

Starting with the document "Workshop for Beginners" from Avent I was able to build a Vivado design using the Zedboard as the target. Then using the Xilinx SDK, I was able to build an application to program into the QSPI device on the Zedboard located at IC15.

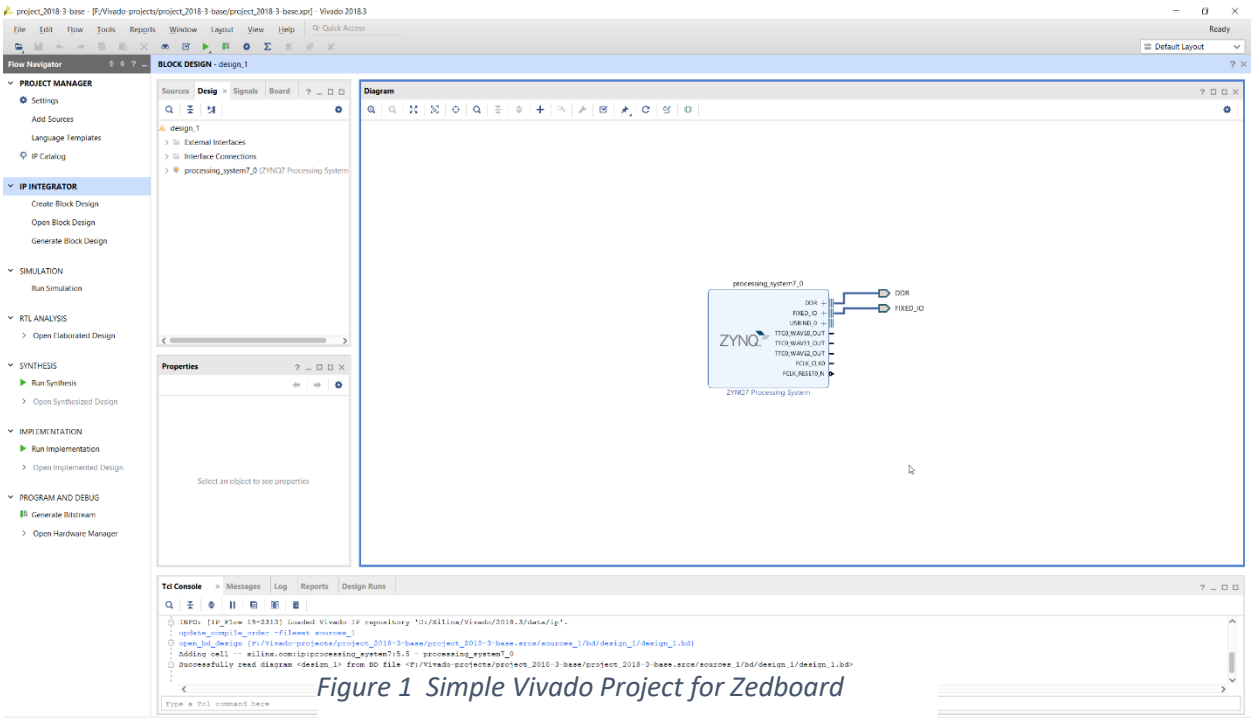

Figure 1 Simple Vivado Project for Zedboard

Trying to use the Platform Cable I was surprised that the programming cable could only load bit files into the FPGA or load code into DDR. I then realized that the Xilinx method was to build a First Stage Boot Loader (FSBL) and adding the application to the FSBL. The FSBL did some DDR and QSPI initialization and then use the code within the ps7_int code to program the QSPI from DDR. From a production perspective it is very dangerous to assume too much about the target status without first testing the fit and interconnects between the devices on the Unit Under Test (UUT). This Xilinx method for production programming would mean a change out of tools from test to programming as the Platform Cable is using the same JTAG interface likely being used for test. In production this change of tools would have a significant hit on the beat rate which would be unacceptable. With ScanWorks the switch from test to programming is simply a different action.

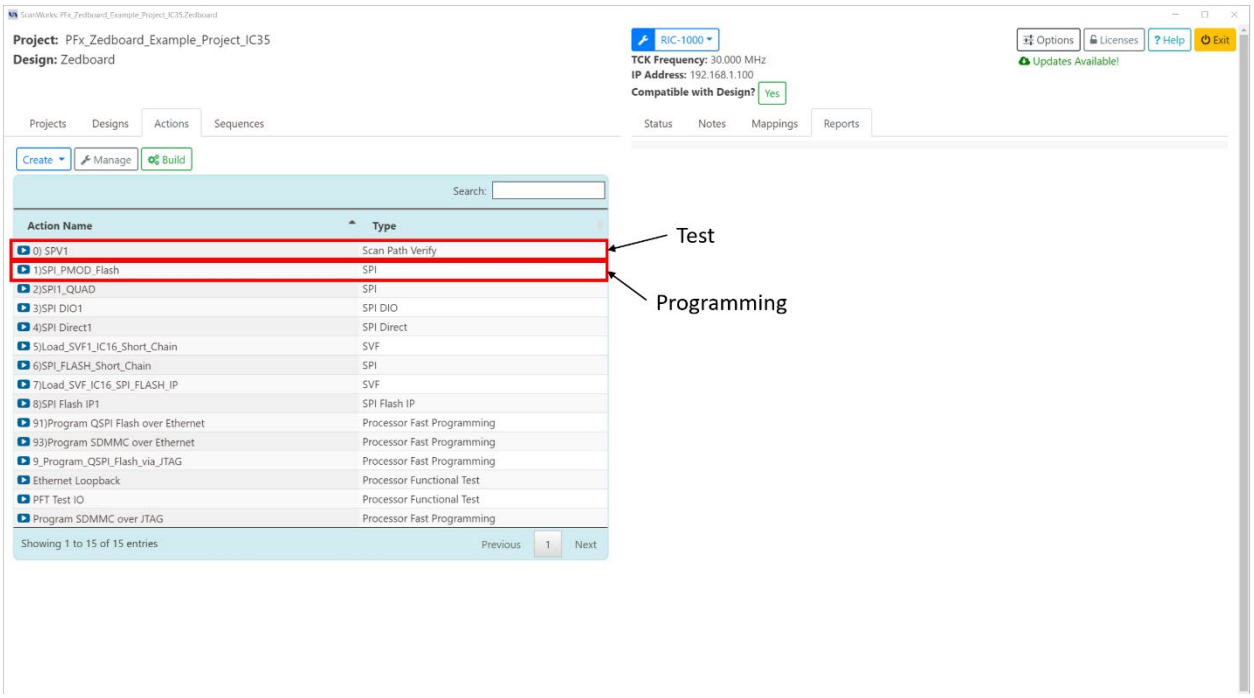

Figure 2 ScanWorks Project for Test and Programming

All the JTAG controllers used in the ScanWorks platform can perform test as well as programming. A single connection and single tool for more efficient for production usage.

QSPI-PL Programming

The next step was to add IP with Vivado in the design for the AXI Quad SPI interface and do some configuration to program the QSPI device connected off the PL. Adding the AXI Quad SPI interface to the design was straight forward and then adding the constraints to handle the connections between the pins of the FPGA bank and the PMOD header was quite easy.

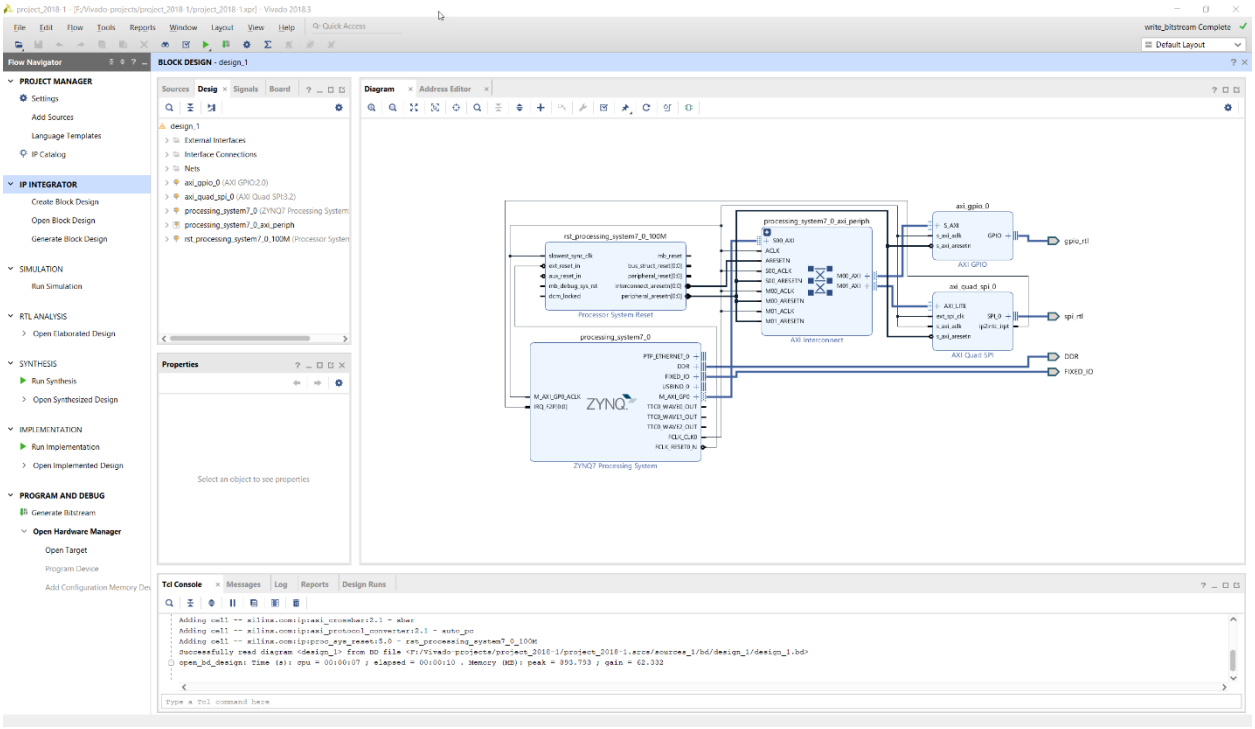

Figure 3 Vivado Project with AXI QUAS SPI IP Block

The data for the constraints is driven by the Zedboard schematics. The Digilent PMOD SF3 32MB NOR Flash uses a N25Q256A SPI device from Micron. This device supports both dual and quad protocols. With the design built and verified, at this point I couldn't see how to program this device unless the Platform Cable had that ability built within. After a call to support, I was informed that a need to write the programming code but, there was an example program that would work. With help from support, they showed me where the code was located and how to include it in the SDK project. Then after building the project and executing it, the programming example failed. Then I was pointed to a reference manual to develop the code.

I used the SDK debugger to only find that the initialization had failed but, there were no error message in the code or clue as to what was causing the failure. During the debug execution, stepping the code behaved badly as the stepping was not in a predictable sequence based upon the source code. This leads to questions about the build and digging further into this issue was not an acceptable use of time.

For ScanWorks the programming of the N25Q256A was a matter of using FPGA Fast Programming action to create the SPI programming bitstream image and program the FPGA. Then run a second action pointing the flash image source and preforming the programming . In the video there are two methods

used to program this flash device connected on the PMOD interface. The first is using short chain (detailed in the video at 26:20) where the JTAG chain of the FPGA is shorted from 100+ cells to just 8 cells. As we are using JTAG to program the device shortening the scan chain increases the programming throughput. The second method is similar to Vivado except we provide the SPI programming IP that does the actual programming of the flash device. This is covered in detail at 29:30 of the video.

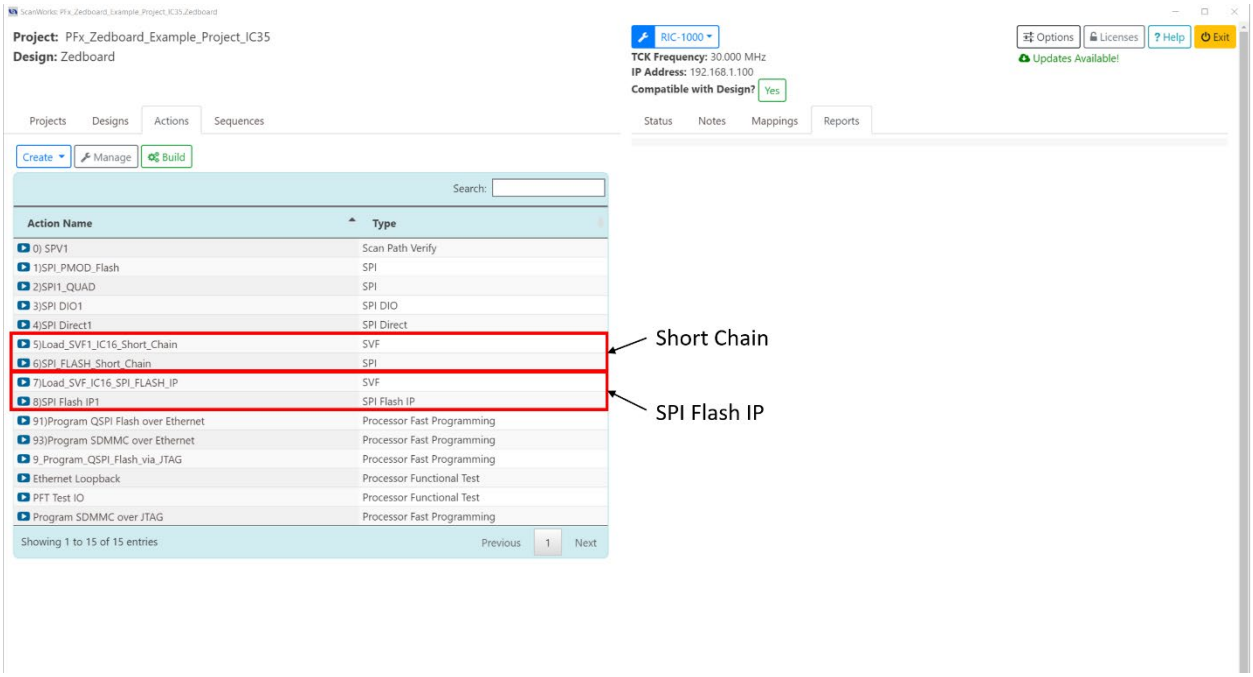

Figure 4 ScanWorks with Short Chain and SPI Flash IP

After hitting a dead end with programming, using Vivado/SDK with the flash device that was connected to the PL. I examine the possibility of programming the same device via a direct method of just using the Platform Cable. I found no way to accomplish this. This is however possible with the ScanWorks. Below is a picture of the target connection using Direct connect. This is also covered within the video.

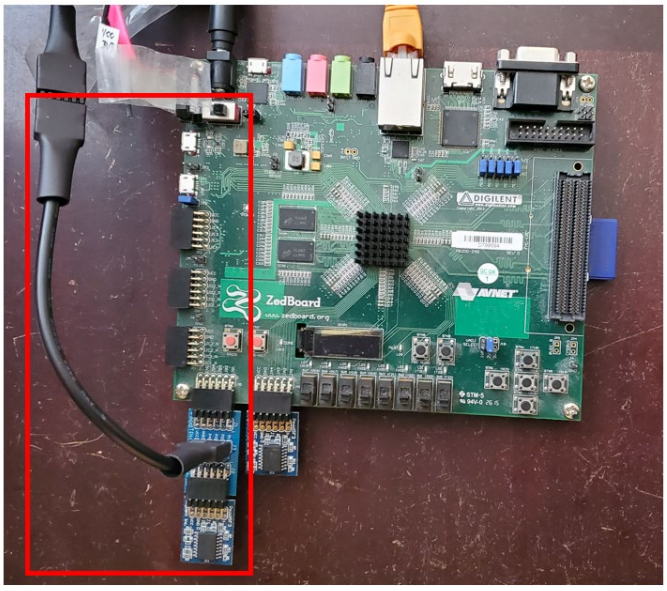

Figure 5 Zedboard with ScanWorks Direct Connect SPI Flash

SD Programming

Moving to SD programming in the exercise provided in the Avnet Workshop for Beginners the method described for using the SD card for autonomous booting was to create a FSBL with a boot image then copy the files generated to the SD card on the PC. That's not programming the SD card from a production perspective. So, I started my search to see if the PS was configured for the use of the SD card. Below is the PS7 IP block diagram which shows SD0 is enabled.

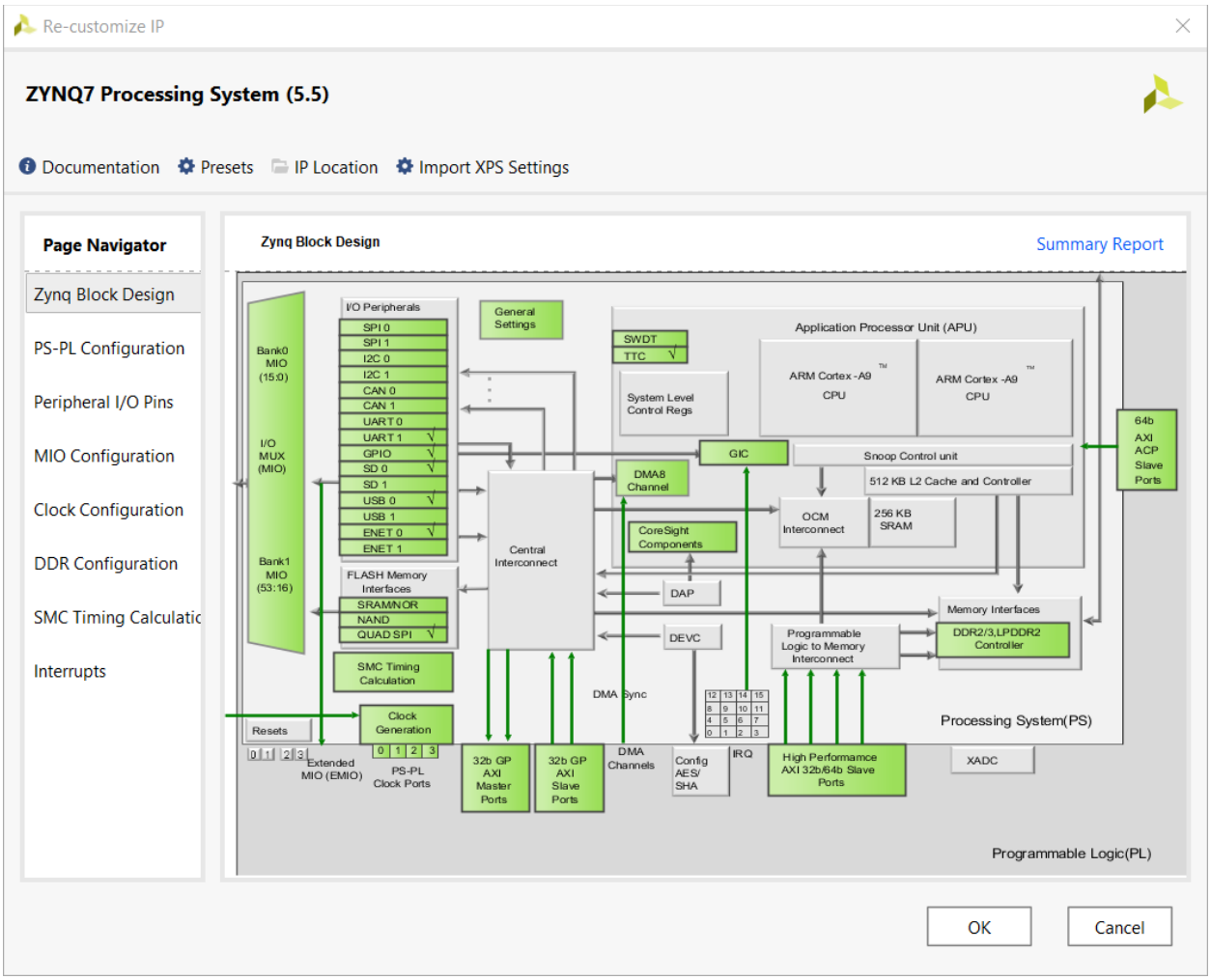

Figure 6 ZYNQ7 with SD0 enabled

And the MIO mapping had been configured.

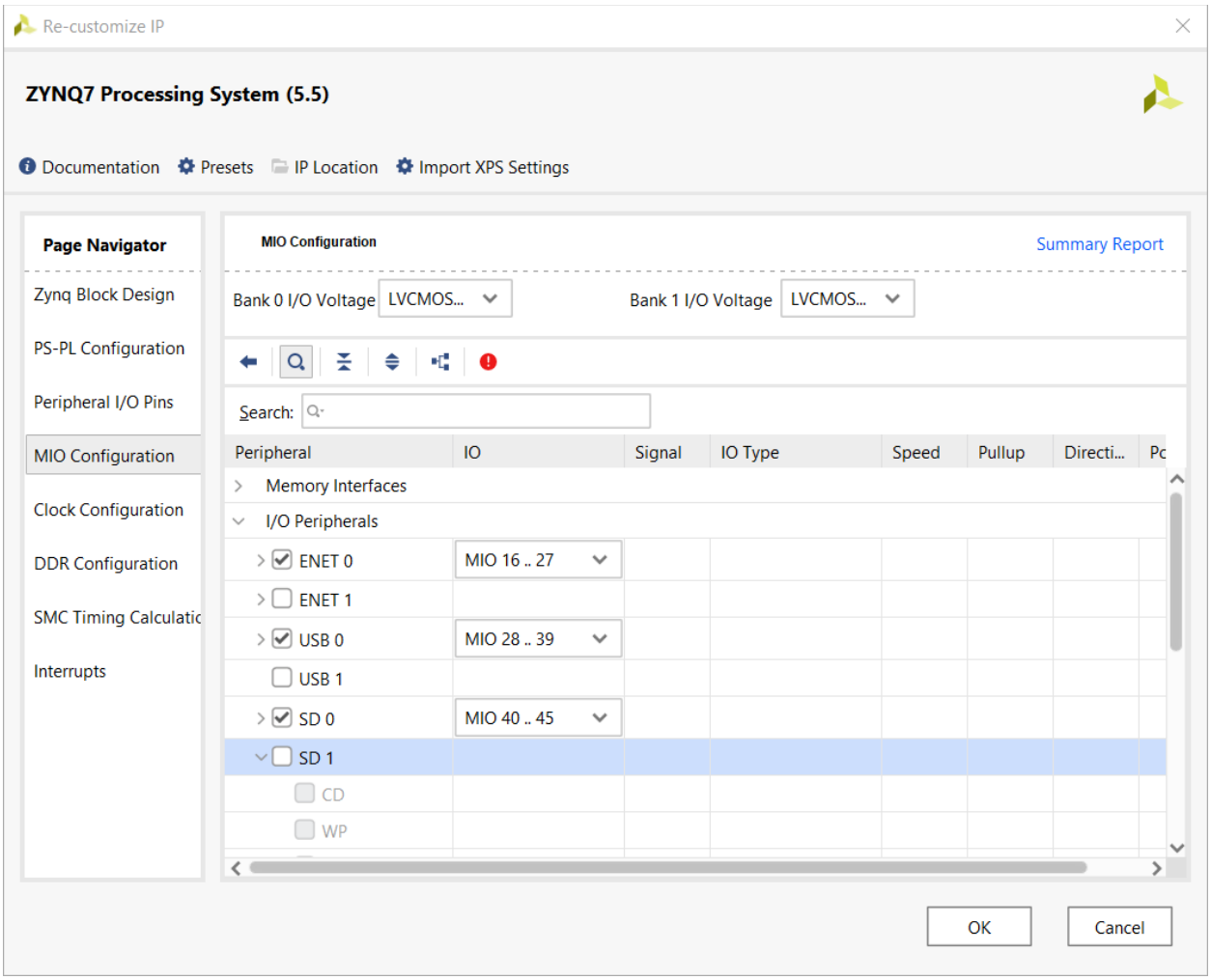

Figure 7 SD 0 MIO Configuration

Then I searched for some example code or guidance on how to program the SD card. The search resulted in finding nothing. My assumption is that the Xilinx approach was documented in the Avnet example which is not acceptable for production.

Within the ScanWorks platform there is built-in capability to program the SD Card or eMMC devices. In this case the programming code is not via the FPGA but rather through the processor. The code is loaded in the OCM, and we also enable the use of the Ethernet on the Zedboard to provide a high-speed data pipe for the flash image. Thus, we run a processor speed and without DDR for an extremely high rate of programming throughput. This method allows for the programming eMMC devices in a production setting.

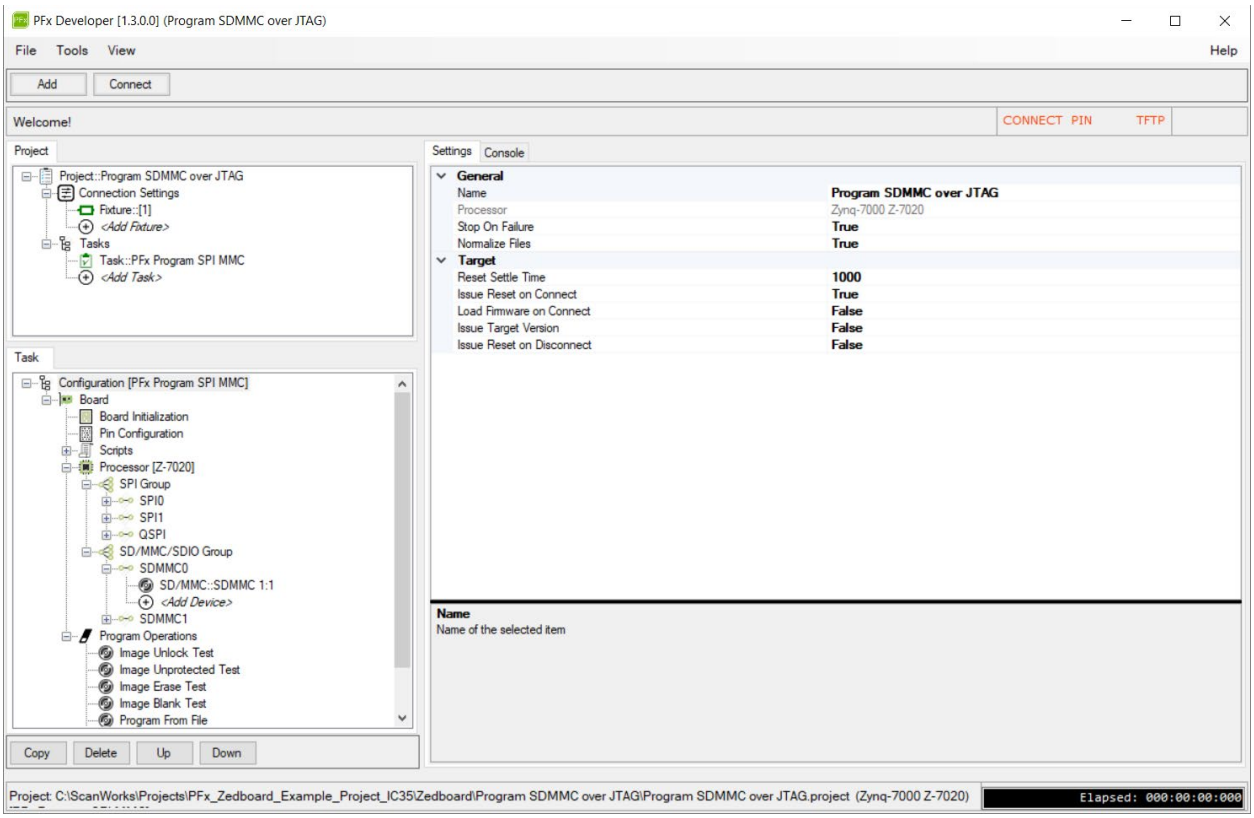

Figure 8 ScanWorks PFx with SD/eMMC Programming

Summary

After weeks of effort to get a grasp on what issues customers might be seeing while using the Platform Cable and the tool chain (Vivado and SDK), I can see why customers are struggling to use this tool in production. However, in the documentation for the Platform Cable, it clearly states that the tool should only be use in a **Prototype environment**. To Xilinx credit there is tons of documentation. It is a challenge to find what documentation is appropriate for the task of simply programming a UUT. Again, the Xilinx tools are designed for FPGA developers not for production test engineers. Test engineers need tools that focus on test engineering roles and responsibilities and ScanWorks is designed for that engineering sector.

I found using the combination of Vivado and SDK user interface too complex to address the needs of production. I realize that development engineers would provide the FSBL with the attached application that could do the programming but, as stated earlier it is too presumptuous to assume the health of the target to be fully operational. I also found that it is possible to have a design rebuilt and without moving the design into the SDK workspace you can build a boot loader with the wrong design and not be aware. There are more detail findings in the white paper “A Users Perspective of Using Xilinx Vivado in Production for Programming”

Overall, the Xilinx tools are not intended for production usage but, they do a great job in address in the needs of the FPGA designer that is using Xilinx silicon.