

A large satellite dish antenna is positioned in a field of tall, dry grass. The dish is white and mounted on a metal structure. The background shows a sunset sky with orange and yellow clouds, and distant mountains. The foreground is a field of tall, dry grass.

Restoring MiniZed to the Factory State

© 2017 Avnet. All rights reserved. All trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Avnet is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Avnet makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Avnet expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Overview

This document contains instructions for restoring the programmable devices on MiniZed to the factory default state. This may be needed if the contents of the on-board memories is corrupted or wrong or if a test of the hardware is required. Corruption of the eMMC memory could occur when programming of either device is interrupted or if Linux is shut down without using the **shutdown -h now** command.

If you do not want to restore MiniZed to the factory state, another option is to put an image in flash that you can revert to if the flash gets corrupted. That mechanism is also explained in this document.

There are three re-programmable devices on the Avnet MiniZed board.

- **U7**: Microchip 2kbit serial EEPROM, with Part Number 93LC56BT-I/OT
- **U2**: Micron 128Mbit QSPI NOR flash, with Part Number MT25QL128ABA8E12-1SIT
- **U3**: Micron 8GB eMMC, with Part Number MTFC8GAKAJCN-4M IT

The EEPROM is programmed in the factory and should not be re-programmed by the user. The content of the EEPROM is proprietary and Avnet does not provide the programming file or support for modifications to its content.

This document should be part of a zip archive that contains the files required for re-programming. MiniZed re-programming is done in two stages:

- a) In the first stage the XSCT (Xilinx Software Command-Line Tool) is used to re-program the QSPI flash (U2) so that MiniZed can boot entirely from flash.
- b) The flash size is limited, so in order to run larger applications (such as Bluetooth), files such as image.ub must be placed into the eMMC memory (U3) instead of in the QSPI flash. The QSPI image is replaced with one that will cause boot to complete by loading image.ub from eMMC. Files for this stage are placed on a USB memory stick that is plugged into MiniZed. Scripts are then executed to format the eMMC, copy over the required files from the USB memory stick to eMMC and to internally re-program the QSPI flash.

Once re-programmed and re-booted, factory-style acceptance tests can be performed to make sure the hardware is OK.

Required Hardware

- 1) **PC/laptop** with one or more USB 2.0-compatible USB ports. This document assumes Windows 7, but another OS or version could also work.
- 2) **MiniZed** PCB assembly ([AES-MINIZED-7Z007-G](#))
- 3) 2 x USB Type A to **Micro USB** cables
- 4) 1GB or larger USB Type 2.0-compatible **memory stick** that is formatted as FAT or FAT32.
- 5) **Wi-Fi access point** with known WPA-PSK user name and password. This can be a smartphone hotspot also.

Required Software

- 1) Xilinx SDK 2017.1 or later from <https://www.xilinx.com/support/download.html> . The XSCT tool of this installation can be used to program the QSPI flash, U2.
- 2) A serial text terminal, such as TeraTerm from <https://osdn.net/projects/ttssh2/releases> , must be used.

Installing the Xilinx Tools

The tool that is used for programming the QSPI flash is XSCT, the Xilinx SDK Command-line Tool. It is installed as part of the Xilinx WebPack that can be downloaded from <https://www.xilinx.com/support/download.html>. A license is not required for the WebPack version. **Version 2017.1 or later must be used.** The steps below are shown for version 2017.1 of the tools, but it is recommended to install the latest version.

- 1) The WebPack can be installed by clicking the link for that. This will download an installer which will manage the download and install. Another option is to download the All OS single file installer. This is a 20GB file and the download may take long but the install process could be simpler. Here it will be assumed that the full 20GB zip file for 2017.1, *Xilinx_Vivado_SDK_2017.1_0415_1.tar*, was downloaded and unzipped.
- 2) Double-click on xsetup.exe, click 'Next' and agree to accept all license agreements before clicking 'Next' again.

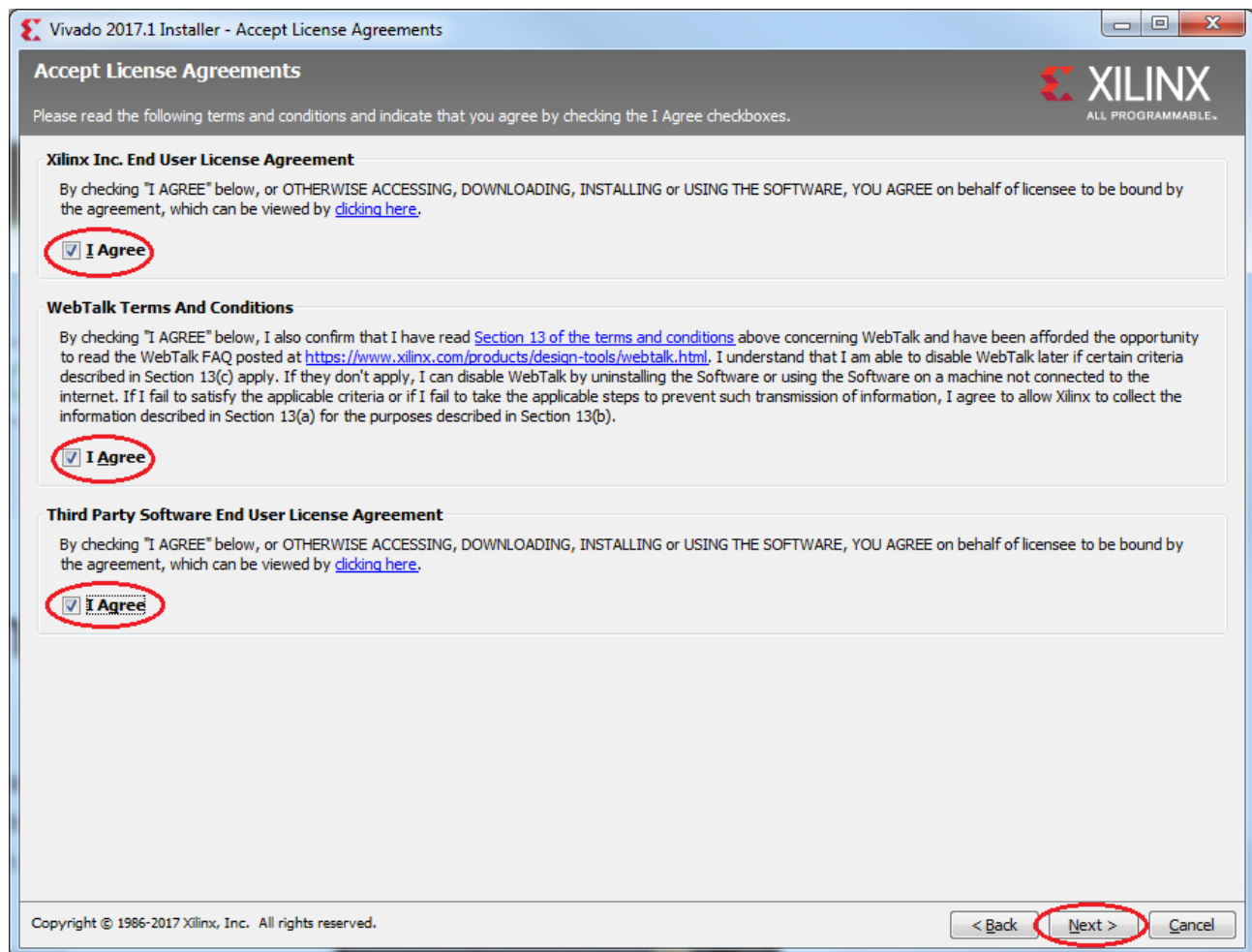


Figure 1 – Vivado Installer check agreements

3) Select the Vivado HL WebPACK and click 'Next'

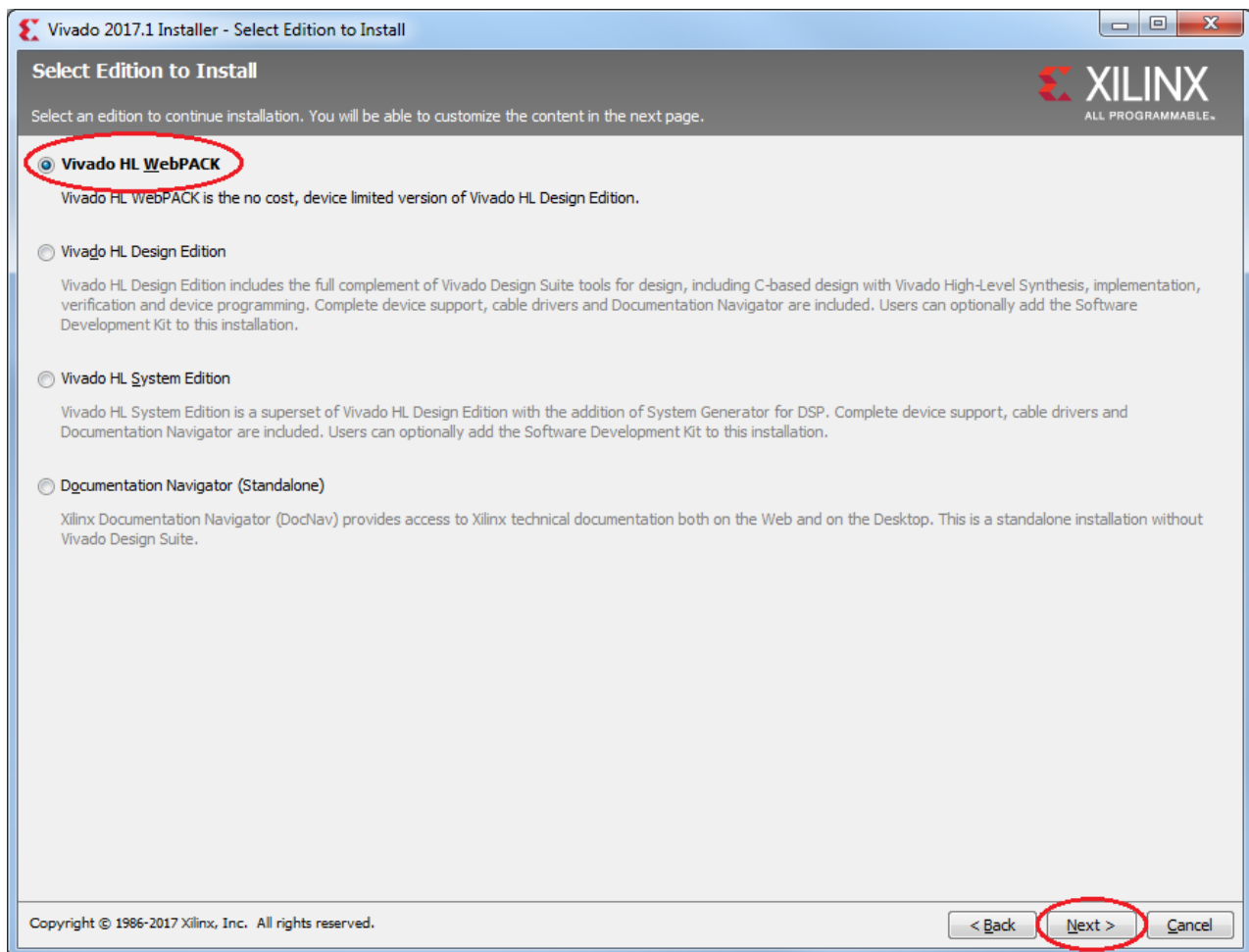


Figure 2 – Vivado Installer select WebPACK

- 4) Check the Software Development Kit (SDK)
Un-check DocNav
Un-check 7 Series, UltraScale and UltraScale+
Expand SoC's and un-check Zynq UltraScale+ MPSoC
Click 'Next'

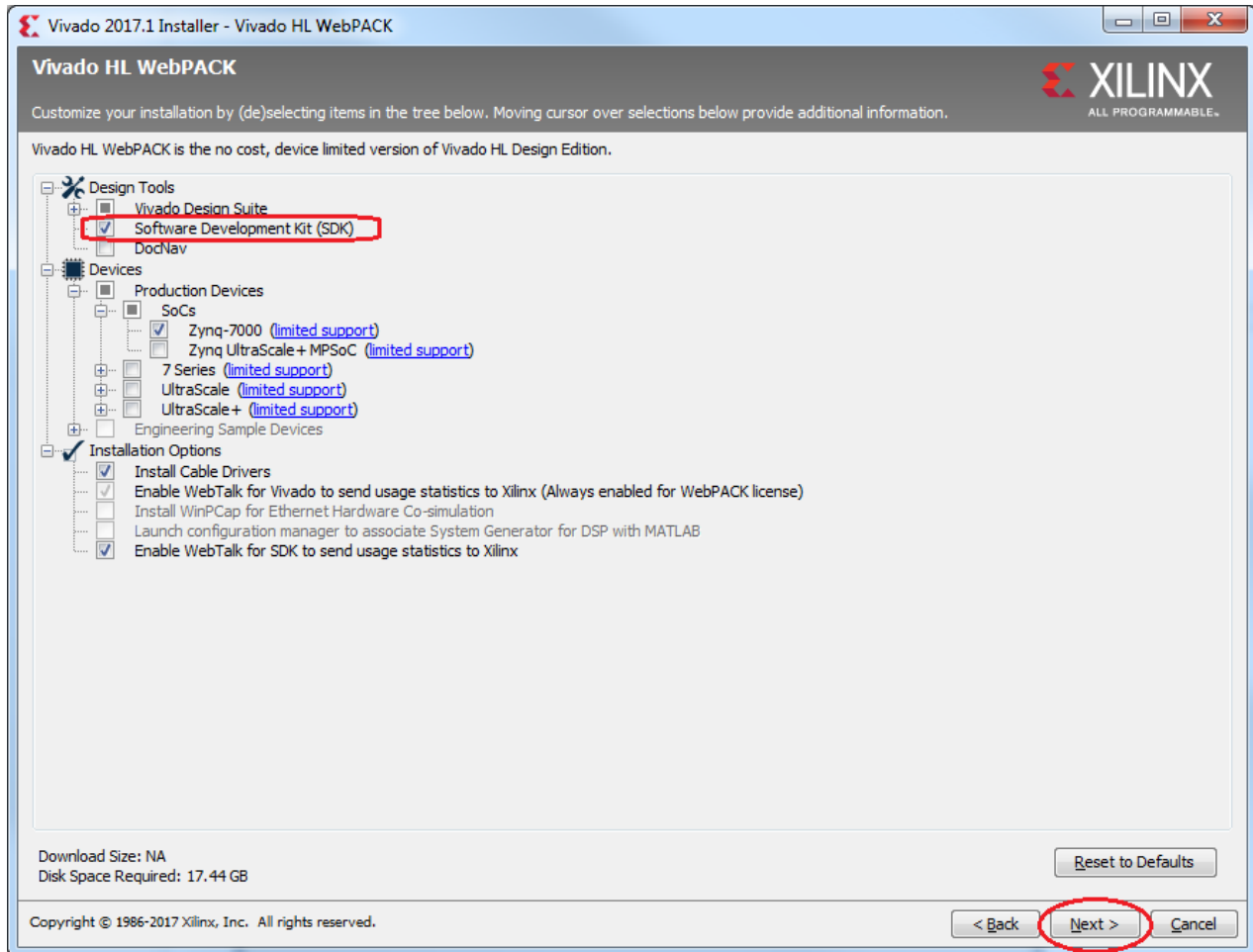


Figure 3 – Vivado Installer select SDK

- 5) Make sure there is no problem with the install directory or disk space, click 'Next' and start the installation.

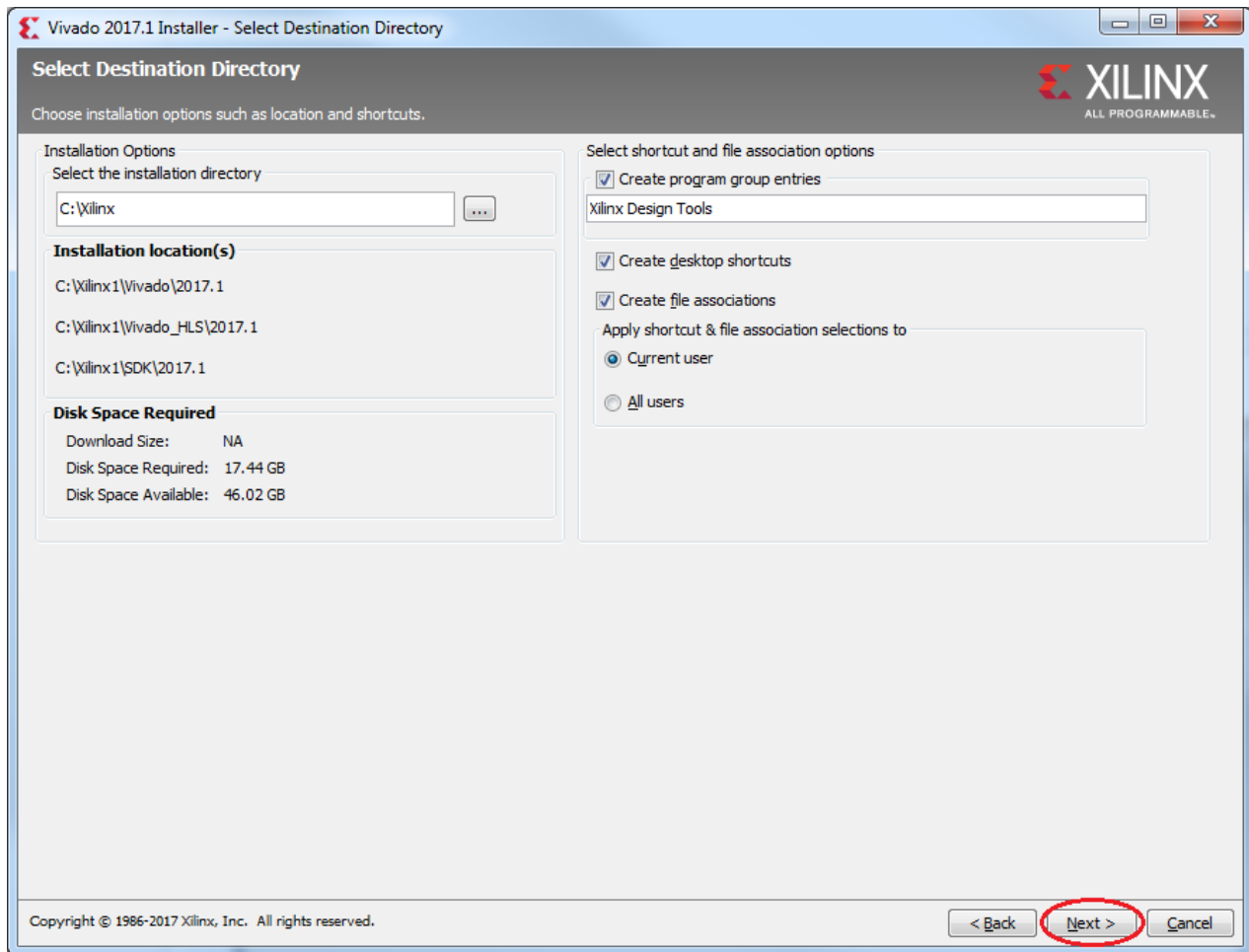


Figure 4 – Vivado Installer Destination Directory

Terminal Configuration

The serial text terminal program must be configured for the right COM port and to use 115200 as the baudrate.

When using Tera Term, for example, you have to select Serial and select the COM port that is detected when you plug in a MiniZed board. Then click 'OK'.

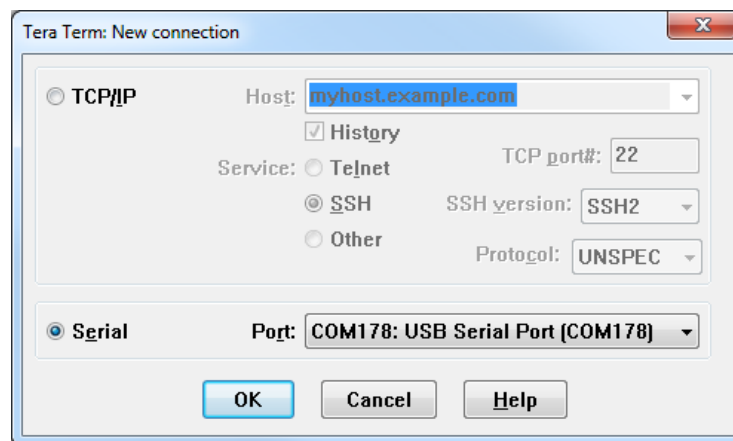


Figure 5 – Selecting the Tera Term COM port

You can specify the baudrate in Setup -> Serial Port. Make sure that **115200** is selected and that the other settings also match Figure 6 below.

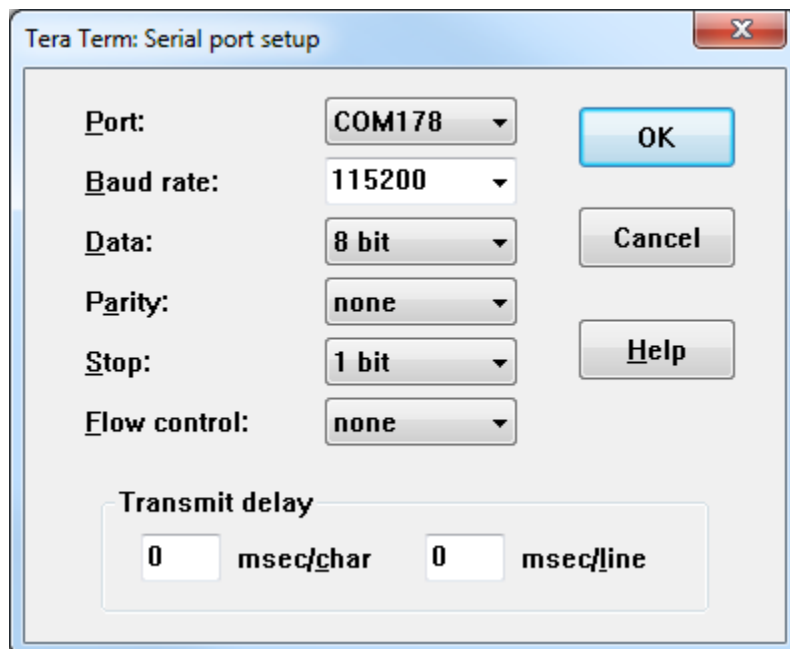


Figure 6 – Selecting the Tera Term baudrate

Programming the QSPI Flash, U2

Hardware Setup

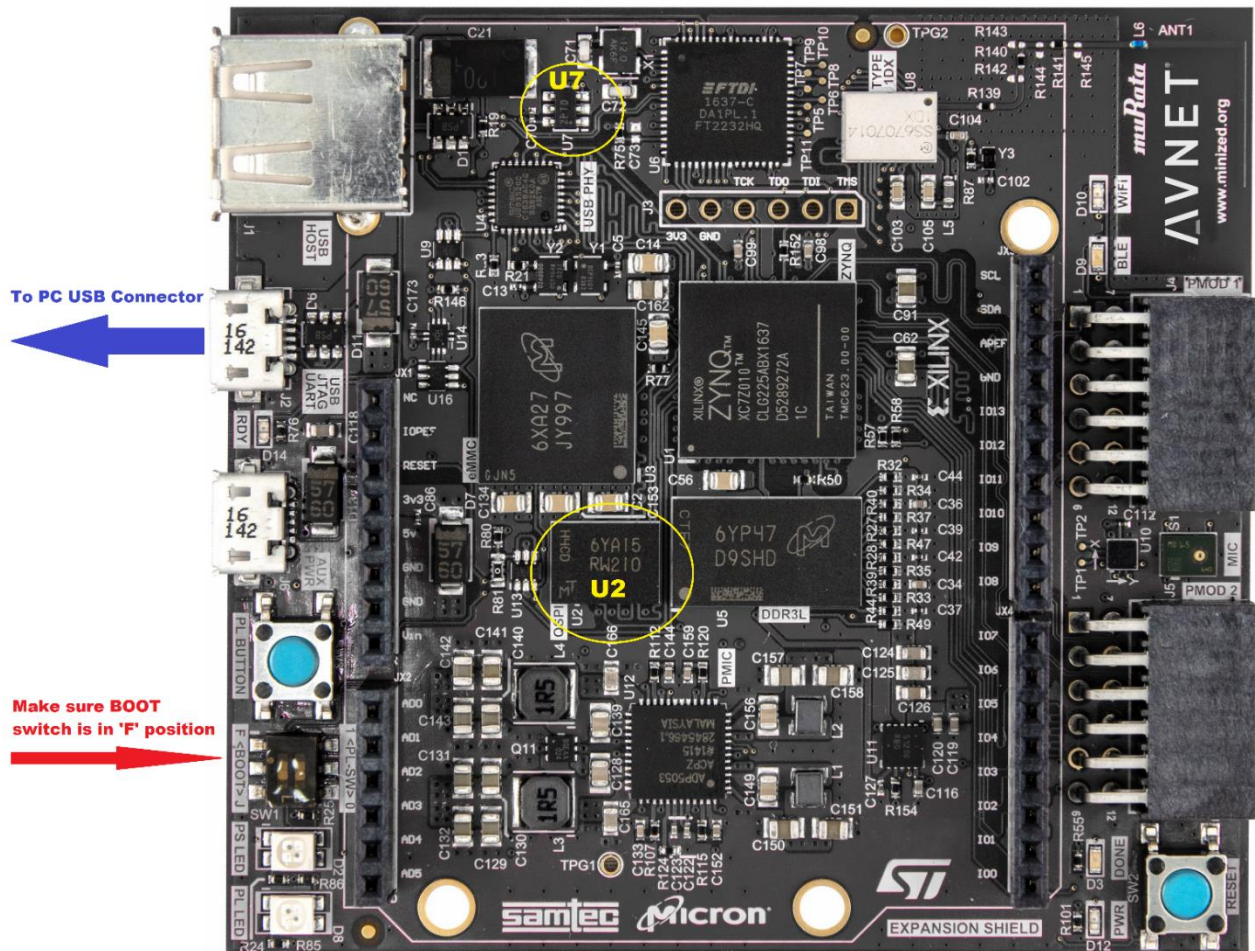


Figure 7 – MiniZed programming setup

Configure the board as in Figure 7 above:

- Connect a Micro USB cable from J2 to a USB connector on your PC/laptop.
- Make sure that the BOOT mode switch, which is the switch of SW1 that is closes to the board edge is in the 'F' position, i.e. towards the PS BUTTON and not towards the PS LED.

Programming the QSPI Flash Using XSCT

- 6) Once the tools are installed, click
Windows Start->Xilinx Design Tools->2017.1->Xilinx Software Command Line Tool
to start XSCT. Note that this is shown for 2017.1, but a later tool version can be used.

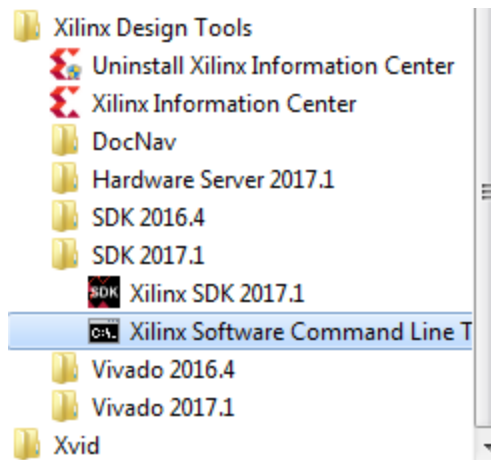
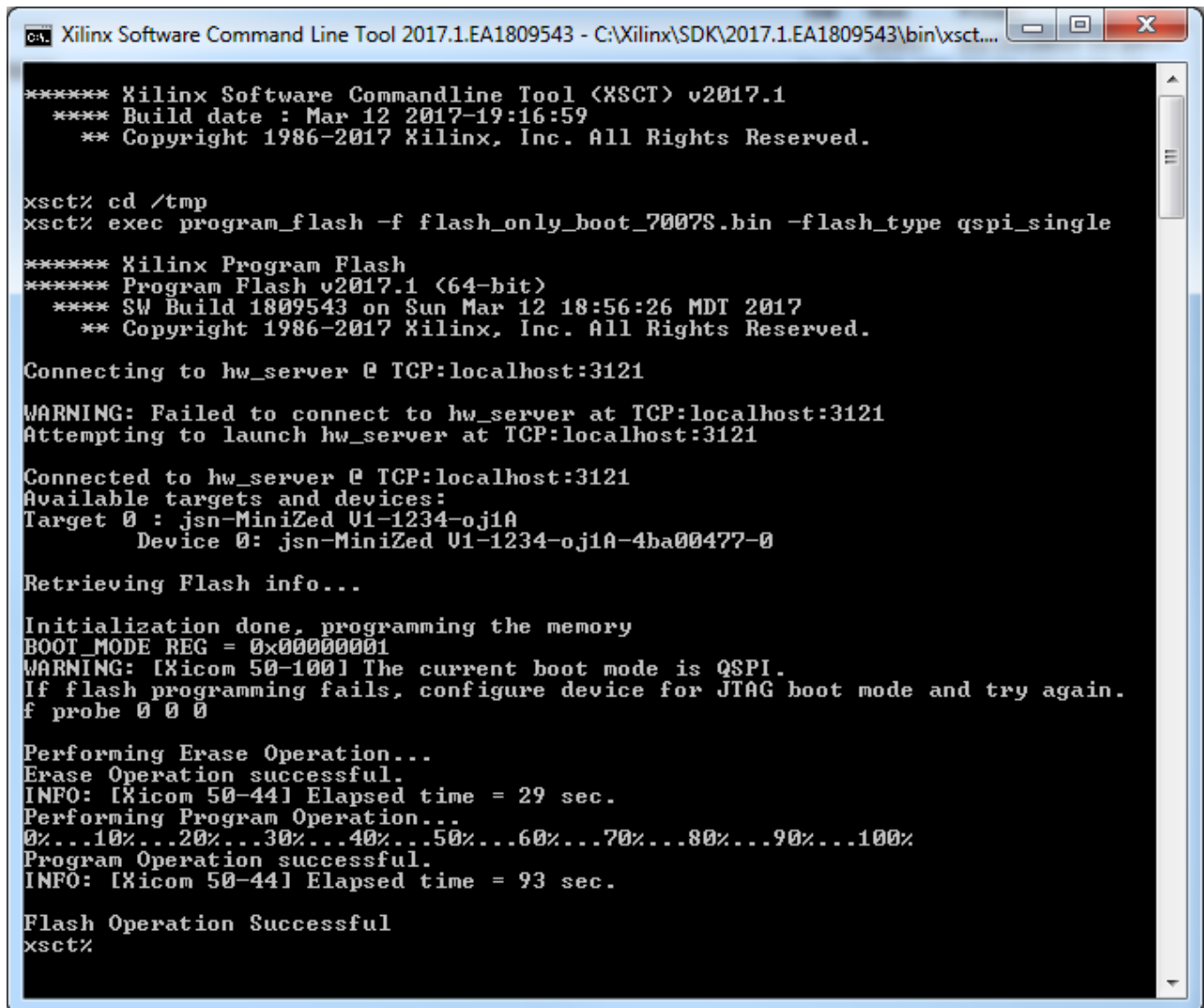


Figure 8 – Launching XSCT from the Windows Start menu

- 7) The file to be programmed into U2 can be found in the “Micron QSPI flash” sub-directory of the “Programming Files” directory of the zip archive that accompanies this document. Copy the file to be programmed, **flash_only_boot_7007S.bin**, to a directory, for example C:/tmp
- 8) As of XSCT 2017.2, the tools require that the FSBL (First-Stage Boot Loader) also be specified. For this, also copy the file **zynq_fsbl.elf** to the directory above (C:/tmp).
- 9) In the XSCT tool, at the xsct% prompt enter:
cd /tmp
- 10) Then, to program the flash enter:
exec program_flash -f flash_only_boot_7007S.bin -fsbl zynq_fsbl.elf -flash_type qspi_single

Note: Under XSCT 2017.1 it was sufficient to enter only:
exec program_flash -f flash_only_boot_7007S.bin -flash_type qspi_single

11) Successful programming will result in an output as in the image below (2017.1).



```
C:\> Xilinx Software Command Line Tool 2017.1.EA1809543 - C:\Xilinx\SDK\2017.1.EA1809543\bin\xsct...

***** Xilinx Software Commandline Tool (XSCT) v2017.1
***** Build date : Mar 12 2017-19:16:59
***** Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.

xsct% cd /tmp
xsct% exec program_flash -f flash_only_boot_7007S.bin -flash_type qspi_single

***** Xilinx Program Flash
***** Program Flash v2017.1 (64-bit)
***** SW Build 1809543 on Sun Mar 12 18:56:26 MDT 2017
***** Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.

Connecting to hw_server @ TCP:localhost:3121
WARNING: Failed to connect to hw_server at TCP:localhost:3121
Attempting to launch hw_server at TCP:localhost:3121

Connected to hw_server @ TCP:localhost:3121
Available targets and devices:
Target 0 : jsn-MiniZed U1-1234-oj1A
Device 0: jsn-MiniZed U1-1234-oj1A-4ba00477-0

Retrieving Flash info...

Initialization done, programming the memory
BOOT_MODE REG = 0x00000001
WARNING: [Xicom 50-100] The current boot mode is QSPI.
If flash programming fails, configure device for JTAG boot mode and try again.
f probe 0 0 0

Performing Erase Operation...
Erase Operation successful.
INFO: [Xicom 50-44] Elapsed time = 29 sec.
Performing Program Operation...
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Program Operation successful.
INFO: [Xicom 50-44] Elapsed time = 93 sec.

Flash Operation Successful
xsct%
```

Figure 9 – Programming the QSPI flash using XSCT

Updating MiniZed for Secondary boot from eMMC

Preparing the USB Memory Stick

Please verify that the USB memory stick is formatted as FAT or FAT32.

Three files from the zip archive that is associated with this document have to be copied to the root of the USB memory stick.

- smallboot.bin
- image.ub
- wpa_supplicant.conf

So that you can test the Wi-Fi, you have to edit the **wpa_supplicant.conf** file with a text editor like Notepad:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
    key_mgmt=WPA-PSK
    ssid="test_HotelStef"    ← Put the SSID of your Wi-Fi access point
    psk="test_74248483"      ← Put the password for your SSID here
}
```

Figure 10 – Editing wpa_supplicant.conf

Hardware Setup

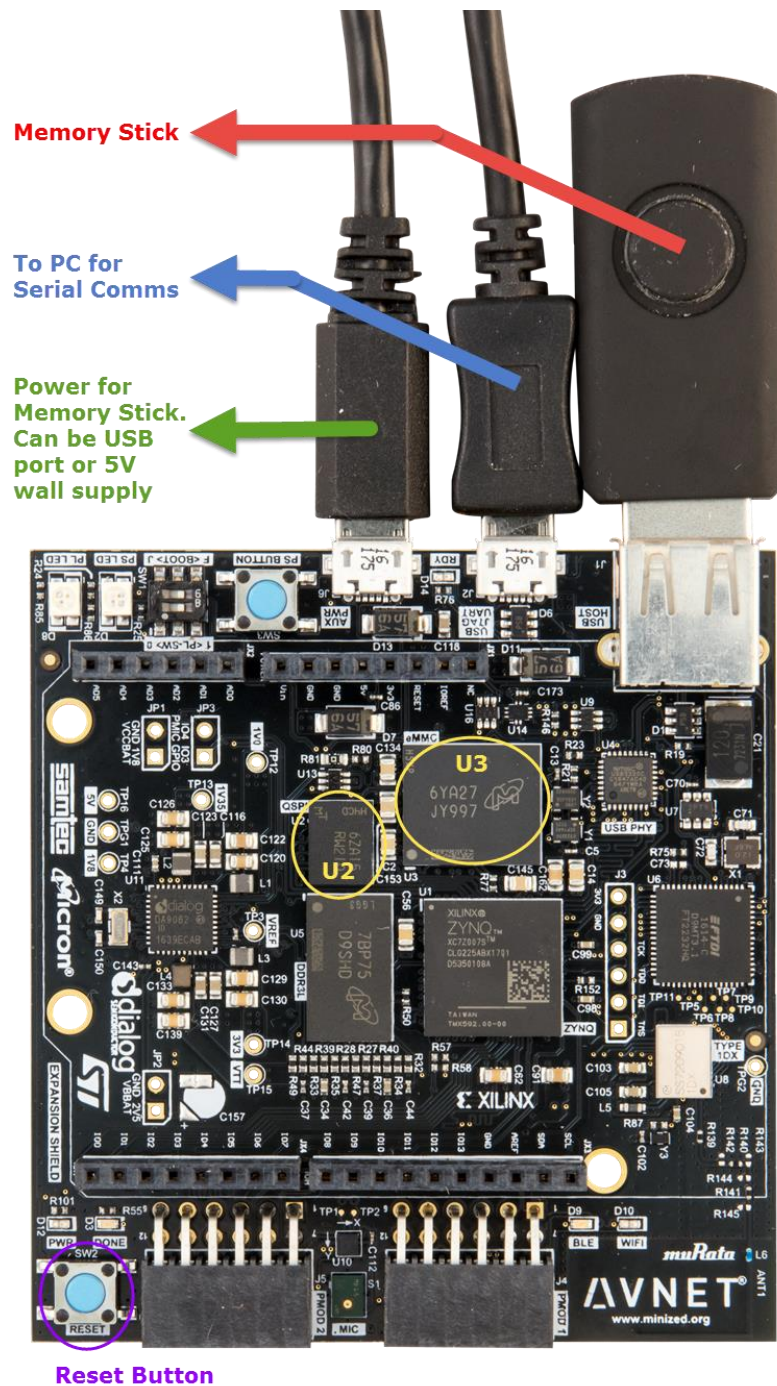


Figure 11 – MiniZed with Memory Stick setup

Connect the board as in Figure 7 above:

- Place the USB stick in J1, the USB Type A receptacle.

- Connect the first Micro USB cable between J2 and your PC/laptop.
- Connect the second Micro USB cable to a 5V supply. This can be a wall supply or a spare port on the test PC/laptop.
- Make sure the Boot Mode switch is set to the 'F' position for boot from flash, as in Figure 12 below. This should be the default position. The position of the other switch can be the same, also towards the USB connectors.

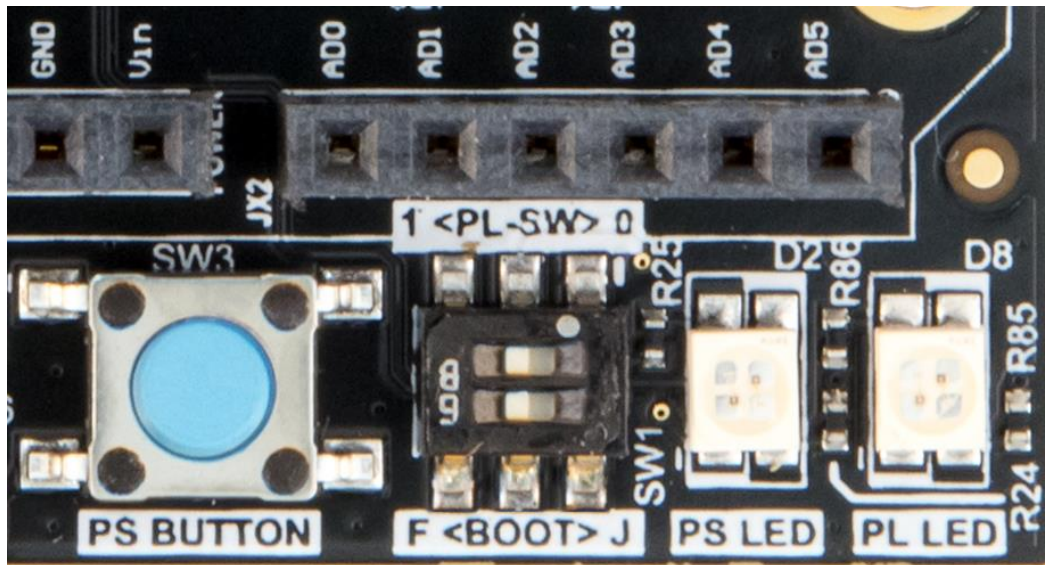


Figure 12 – Boot Mode switch position

Sequence for Programming-from-Memory-Stick

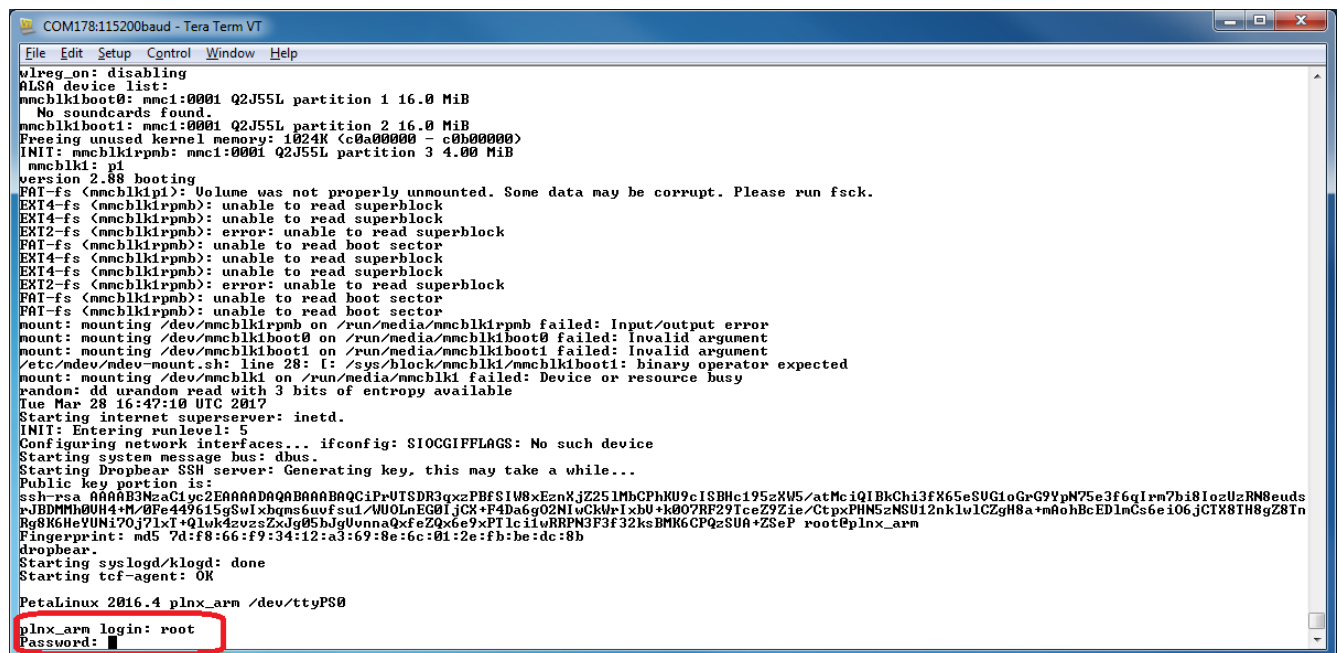
Step 1 – Check the default LEDs

When power is applied to the board, the LEDs should behave as follows:

- The **green power LED**, D12, must be on.
- The **PS LED**, D2, should be amber.
- The **PL LED**, D8, will reflect microphone activity. It should be off when there is silence and will flicker with input.
- The **amber debug port RDY LED**, D14 will come on when the USB FTDI driver enumerates the connection through J2.
- The **blue Configuration Done LED**, D3, should be off during reset but will come on within a second after that.

Step 2 – Linux Boot Process

Press the reset pushbutton, SW2, indicated in Figure 7. The terminal should display the output of the Linux boot process. It should halt at the login password prompt. Enter “root” as the login and also “root” as the password, as indicated in Figure 13.



```
COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help
wlrreg_on: disabling
ALSA device list:
mmcblk1boot0: mmc1:0001 Q2J55L partition 1 16.0 MiB
  No soundcards found.
mmcblk1boot1: mmc1:0001 Q2J55L partition 2 16.0 MiB
Freeing unused kernel memory: 1824K (c9a00000 - c0b00000)
INIT: mmcblk1rpnb: mmc1:0001 Q2J55L partition 3 4.00 MiB
mmcblk1: p1
version 2.88 booting
FAT-fs (mmcblk1p1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
EXT4-fs (mmcblk1rpnb): unable to read superblock
EXT4-fs (mmcblk1rpnb): unable to read superblock
EXT2-fs (mmcblk1rpnb): error: unable to read superblock
FAT-fs (mmcblk1rpnb): unable to read boot sector
EXT4-fs (mmcblk1rpnb): unable to read superblock
EXT4-fs (mmcblk1rpnb): unable to read superblock
EXT2-fs (mmcblk1rpnb): error: unable to read superblock
FAT-fs (mmcblk1rpnb): unable to read boot sector
FAT-fs (mmcblk1rpnb): unable to read boot sector
mount: mounting /dev/mmcblk1rpnb on /run/media/mmcblk1rpnb failed: Input/output error
mount: mounting /dev/mmcblk1boot0 on /run/media/mmcblk1boot0 failed: Invalid argument
mount: mounting /dev/mmcblk1boot1 on /run/media/mmcblk1boot1 failed: Invalid argument
/etc/udev/mdev-mount.sh: line 28: I: /sys/block/mmcblk1/mmcblk1boot1: binary operator expected
mount: mounting /dev/mmcblk1 on /run/media/mmcblk1 failed: Device or resource busy
random: dd urandom read with 3 bits of entropy available
Tue Mar 28 16:47:10 UTC 2017
Starting internet superserver: inetd.
INIT: Entering runlevel: 5
Configuring network interfaces... ifconfig: SIOCGIFFLAGS: No such device
Starting system message bus: dbus.
Starting Dropbear SSH server: Generating key, this may take a while...
Public key portion is:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAiPrUTSDR3qxzPBFsIU8xEnXjZ251MbCPhKU9cISBHC195zXW5/atMcIQIBkCh13fX65eSUG1oGrG9YpN75e3f6q1rm7b181ozUzRN8euds
rJBDMMH0UH4+M/0Fe449615gSulxbqms6uofsu1/WUOLnEG01jCX+P4Da6g02N1wCkMw1xbU+k007RF29IceZ9Z1e/CtpxPHN5zNSU12nklw1CZgH8a+naohBcED1nCs6e106jCTX8TH8gZ8In
RgB6heYUHI70J71xT+Q1u4ezvzZxJg05bJgUonnaQxFeZ4x0e9xPT1c11wRRP3F3F32k3BMK6CPqzSUA+ZSeP root@plnx_arm
Fingerprint: md5 7d:f8:66:f9:34:12:a3:69:8e:6c:01:2e:fb:be:dc:8b
dropbear.
Starting syslogd/klogd: done
Starting tcf-agent: OK
PetaLinux 2016.4 plnx_arm /dev/ttyPS0
plnx_arm login: root
Password: root
```

Figure 13 – Enter the plnx_arm login “root” and the Password “root”

Step 3 – eMMC Initialization, reprogramming and Hardware tests

At the prompt, enter:

onetest.sh

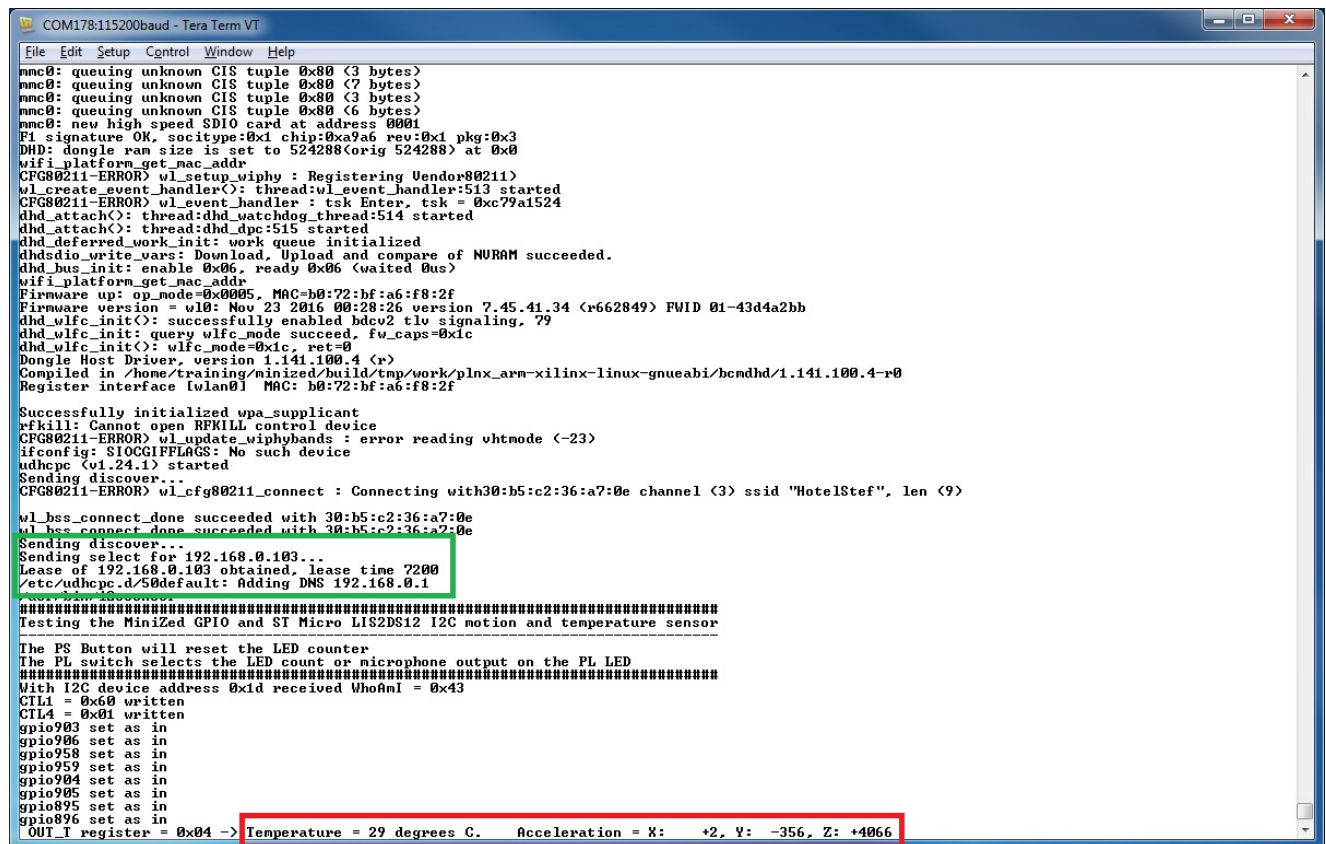
Tip: after only typing “on”, you can hit Tab and it will auto-complete

This should:

- format the eMMC
- copy files to eMMC from USB
- re-program the flash internally
- test the Wi-Fi and
- run a program that will test the I2C sensor while toggling the an LED
- after a key-press, shut down the system

There are a number of things that can be checked here.

- 1) Check that an **IP address** was obtained, as in the green box in Figure 14 below. Here the IP address is 192.168.0.103, but it could be another number also.
- 2) Check that Temperature and Acceleration values from the I2C sensor, U10, are displayed as in the red box in Figure 14 below.



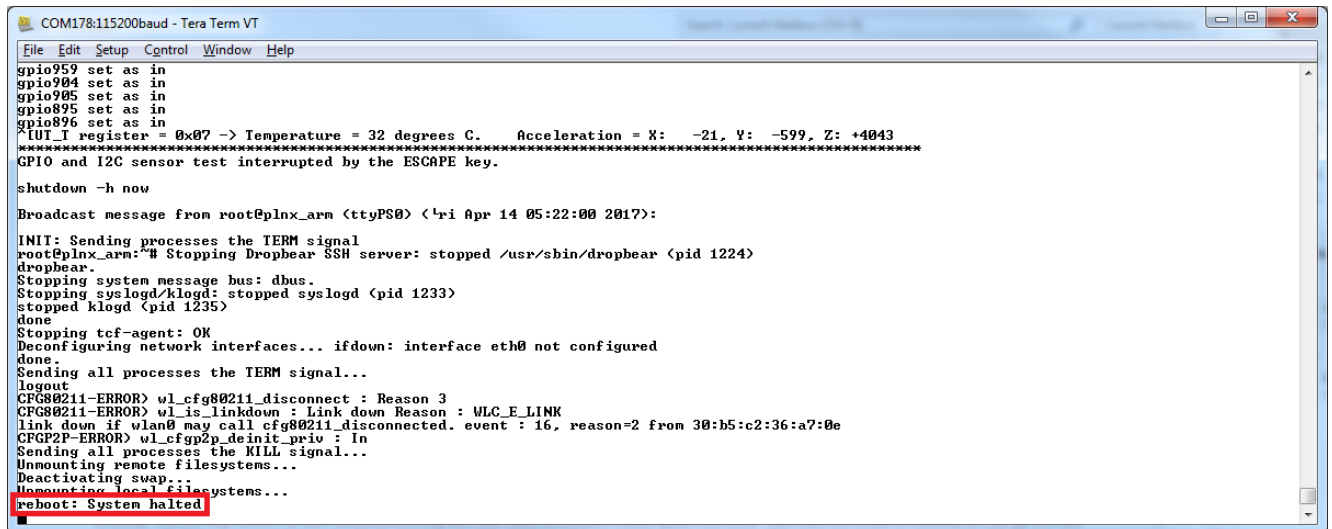
```
COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help
mmc0: queuing unknown CIS tuple 0x80 (3 bytes)
mmc0: queuing unknown CIS tuple 0x80 (7 bytes)
mmc0: queuing unknown CIS tuple 0x80 (3 bytes)
mmc0: queuing unknown CIS tuple 0x80 (6 bytes)
mmc0: new high speed SDIO card at address 0001
F1 signature OK, ecctype=0x1 chip:0xa9a6 rev:0x1 pkg:0x3
DHD: dongle ram size is set to 524288(orig 524288) at 0x0
wifi_platform_get_mac_addr
CFG80211-ERROR! wl_setup_wiphy : Registering Vendor80211
wl_create_event_handler(): thread:wl_event_handler:513 started
CFG80211-ERROR! wl_event_handler : tsk Enter, tsk = 0xc79a1524
dhd_attach(): thread:dhd_watchdog_thread:514 started
dhd_attach(): thread:dhd_dpc:515 started
dhd_deferred_work_init: work queue initialized
dhdssdio_write_vars: Download, Upload and compare of NVRAM succeeded.
dhd_bus_init: enable 0x06, ready 0x06 (waited 0us)
wifi_platform_get_mac_addr
Firmware up: op_mode=0x0005, MAC=b0:72:bf:a6:f8:2f
Firmware version = v10: Nov 23 2016 00:28:26 version 7.45.41.34 (r662849) FWID 01-43d4a2bb
dhd_wlfc_init(): successfully enabled bdcv2 tlv signaling, 79
dhd_wlfc_init: query wlfc_mode succeed, fw_caps=0x1c
dhd_wlfc_init(): wlfc_mode=0x1c, ret=0
Dongle Host Driver, version 1.141.100.4 (r)
Compiled in /home/training/minized/build/tnp/work/plnx_arm-xilinx-linux-gnueabi/bcmdhd/1.141.100.4-r0
Register interface wlan0 MAC: b0:72:bf:a6:f8:2f
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
CFG80211-ERROR! wl_update_wiphybands : error reading vhtmode (-23)
ifconfig: SIOCGIFFLAGS: No such device
udhcpc (v1.24.1) started
Sending discover...
CFG80211-ERROR! wl_cfg80211_connect : Connecting with30:b5:c2:36:a7:0e channel (3) ssid "HotelStef", len (9)
wl_bss_connect_done succeeded with 30:b5:c2:36:a7:0e
wl_bss_connect_done succeeded with 30:b5:c2:36:a7:0e
Sending discover...
Sending select for 192.168.0.103...
Lease of 192.168.0.103 obtained, lease time 7200
/etc/udhcpc.d/50default: Adding DNS 192.168.0.1
#####
Testing the MiniZed GPIO and ST Micro LIS2DS12 I2C motion and temperature sensor
The PS Button will reset the LED counter
The PL switch selects the LED count or microphone output on the PL LED
#####
With I2C device address 0x1d received WhoAmI = 0x43
C1L1 = 0x60 written
C1L4 = 0x01 written
gpio903 set as in
gpio906 set as in
gpio958 set as in
gpio959 set as in
gpio904 set as in
gpio905 set as in
gpio895 set as in
gpio896 set as in
OUT_I register = 0x04 -> Temperature = 29 degrees C. Acceleration = X: +2, Y: -356, Z: +4066
```

Figure 14 – Result of the “onetest.sh” script

- 3) When the Wi-Fi was enabled, the **amber Wi-Fi power LED**, D10, should have turned on.
- 4) Check that the **PS LED**, D2, is toggling in the sequence **red-green-amber-off**.
- 5) The **PL LED**, D8, by default represents the **volume of the microphone** output. With little audio, it should be off, with a speaking voice, it should flicker green. With a louder voice it will be a little amber and red.
- 6) Optional: If you move the **user switch**, which is next to the Boot Mode switch on SW1, to the '0' position (away from the USB connectors), both user LEDs should now toggle in a binary **red-green-amber-off** count.
- 7) Both user LEDs will remain off while you press the **user PS pushbutton**, SW3. If the user switch remains in the default '1' position, only the PS LED, D2, will be off.

Step 4 – Shut down, restart and run image.ub from eMMC

Press any key and wait for the Linux shutdown to complete. This is indicated when “reboot: System halted” is displayed as in Figure 15 below.



```

COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help
gpio959 set as in
gpio904 set as in
gpio905 set as in
gpio895 set as in
gpio896 set as in
^IUI_T register = 0x07 -> Temperature = 32 degrees C. Acceleration = X: -21, Y: -599, Z: +4043
*****
GPIO and I2C sensor test interrupted by the ESCAPE key.
shutdown -h now
Broadcast message from root@plnx_arm (ttyPS0) (Fri Apr 14 05:22:00 2017):
INIT: Sending processes the TERM signal
root@plnx_arm:~# Stopping Dropbear SSH server: stopped /usr/sbin/dropbear (pid 1224)
dropbear.
Stopping system message bus: dbus.
Stopping syslogd/klogd: stopped syslogd (pid 1233)
stopped klogd (pid 1235)
done
Stopping tcf-agent: OK
Deconfiguring network interfaces... ifdown: interface eth0 not configured
done.
Sending all processes the TERM signal...
logout
CFG80211-ERROR) wl_cfg80211_disconnect : Reason 3
CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_LINK
link down if wlan0 may call cfg80211_disconnected. event : 16, reason=2 from 30:b5:c2:36:a7:0e
CFG80211-ERROR) wl_cfg80211_deinit_priv : In
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
Unmounting local filesystems...
reboot: System halted

```

Figure 15 – Result of the shutdown process

Press the **reset pushbutton**, SW2, again. At the login password prompt, enter “**root**” as the login and also “**root**” as the password.

Step 5 – Test Bluetooth

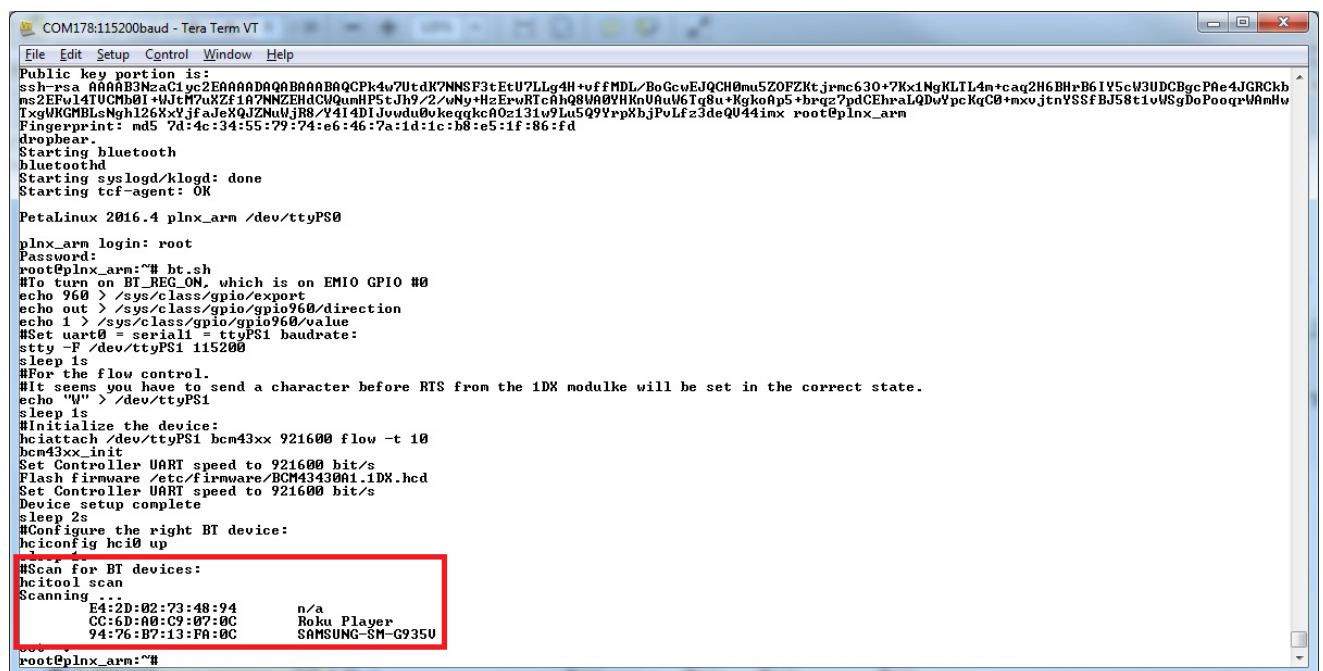
At the prompt, enter:

bt.sh

Tip: after only typing “bt.”, you can hit Tab and it will auto-complete

This test turns the blue **Bluetooth power LED**, D10, on before scanning for available Bluetooth devices. Once scanning completes, **discovered devices are listed** as in the red box in Figure 16 below.

Note: This test can take a while to complete. It is therefore **not necessary to run it for every device**. Spot checks should be sufficient. So once the blue LED comes on, it is OK to abort the test. It is important that **the bt.sh script must be run** though. This command is not available in the initial flash image. So if the command is available, this means that flash re-programming was successful and that a new image.ub was loaded from eMMC.



```
COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help
Public key portion is:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCPk4w7UtdK7NNSF3tEtU7LLq4H+offMDL/BoGcvEJQCH0mu5Z0FZKt.jrnc630+7Kx1NqKLT4n+caq2H6BH+B6IY5cW3UDCBypPa=4JGRChb
ms2EFu4TUCMb01+4UtM7u5ZF1A7NNZEHDcMQuMHP5tJh9/2/uNy+HzErwRTcAhQ8U60VHKnU8u06Tq8u+KqkoAp5+brqz7pdCEhralQDwYpcKqC0+mxvjtnYSSfBJ58t1v0SgDoPooqrWanHu
TxgVKGMBLsNgh126X+YjfadeXQJZnuUjR8/v414D1Jvudu0vkeqqkcA0z131v9Lu5Q9YrpXbjPvLfz3deQU44inx root@plnx_arm
Fingerprint: md5 7d:4c:34:55:79:74:e6:46:7a:1d:1c:b8:e5:1f:86:fd
dropbear.
Starting bluetooth
bluetoothd
Starting syslogd/klogd: done
Starting tcf-agent: OK
Petalinux 2016.4 plnx_arm /dev/ttyPS0
plnx_arm login: root
Password:
root@plnx_arm:~# bt.sh
#To turn on BT_REG_ON, which is on EMIO GPIO #0
echo 960 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio960/direction
echo 1 > /sys/class/gpio/gpio960/value
#Set uart0 = serial1 = ttyPS1 baudrate:
stty -F /dev/ttyPS1 115200
sleep 1s
#For the flow control.
#It seems you have to send a character before RTS from the 1DX module will be set in the correct state.
echo "u" > /dev/ttyPS1
sleep 1s
#Initialize the device:
hciattach /dev/ttyPS1 bcm43xx 921600 flow -t 10
bcm43xx_init
Set Controller UART speed to 921600 bit/s
Flash firmware /etc/firmware/BCM43430A1.1DX.hcd
Set Controller UART speed to 921600 bit/s
Device setup complete
sleep 2s
#Configure the right BT device:
hciconfig hci0 up
#Scan for BT devices:
hcitool scan
Scanning...
E4:2D:02:73:48:94 n/a
CC:6D:A0:C9:07:0C Roku Player
94:76:B7:13:FA:0C SAMSUNG-SM-G935U
root@plnx_arm:~#
```

Figure 16 – Result of the “bt.sh” script

Step 6 – Test the Dialog PMIC

The Dialog DA9062 Power-Management IC has an I2C interface that can be used to change the RTC (real-time clock) or to change the PMIC operating mode or even to adjust the pre-programmed voltage levels. Any adjustment of the power levels should of course be done with care, as changing these levels out of spec could damage components on MiniZed. Please refer to the DA9062 datasheet for details on the I2C registers and the associated command set.

At the prompt, enter:

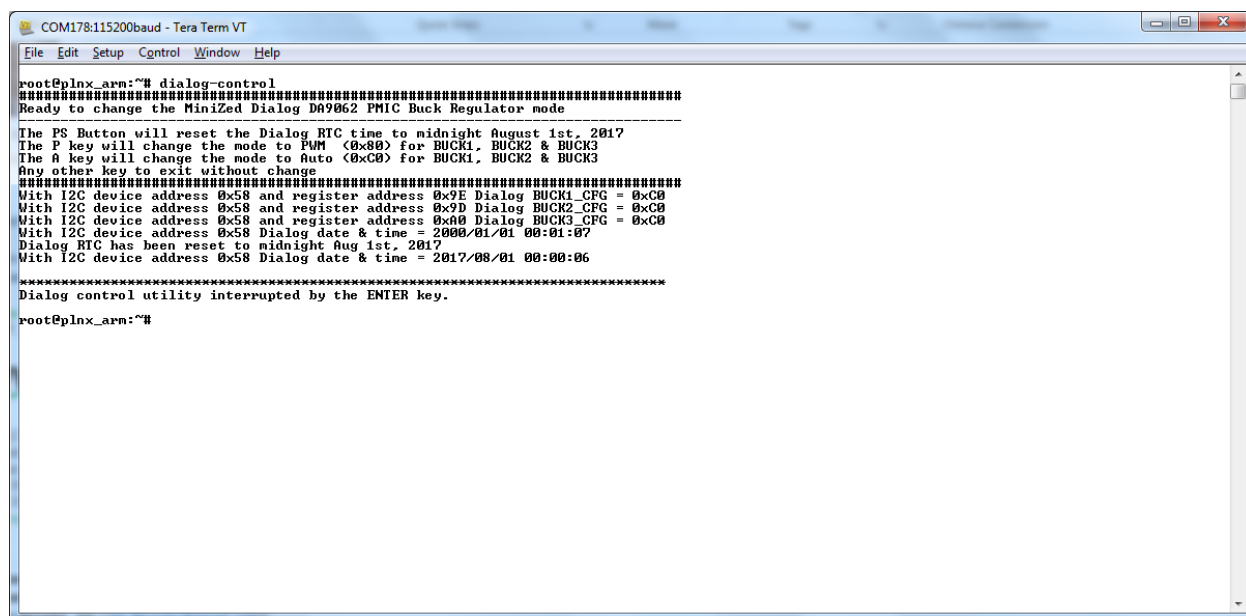
dialog-control

Tip: after only typing “dia”, you can hit Tab and it will auto-complete

On power-up, the PMIC should reset the RTC time to midnight on January 1st, 2000. The only way for the RTC to retain the time beyond a power cycle would be to add a super-cap or external battery supply, as per the schematic. The reset time plus any elapsed time since power-up should now display and should increment each second.

Press the PS button, SW3, on MiniZed. This will cause a new time of midnight August 1st, 2017 to be written to the RTC and for the time to increment from there.

Press any key to exit the utility.



```
COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help
root@plnx_arm:~# dialog-control
=====
Ready to change the MiniZed Dialog DA9062 PMIC Buck Regulator mode
=====
The PS Button will reset the Dialog RTC time to midnight August 1st, 2017
The P key will change the mode to PWM <0x80> for BUCK1, BUCK2 & BUCK3
The A key will change the mode to Auto <0xC0> for BUCK1, BUCK2 & BUCK3
Any other key to exit without change
=====
With I2C device address 0x58 and register address 0x9E Dialog BUCK1_CFG = 0xC0
With I2C device address 0x58 and register address 0x9D Dialog BUCK2_CFG = 0xC0
With I2C device address 0x58 and register address 0x9C Dialog BUCK3_CFG = 0xC0
With I2C device address 0x58 Dialog date & time = 2000/01/01 00:01:07
Dialog RTC has been reset to midnight Aug 1st, 2017
With I2C device address 0x58 Dialog date & time = 2017/08/01 00:00:06
=====
Dialog control utility interrupted by the ENTER key.
root@plnx_arm:~#
```

Figure 17 – Output from running the “dialog-control” utility

If you re-build the MiniZed PetaLinux project using the provided BSP on minized.org, you will be able to look at the source code for the “dialog-control” utility in the **dialog-control.cpp** file in the `~/<project>/project-spec/meta-user/recipes-apps/dialog-control/dialog-control` directory. It can be modified for user-specific tests.

Test Scripts

The **onetest.sh** script that is used in the programming/test sequence above combines many steps and this is fine when the tests all pass. When troubleshooting a failing board though, it is more useful to run smaller tests to establish where the board fault lies.

The available test scripts are in the Linux **/etc/local/bin/** folder. Useful scripts would be:

- **format_emmc.sh** - Formats the eMMC
- **update_from_usb.sh** - Copies files from the USB stick to the eMMC
- **program_qspi_from_usb.sh** - Internally program the QSPI flash with smallboot.bin
- **wifi.sh** - Makes the Wi-Fi connection
- **bt.sh** - Scans for Bluetooth devices
- **i2c_test.sh** - Reads the motion sensor and tests GPIO like LEDs

To make sure of exactly what they do, you can check the contents of any of these scripts by listing them.

For example:

cat /etc/local/bin/program_qspi_from_usb.sh

will display the contents of the routine that internally re-programs the QSPI flash.

Programming a Fall-back image into QSPI Flash

Since unexpected events could cause the image.ub in eMMC to fail, a safe recovery mechanism could be to have a backup image.ub in flash. If you followed the instructions above with the images in this file, you will have the fall-back image for PetaLinux 2017.4. Otherwise, you can follow the steps below to create your own. Steps are shown here for 2017.2 but you should use the latest version that is available.

Under the MiniZed Reference Designs on <http://zedboard.org/product/minized> there is a BSP (Board Support Package) named minized_qspi. Download the latest image.

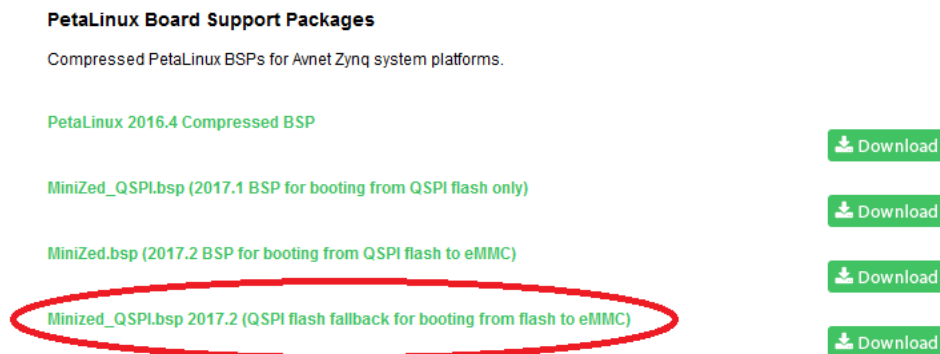


Figure 18 – Download minized_qspi.bsp

Under your PetaLinux development system you can create this as a project with

```
petalinux-create -t project -n minized_qspi -s ~/Documents/MiniZed/BSP/minized_qspi.bsp
```

and you can build the new project with

```
petalinux-build
```

When this completes, you can place the files bootgen.bif and boot_gen.sh in the project root and then create a binary file with

```
./boot_gen.sh
```

The output of this process is boot.bin, but it has been renamed to flash_fallback_7007S.bin and can be found in the “Micron QSPI flash” sub-directory of the “Programming Files” directory of the zip archive that accompanies this document. Copy the file to be programmed, **flash_fallback_7007S.bin**, as well as **zynq_fsbl.elf** to a directory, for example C:/tmp. In XSCT enter:

```
exec program_flash -f flash_fallback_7007S.bin -fsbl zynq_fsbl.elf -flash_type qspi_single
```

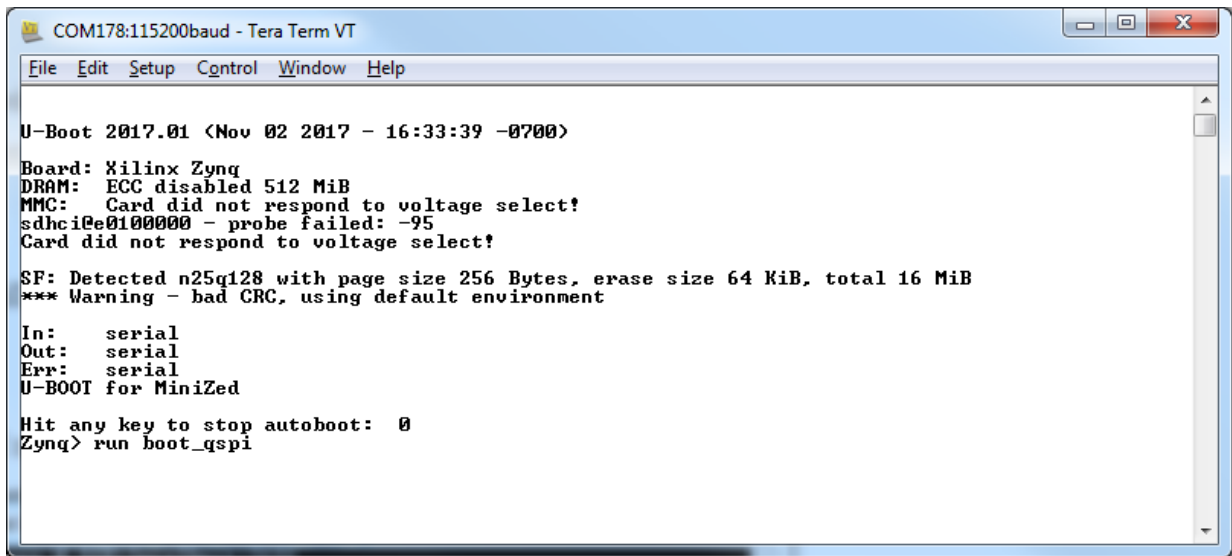
Wait for “Program Operation Successful”.

Alternatively, the binary can be programmed directly from eMMC with a command like

```
flashcp /mnt/emmc/flash_fallback_7007S.bin /dev/mtd0
```

When you reset MiniZed, it will automatically attempt to load image.ub from eMMC. But if it is not found, or if you interrupt u-boot by hitting any key, you will be able to run image.ub inside the flash by executing

run boot_qspi



```
COM178:115200baud - Tera Term VT
File Edit Setup Control Window Help

U-Boot 2017.01 <Nov 02 2017 - 16:33:39 -0700>
Board: Xilinx Zynq
DRAM: ECC disabled 512 MiB
MMC: Card did not respond to voltage select!
sdhci@01000000 - probe failed: -95
Card did not respond to voltage select!
SF: Detected n25q128 with page size 256 Bytes, erase size 64 KiB, total 16 MiB
*** Warning - bad CRC, using default environment

In: serial
Out: serial
Err: serial
U-BOOT for MiniZed
Hit any key to stop autoboot: 0
Zynq> run boot_qspi
```

Figure 19 – Run boot_qspi

This should bring up the image.ub that was built into the .bin file that was programmed into QSPI flash.

Erasing Environmental Variables

In some cases where environmental variables were modified or added and then saved to QSPI flash with 'saveenv', these settings can cause MiniZed not to boot normally, even after re-programming. Even though the programming process first erases the flash before programming, saveenv saves to a different area in flash and those variables could persist beyond re-programming.

Two options for dealing with this are covered below.

Erasing the QSPI Flash from u-boot

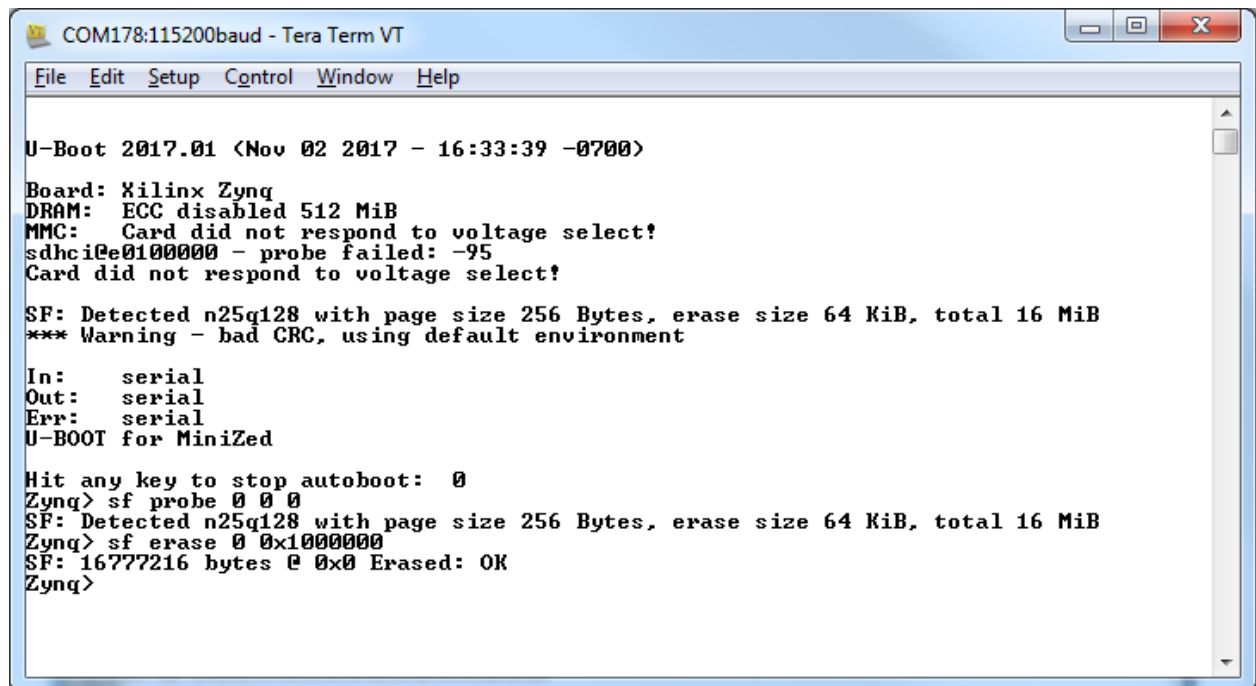
If the flash is able to boot into u-boot, the boot process can be interrupted by pressing any key while the "Hit any key to stop autoboot" countdown is active. After interrupting this countdown, enter

sf probe 0 0 0

followed by

sf erase 0 0x1000000

This should erase everything in flash. Then you can re-program it using XSCT as described earlier.

The image is a screenshot of a Tera Term VT window titled "COM178:115200baud - Tera Term VT". The window contains the following text:

```
U-Boot 2017.01 (Nov 02 2017 - 16:33:39 -0700)
Board: Xilinx Zynq
DRAM: ECC disabled 512 MiB
MMC: Card did not respond to voltage select!
sdhci@e0100000 - probe failed: -95
Card did not respond to voltage select!
SF: Detected n25q128 with page size 256 Bytes, erase size 64 KiB, total 16 MiB
*** Warning - bad CRC, using default environment

In: serial
Out: serial
Err: serial
U-BOOT for MiniZed

Hit any key to stop autoboot: 0
Zynq> sf probe 0 0 0
SF: Detected n25q128 with page size 256 Bytes, erase size 64 KiB, total 16 MiB
Zynq> sf erase 0 0x1000000
SF: 16777216 bytes @ 0x0 Erased: OK
Zynq>
```

Figure 20 – Use sf erase to clear everything in QSPI Flash

Programming the QSPI Flash from the Vivado Hardware Manager

XSCT is convenient and simple, but a very reliable method for programming the flash is to use the Vivado Hardware Manager. This method also provides the option of re-programming the entire flash, which will ensure that environmental variables are all erased and overwritten.

Note that since this chapter was added in November of 2017, **these steps are shown for Vivado 2017.3**. They may work in earlier or later versions and some steps may differ.

- Launch Vivado and Open the Hardware Manager

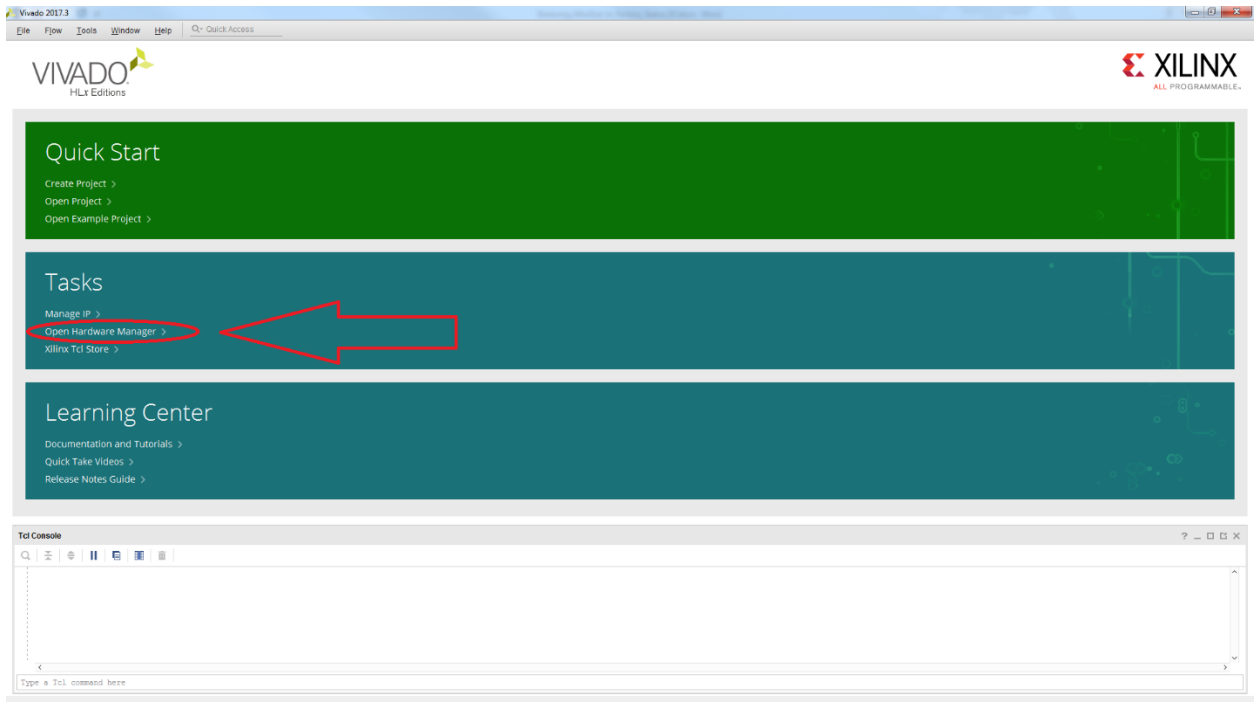


Figure 21 – Open the Vivado Hardware Manager

- Click 'Open Target', then 'Open New Target'

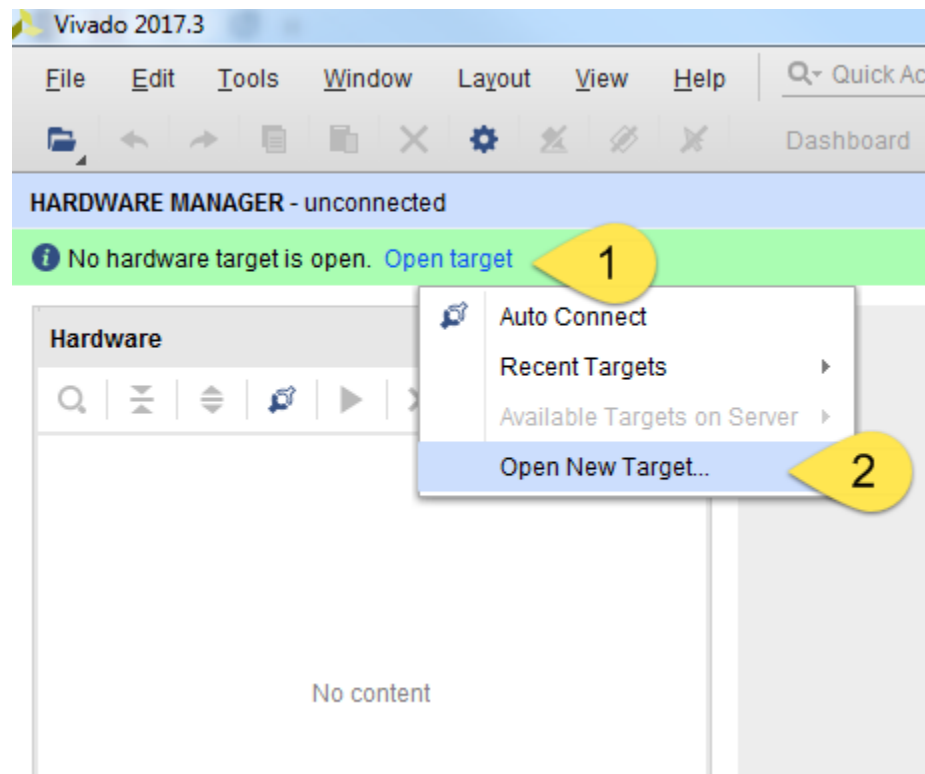


Figure 22 – Open New Target

- Click 'Next'
- Make sure that a MiniZed is connected to your machine via the micro USB cable to the J2 connector labelled 'USB JTAG UART'.
- Select 'Local server' in the drop-down box and click 'Next'.

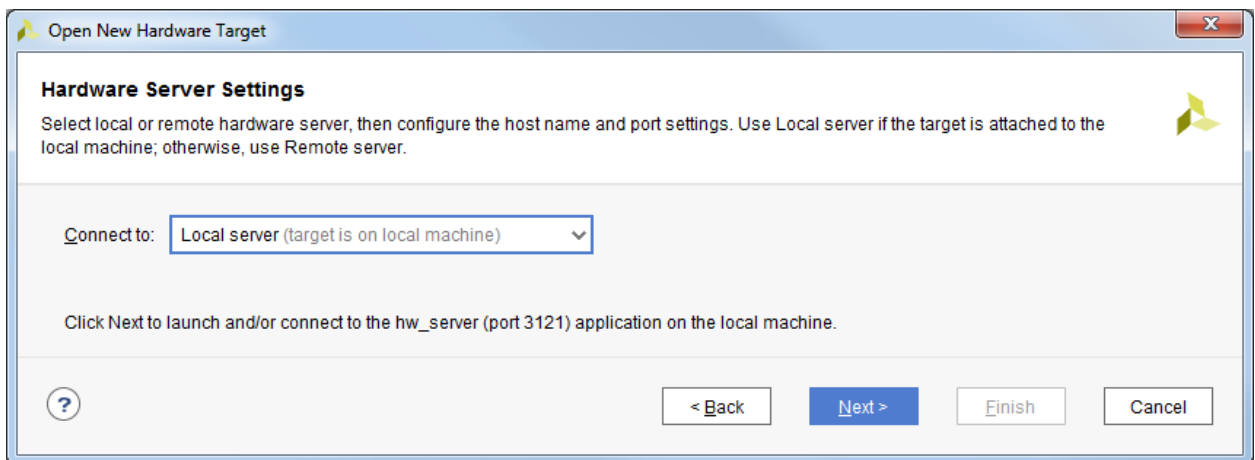


Figure 23 – Select the Local server

- Wait while the tool searches for and connects to the server. Then click 'Next'

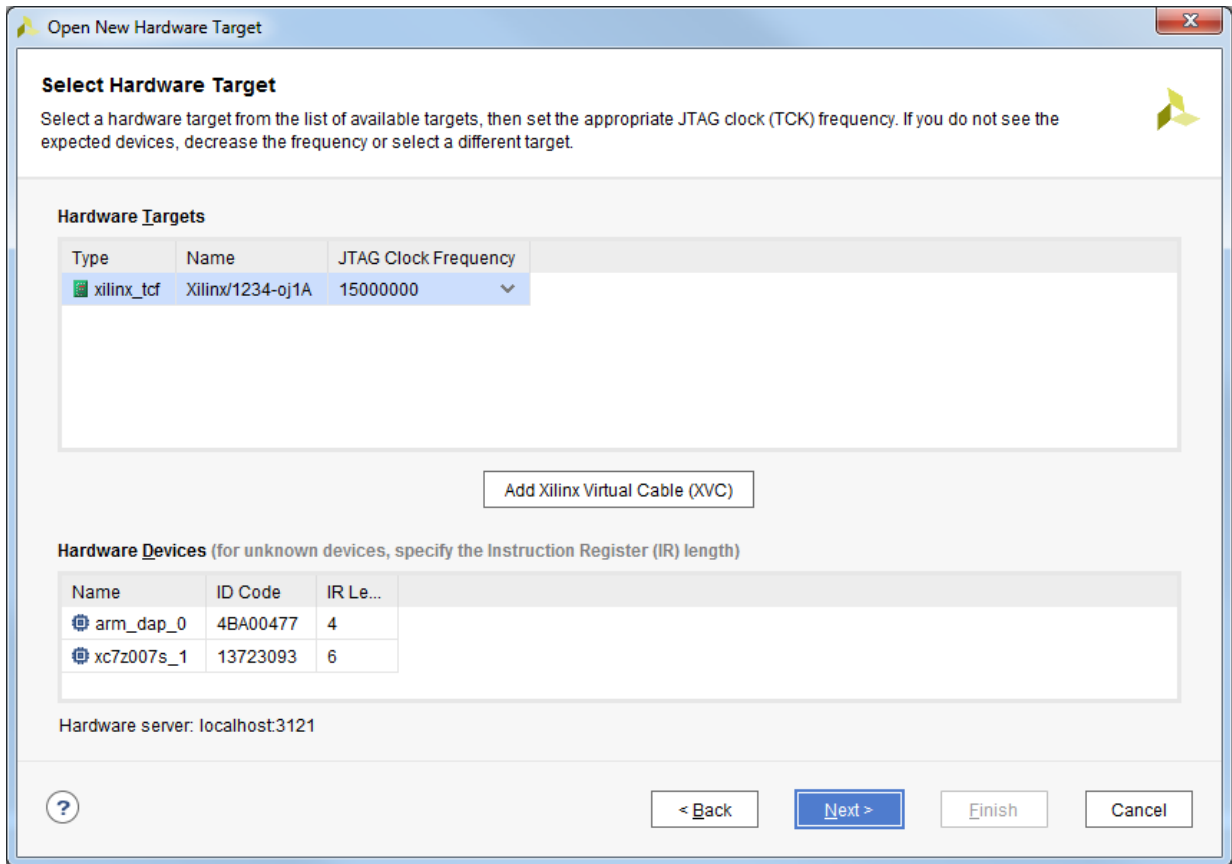


Figure 24 – Open a New Hardware Target

- Click 'Finish'.

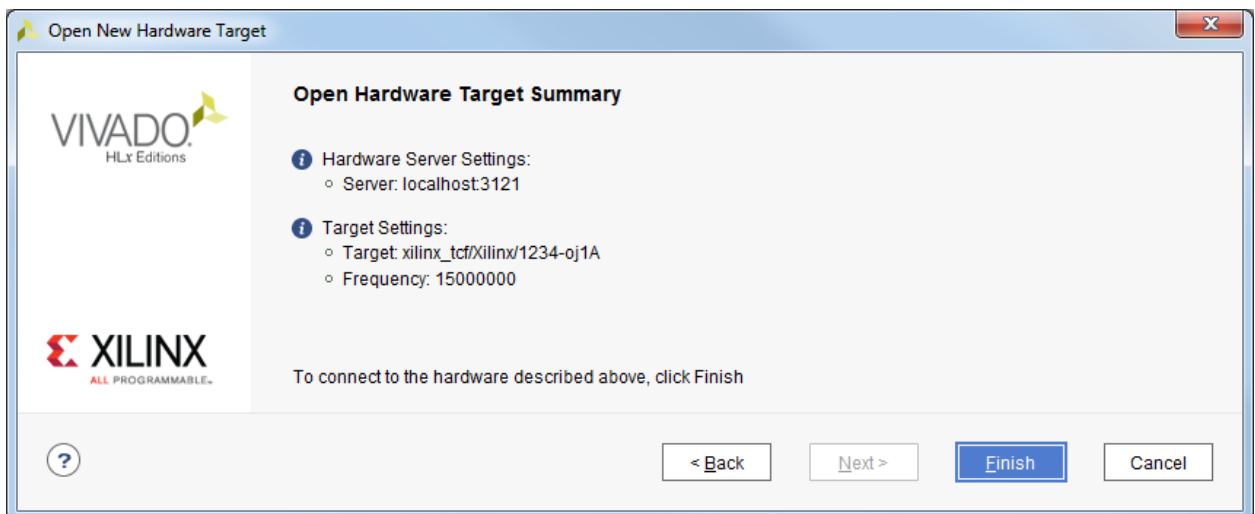


Figure 25 – Open New Target

- From the 'Tools' menu, select 'Add Configuration Memory Device' and 'xc7z007s_1'

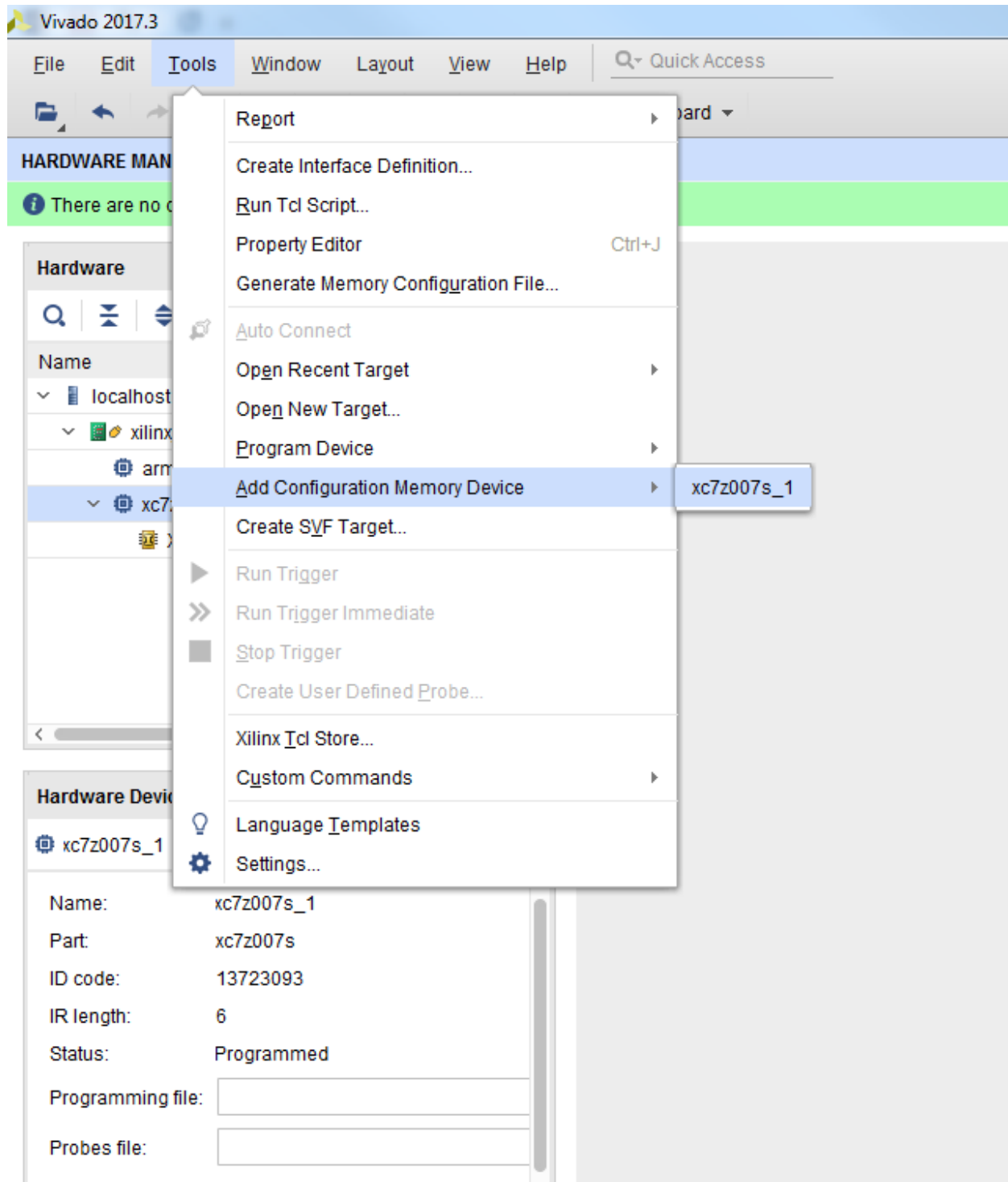


Figure 26 – Add a Configuration Memory Device

- In the selection filter, pick a Micron QSPI device of size 128Mb in single width. Then pick the mt25ql128-qspi-x1-single part and click 'OK'.

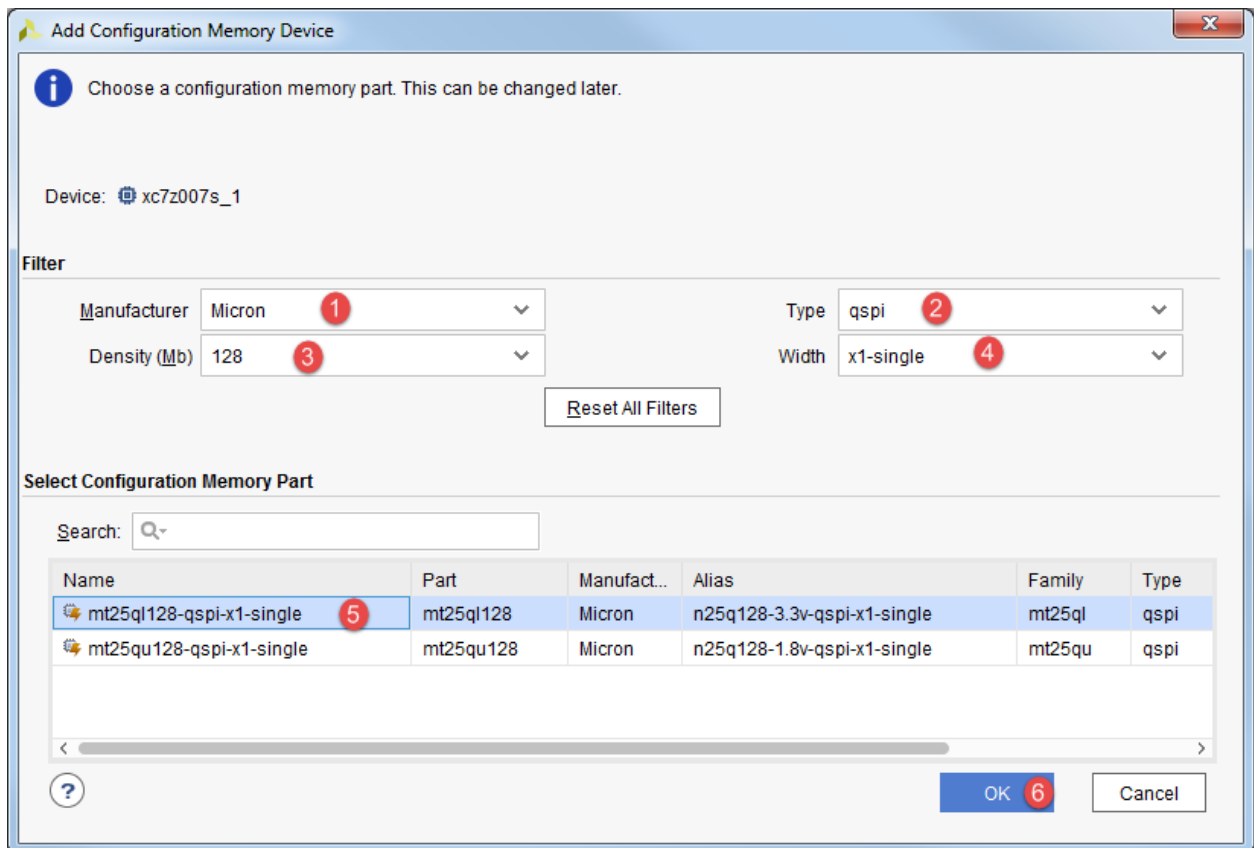


Figure 27 – Memory Device Selection

- If asked whether you want to program the configuration memory device now, click 'OK'.

- Pick the file you want to program, as well as the FSBL file, **zynq_fsbl.elf**.
- Select “**Entire Configuration Memory Device**”. Then click ‘OK”

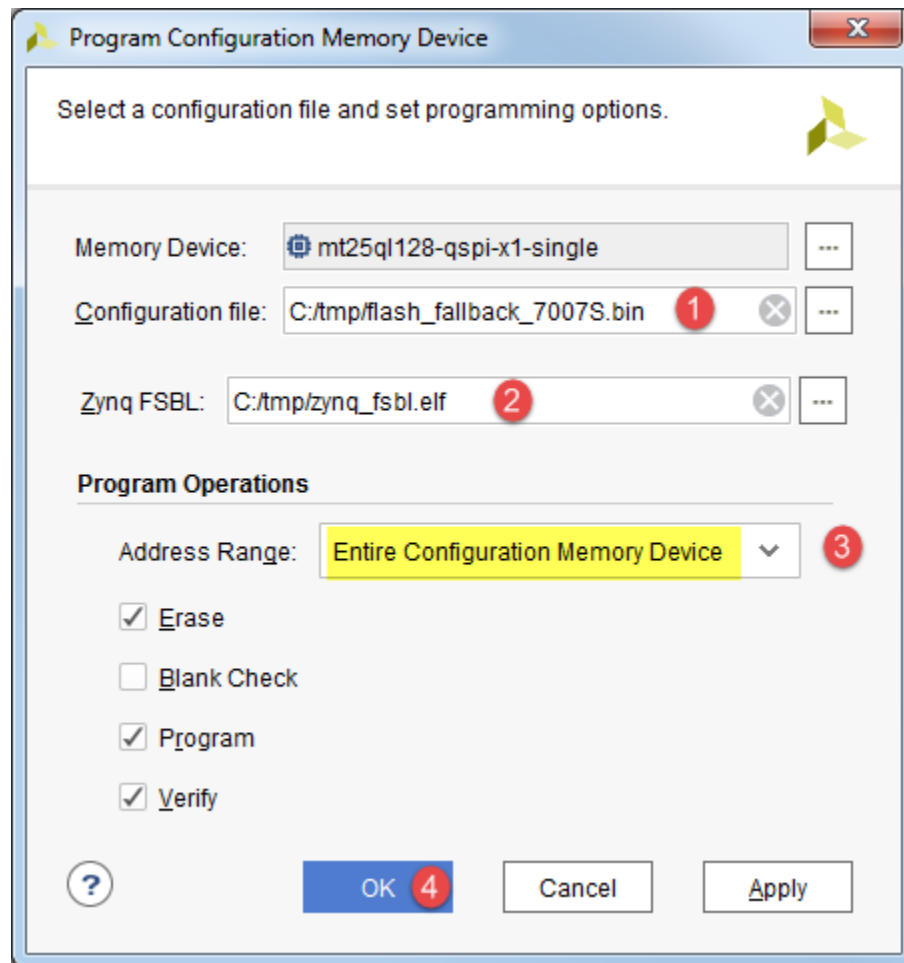


Figure 28 – Choose the file to Program into QSPI Flash

When programming completes, the entire flash will be re-programmed.

Revision History

Date	Version	Revision
09 Sep 17	01	Initial Release
20 Nov 17	02	Updates for using the FSBL and the flash fallback
10 Apr 18	03	Updates for 2017.4