DG0852
Demo Guide
PolarFire FPGA Temperature and Voltage Sensor

Microsemi
a Microchip company

Microsemi 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
Email: sales.support@microsemi.com
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.

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center 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; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 3.0

The following is a summary of the changes made in this revision.

- Added Appendix 2: Running the TCL Script, page 15.
- Updated Figure 2, page 4.
- Updated Figure 3, page 5.

1.2 Revision 2.0

The following is a summary of the changes made in this revision.

- Updated the document for Libero SoC v12.2.
- Removed the references to Libero version numbers.

1.3 Revision 1.0

The first publication of this document.

2 PolarFire FPGA Temperature and Voltage Sensor

Each PolarFire device is equipped with a Temperature and Voltage Sensor (TVS). TVS reports die temperature and voltage of device supply rails in digital form to the FPGA fabric.

TVS is implemented using a 4-channel ADC and the channel information is given as follows:

- Channel 0 – 1 V voltage supply
- Channel 1 – 1.8 V voltage supply
- Channel 2 – 2.5 V voltage supply
- Channel 3 – Die temperature

The TVS outputs a 16-bit encoded value that represents voltage or temperature, and corresponding channel number. The temperature and voltage information is translated into standard temperature and voltage values. For more information, see *UG0753: PolarFire FPGA Security User Guide*.

This demo highlights the TVS feature of the PolarFire using an UART-based application (GUI). The demo design continuously pumps the data from TVS channels to UART, which is displayed on the GUI. This demo design also shows how to simulate the TVS feature of the PolarFire device.

The demo design can be programmed using any of the following options:

- Using the job file: To program the device using the job file provided along with the design files, see Appendix 1: Programming the Device Using FlashPro Express, page 12.
- Using Libero SoC: To program the device using Libero SoC, see Libero Design Flow, page 8. Use this option when the demo design is modified.

2.1 Design Requirements

The following table lists the hardware and software requirements for this demo design.

Table 1 • **Design Requirements**

Requirement	**Version**
Operating system	64-bit Windows 7, 8, or 10
Hardware	
PolarFire Evaluation Kit (MPF300-EVAL-KIT)	Rev D or later
Software	
Libero SoC	**Note:** Refer to the readme.txt file provided in the design files for the software versions used with this reference design.
ModelSim	
FlashPro Express	

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

2.2 Prerequisites

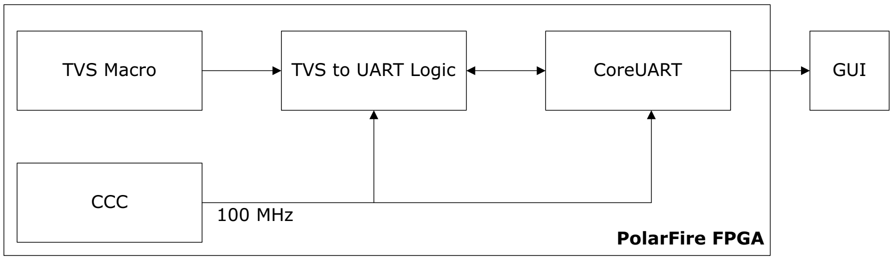
Before you begin:

1. For demo design files download link:
   http://soc.microsemi.com/download/rsc/?f=mpf_dg0852_df
2. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location:
   https://www.microsemi.com/product-directory/design-resources/1750-libero-soc
   The latest versions of ModelSim, Synplify Pro, and FTDI drivers are included in the Libero SoC installation package.

2.3 Demo Design

The top-level block diagram of the TVS design is shown in the following figure. All four channels of TVS are enabled in the design to monitor the die temperature and voltage rails. The Fabric logic captures the TVS channels output and sends to UART IF through CoreUART IP.

Figure 1 • **TVS Block Diagram**



The GUI receives channel wise TVS values and decodes as described to display them:

Die Temperature:

The temperature channels 16-bit output value is represented in Kelvin and can be decoded as listed in the following table. For example, the temperature channel's output value of 0x133B implies 307.56 Kelvin.

Table 2 • **Temperature Channel Value Decoding**

Bit Number	Description
15	Reserved
[14:4]	Integer value of temperature
[3:0]	Fractional value of temperature

Voltage:

The data present on the VALUE and CHANNEL outputs is valid only when the VALID output is asserted. When a channel is disabled by deasserting the corresponding channel enable input, then the channel data present on the outputs is not valid even if the VALID output is asserted. The voltage channels 16-bit output value is represented in millivolts (mV) and can be decoded as listed in the following table. For example, the voltage channels output value of 0x385E implies 1803.75 mV.

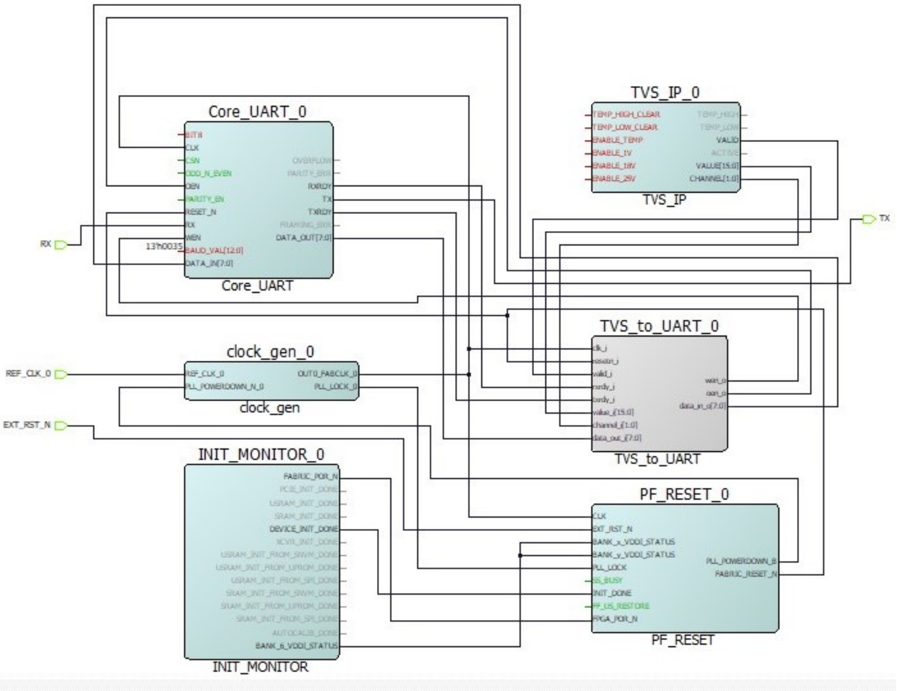
Table 3 • **Voltage Channel Value Decoding**

Bit Number	Description
15	Signed bit
[14:3]	Integer value of voltage
[2:0]	Fractional value of voltage

2.3.1 Design Implementation

The following figure shows the Libero SoC software design implementation of the TVS demo design.

Figure 2 • **TVS Demo Design**



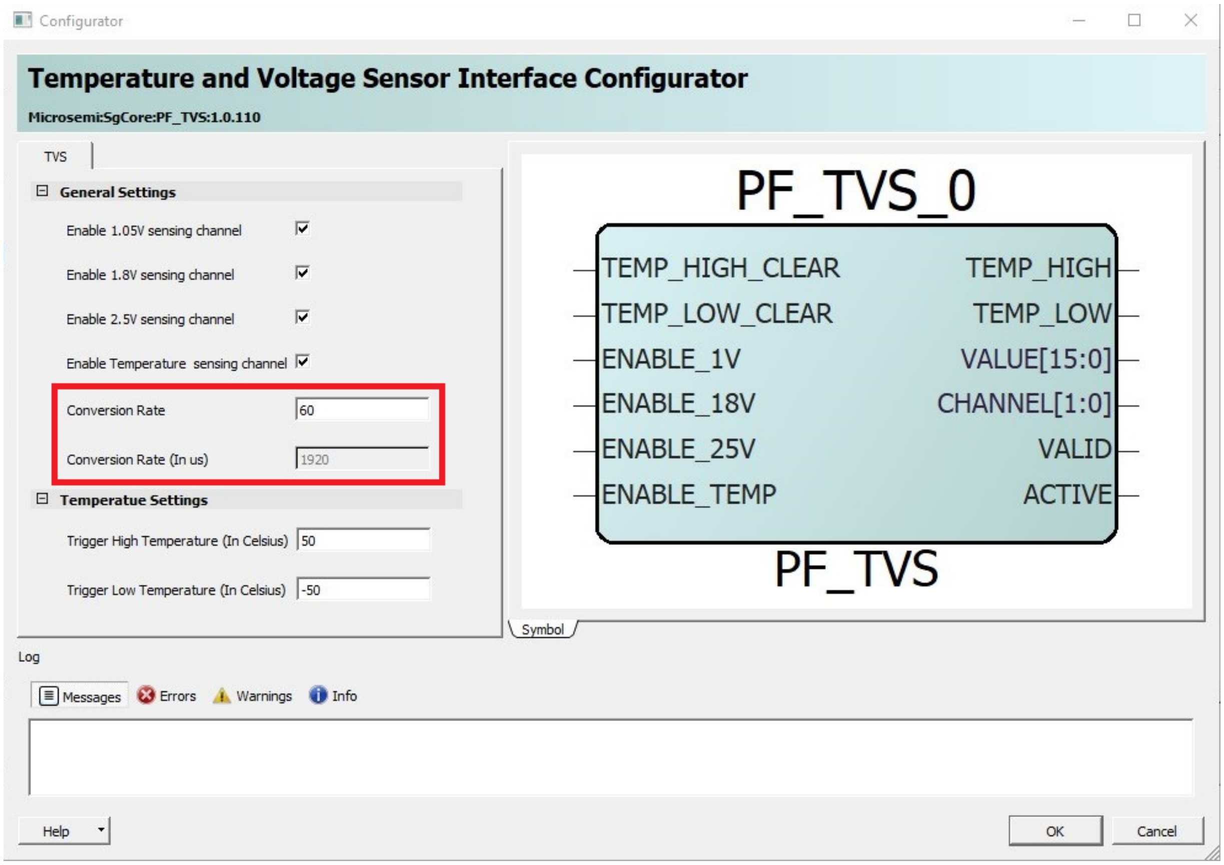
The top-level design includes the following components:

- TVS_IP_0 Macro
- Core_UART_0
- TVS_to_UART_0 logic
- clock_gen_0
- INIT_MONITOR_0 and PF_RESET_0

2.3.1.1 TVS_IP_0 Macro

The following figure shows the TVS interface configurator.

Figure 3 • **TVS Configurator**



Configurator

Temperature and Voltage Sensor Interface Configurator

Microsemi:SgCore:PF_TVS:1.0.110

TVS

General Settings

- Enable 1.05V sensing channel ☑
- Enable 1.8V sensing channel ☑
- Enable 2.5V sensing channel ☑
- Enable Temperature sensing channel ☑
- Conversion Rate: 60
- Conversion Rate (In us): 1920

Temperatue Settings

- Trigger High Temperature (In Celsius): 50
- Trigger Low Temperature (In Celsius): -50

The GUI displays the die temperature in degree Celsius by converting Kelvin values.

Celsius value = Kelvin value - 273.15

2.3.1.2 TVS_to_UART_0

The TVS to UART logic captures the Temperature and Voltage values from the TVS macro and sends the data to Core_UART_0.

2.3.1.3 clock_gen_0

CCC is configured to generate the 100 MHz clock.

2.4 Simulation Flow

The TVS simulation model updates the outputs of the TVS macro based on reading instructions given in the .mem file or .txt file. The file name must be passed to the simulation model for the TVS outputs to toggle. The parameter used to store the `.mem` file name is called "TVS_MEMFILE". Add the following vsim command to pass the file name.

```
-gTVS_MEMFILE="PATH_TO_FILE_RELATIVE_TO_SIMULATION_FOLDER"
```

.MEM File Format

The following format of the file is in hex:

```
<simulation time (τ) in hexadecimal>

<value of channel 0 at time τ>

<value of channel 1 at time τ>

<value of channel 2 at time τ>

<value of channel 3 at time τ>
```

The .mem file contains the simulation time followed by the digital values (16-bit) of the four ADC channels at that time instance. A value is required for the channel even if it is not used. The value can be 0. The simulation starts with all channel outputs being 0. The pattern can be repeated several times in the .mem file to reflect several values of the channel outputs. The content of the mem file is limited to 256 lines.

2.4.1 Simulating the Design

The Libero project includes a testbench to simulate the TVS block. The testbench captures all the four TVS channel values using CoreUART IP. The digital values for the four channels is passed through the .mem file.

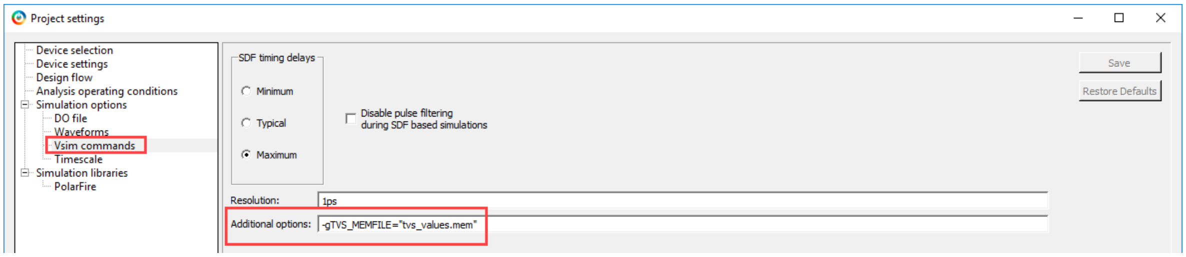
2.4.1.1 Simulation Settings

To pass the mem file for simulation, perform the following steps:

1. Open the Libero SoC project settings (**Project** > **Project Settings**).
2. Select **Vsim commands** under the **Simulation options**. Enter `-gTVS_MEMFILE="tvs_values.mem"` in the **Additional options** field and then click **Save**.

A sample tvs_values.mem is provided in the simulation folder. The .mem file must be available in simulation folder of the Libero project. The tvs_values.mem file captures the 16-bit digital output of the TVS block at different time instances.

Figure 4 • **Simulation Options—Vsim Command**



○ Minimum
○ Typical
● Maximum

□ Disable pulse filtering during SDF based simulations

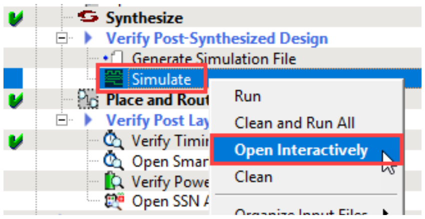
Resolution: 1ps

Additional options: -gTVS_MEMFILE="tvs_values.mem"

To simulate the design, perform the following steps:

1. In the **Design Flow** tab, right-click **Simulate** under **Verify Pre-Synthesis Design** and then select **Open Interactively**.

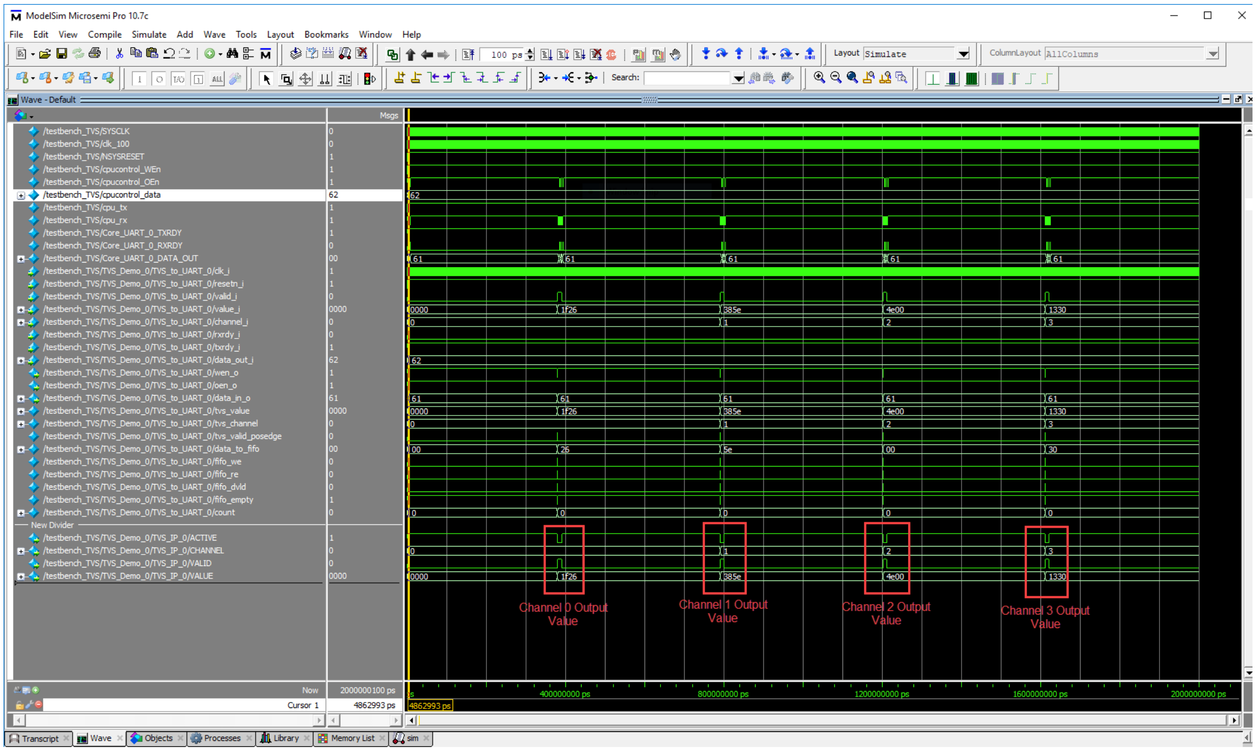
Figure 5 • **Design Flow—Simulate**



When simulation is completed, the Wave window appears as shown in the following figure. Since all the four channels are enabled, TVS circuit outputs value of the four channels at a given point in time on the VALUE output along with channel number on the CHANNEL output. The data present on the VALUE and CHANNEL outputs are valid only when the VALID output is asserted. Observe the following from the simulation results:

- After the channel is enabled for conversion, the TVS block takes 390 microseconds to complete the conversion.
- Each channel has a conversion delay of 410 microseconds.
- The conversion rate is equal to 1920 microseconds, which is same as the conversion rate set in the TVS configurator.
- TVS block generate the output values based on the values given in the tvs_values.mem file.

Figure 6 • **ModelSim Pro ME Wave Window**



2. Close ModelSim Pro ME and the Libero project.

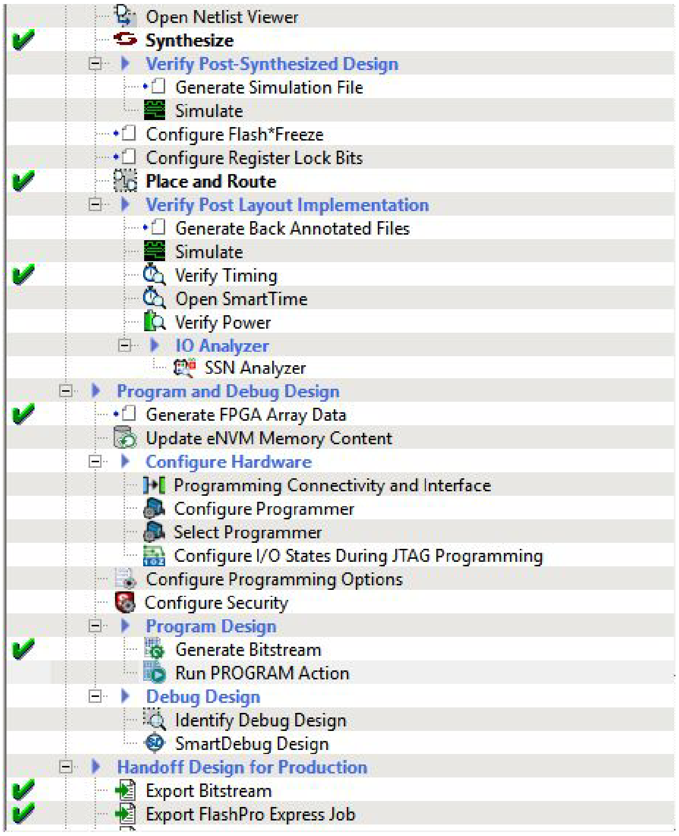
3 Libero Design Flow

This chapter describes the Libero design flow of the demo design. The Libero design flow involves the following steps:

- Synthesize
- Place and route
- Verify Timing
- Generate Bitstream
- Run PROGRAM Action

The following figure shows these options in the Design Flow tab.

Figure 7 • **Libero Design Flow Options**



- Open Netlist Viewer
- ✓ **Synthesize**
- **Verify Post-Synthesized Design**
  - Generate Simulation File
  - Simulate
- Configure Flash*Freeze
- Configure Register Lock Bits
- ✓ **Place and Route**
- **Verify Post Layout Implementation**
  - Generate Back Annotated Files
  - Simulate
  - ✓ Verify Timing
  - Open SmartTime
  - Verify Power
  - **IO Analyzer**
    - SSN Analyzer
- **Program and Debug Design**
  - ✓ Generate FPGA Array Data
  - Update eNVM Memory Content
  - **Configure Hardware**
    - Programming Connectivity and Interface
    - Configure Programmer
    - Select Programmer
    - Configure I/O States During JTAG Programming
  - Configure Programming Options
  - Configure Security
  - **Program Design**
    - ✓ Generate Bitstream
    - Run PROGRAM Action
  - **Debug Design**
    - Identify Debug Design
    - SmartDebug Design
- **Handoff Design for Production**
  - ✓ Export Bitstream
  - ✓ Export FlashPro Express Job

3.1 Synthesize

To synthesize the design, perform the following steps:

1. From the **Design Flow** window, double-click **Synthesize**.

When the synthesis is successful, a green tick mark appears as shown in Figure 7, page 8.

2. Right-click **Synthesize** and select **View Report** to view the synthesis report and log files in the Reports tab.

3.2 Place and Route

1. From the **Design Flow** window, double-click **Place and Route**.
   When place and route is successful, a green tick mark appears as shown in Figure 7, page 8.
2. Right-click **Place and Route** and select **View Report** to view the place and route report and log files in the Reports tab.

3.2.1 Resource Utilization

The following table lists the resource utilization of the design after place and route. These values may vary slightly for different Libero runs, settings, and seed values.

Table 4 • **Resource Utilization**

Type	Used	Total	Percentage
4LUT	269	299544	0.09
DFF	232	299544	0.08
I/O Register	0	510	0.00
Logic Elements	303	299544	0.10

3.3 Verify Timing

To verify timing, perform the following steps:

1. From the **Design Flow** window, double-click **Verify Timing**.
2. When the design successfully meets the timing requirements, a green tick mark appears as shown in Figure 7, page 8.
3. Right-click **Verify Timing** and select **View Report** to view the verify timing report and log files in the Reports tab.

3.4 Generate FPGA Array Data

To generate FPGA array data, double-click **Generate FPGA Array Data** from the **Design Flow** window.

A green tick mark is displayed after the successful generation of the FPGA array data as shown in Figure 7, page 8.

3.5 Generate Bitstream

To generate the bitstream, perform the following steps:

1. Double-click **Generate Bitstream** from the **Design Flow** tab.
   When the bitstream is successfully generated, a green tick mark appears as shown in Figure 7, page 8.
2. Right-click **Generate Bitstream** and select **View Report** to view the corresponding log file in the **Reports** tab.

3.6 Run PROGRAM Action

After generating the bitstream, the PolarFire device must be programmed. To program the PolarFire device, perform the following steps:

1. Ensure that the following Jumper Settings are set on the board.

Table 5 • **Jumper Settings**

Jumper	Description	Default
J18, J19, J20, J21, and J22	Short pin 2 and 3 for programming the PolarFire FPGA through FTDI	Closed
J28	Short pin 1 and 2 for programming through the on-board FlashPro5	Open
J26	Short pin 1 and 2 for programming through the FTDI SPI	Closed
J4	Short pin 1 and 2 for manual power switching using SW3	Closed
J12	Short pin 3 and 4 for 2.5 V	Closed

2. Connect the power supply cable to the **J9** connector on the board.
3. Connect the USB cable from the Host PC to **J5** (FTDI port) on the board.
4. Power on the board using the **SW3** slide switch.
5. Double-click **Run PROGRAM Action** from the **Libero > Design Flow** tab.

When the device is programmed successfully, a green tick mark appears as shown Figure 7, page 8.

4 Running the Demo

This chapter describes how to install and use the Graphic User Interface (GUI) to run the TVS demo. The PolarFire TVS demo application is a simple GUI that runs on the host PC to communicate with the PolarFire Device.

To install the GUI, perform the following steps:

1. Extract the contents of the `mpf_dg0852_df.rar` file. From the `mpf_dg0852_df\GUI\TVS_Monitor_GUI_Installer` folder, double-click the `setup.exe` file.
2. Follow the instructions displayed on the installation wizard.

After successful installation, TVS_Monitor_GUI appears on the **Start** menu of the host PC desktop.

To run the TVS demo, perform the following steps:

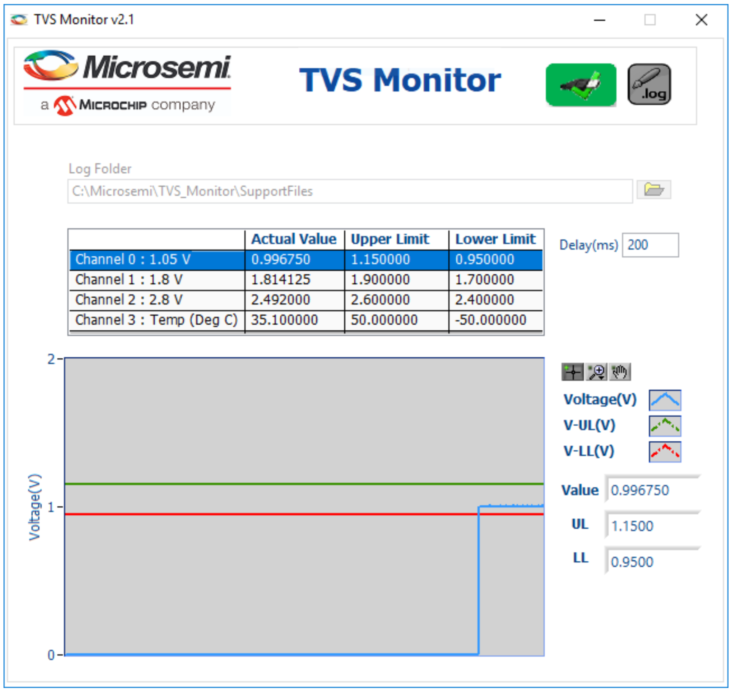
1. From the **Start** menu, click **TVS_Monitor_GUI** to launch the application. Ensure that the board is connected and appropriate Log Folder is selected.
2. Click **Connect**. On successful connection, the GUI shows the temperature and voltage values. Log file is created with time stamp in the file name at the Log Folder location.
   By default, Log Folder points to the 'SupportFiles' folder in the installation directory. User can modify the Log Folder location before connecting to the board.

Note: Ensure that the Log Folder is not a system restricted location. In this case, user is required to launch the GUI with admin privileges (right-click and **run as admin**).

3. Upper Limit, Lower Limit, and the minimum variation to log for each of the channels are configurable in the setup.ini file. Channel values are logged in the log file if there is a variation exceeding the specified 'min var' values in the setup.ini file.

The following figure shows the standard temperature and voltage values of channel 0 (1.05 V). The plot corresponds to the values of Channel 0. Similarly, select the other channels and view their corresponding values and plots.

Figure 8 • **Selecting COM Port and Connecting—Channel 0**



TVS Monitor v2.1

TVS Monitor

Log Folder
C:\Microsemi\TVS_Monitor\SupportFiles

	Actual Value	Upper Limit	Lower Limit
Channel 0 : 1.05 V	0.996750	1.150000	0.950000
Channel 1 : 1.8 V	1.814125	1.900000	1.700000
Channel 2 : 2.8 V	2.492000	2.600000	2.400000
Channel 3 : Temp (Deg C)	35.100000	50.000000	-50.000000

Delay(ms) 200

Value 0.996750
UL 1.1500
LL 0.9500

Note: The GUI updates the TVS channel values with the delay entered in the **Delay (ms)** field.

5 Appendix 1: Programming the Device Using FlashPro Express

This section describes how to program the PolarFire device with the .job programming file using FlashPro Express. The .job file is available at the following design files folder location:

```
mpf_dg0852_df\Programming_Job
```

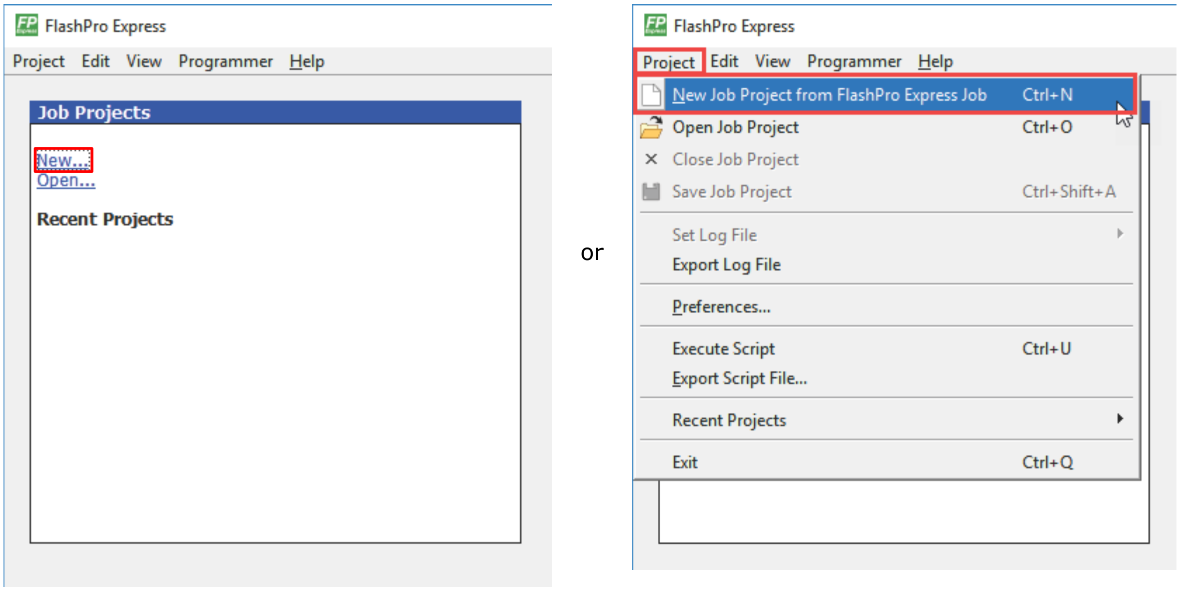
To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as listed in Table 5, page 10.

Note: The power supply switch must be switched off while making the jumper connections.

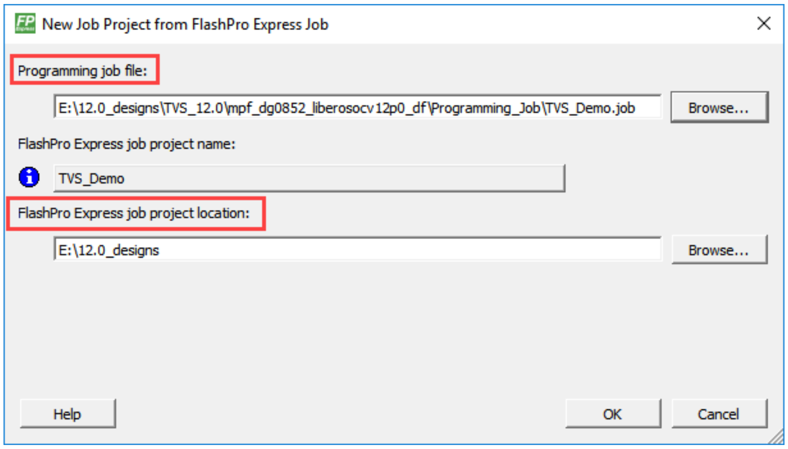
2. Connect the power supply cable to the **J9** connector on the board.
3. Connect the USB cable from the Host PC to the **J5** (FTDI port) on the board.
4. Power on the board using the **SW3** slide switch.
5. On the host PC, launch the **FlashPro Express** software.
6. Click **New** or select **New Job Project from FlashPro Express Job** from **Project** menu to create a new job project, as shown in the following figure.

Figure 9 • **FlashPro Express Job Project**



7. Enter the following in the **New Job Project from FlashPro Express Job** dialog box:
   - **Programming job file**: Click **Browse**, navigate to the location where the .job file is located, and select the file. The default location is: *<download_folder>\mpf_dg0852_df\Programming_Job*.
   - **FlashPro Express job project location**: Click **Browse** and navigate to the location where you want to save the project.

Figure 10 • **New Job Project from FlashPro Express Job**



Programming job file:

E:\12.0_designs\TVS_12.0\mpf_dg0852_liberosocv12p0_df\Programming_Job\TVS_Demo.job

FlashPro Express job project name:

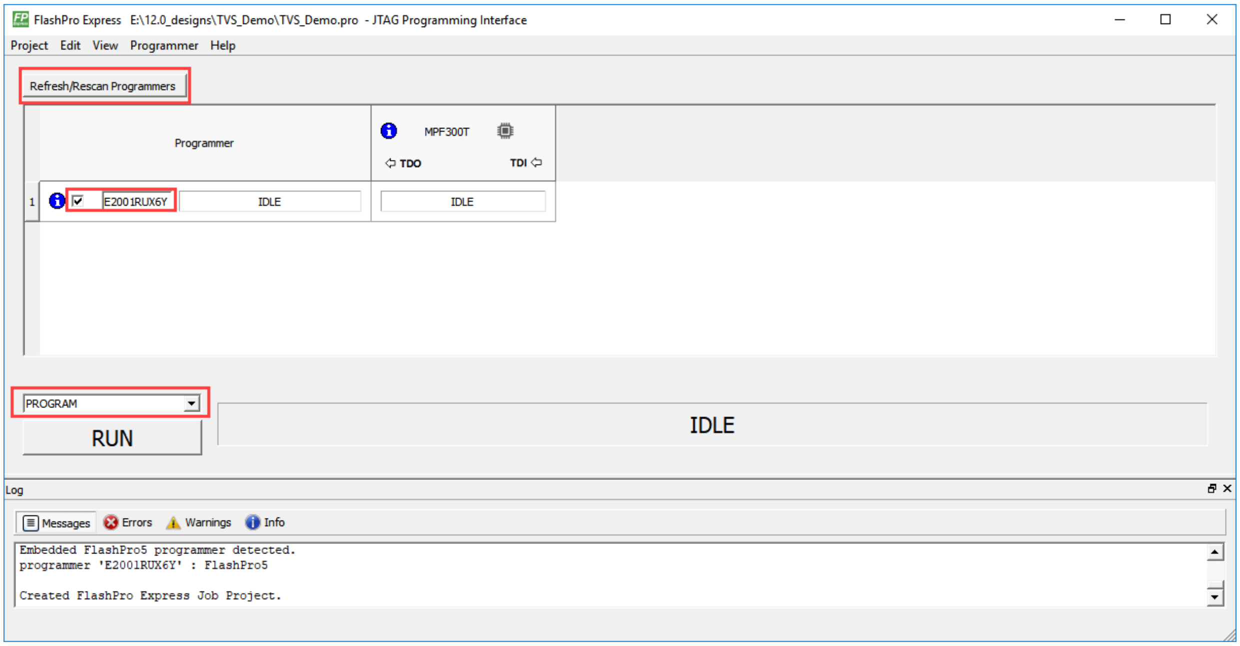
TVS_Demo

FlashPro Express job project location:

E:\12.0_designs

8. Click **OK**. The required programming file is selected and ready to be programmed in the device.
9. The FlashPro Express window appears as shown in the following figure. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan Programmers**.

Figure 11 • **Programming the Device**

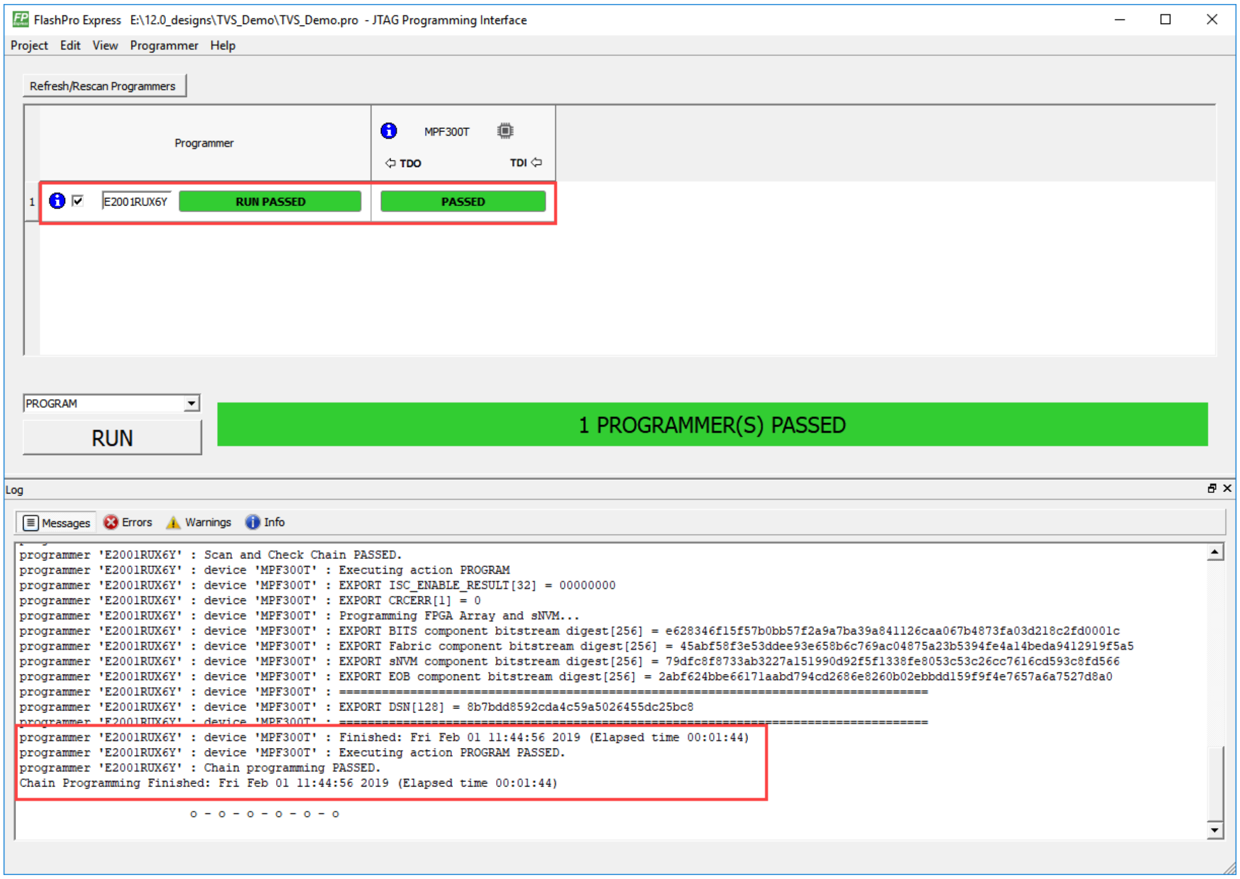

FlashPro Express E:\12.0_designs\TVS_Demo\TVS_Demo.pro - JTAG Programming Interface
Project Edit View Programmer Help
Refresh/Rescan Programmers
Programmer
MPF300T
TDO
TDI
1
E2001RUX6Y
IDLE
IDLE
PROGRAM
RUN
IDLE
Log
Messages
Errors
Warnings
Info

```
Embedded FlashPro5 programmer detected.
programmer 'E2001RUX6Y' : FlashPro5

Created FlashPro Express Job Project.
```

10. Click **RUN** to program the device. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in the following figure. See Running the Demo, page 11 to run the TVS demo.

Figure 12 • **FlashPro Express—RUN PASSED**



```
programmer 'E2001RUX6Y' : Scan and Check Chain PASSED.
programmer 'E2001RUX6Y' : device 'MPF300T' : Executing action PROGRAM
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT ISC_ENABLE_RESULT[32] = 00000000
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT CRCERR[1] = 0
programmer 'E2001RUX6Y' : device 'MPF300T' : Programming FPGA Array and sNVM...
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT BITS component bitstream digest[256] = e628346f15f57b0bb57f2a9a7ba39a841126caa067b4873fa03d218c2fd0001c
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT Fabric component bitstream digest[256] = 45abf58f3e53ddee93e658b6c769ac04875a23b5394fe4a14beda9412919f5a5
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT sNVM component bitstream digest[256] = 79dfc8f8733ab3227a151990d92f5f1338fe8053c53c26cc7616cd593c8fd566
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT EOB component bitstream digest[256] = 2abf624bbe66171aabd794cd2686e8260b02ebbdd159f9f4e7657a6a7527d8a0
programmer 'E2001RUX6Y' : device 'MPF300T' : =====================================================================================
programmer 'E2001RUX6Y' : device 'MPF300T' : EXPORT DSN[128] = 8b7bdd8592cda4c59a5026455dc25bc8
programmer 'E2001RUX6Y' : device 'MPF300T' : =====================================================================================
programmer 'E2001RUX6Y' : device 'MPF300T' : Finished: Fri Feb 01 11:44:56 2019 (Elapsed time 00:01:44)
programmer 'E2001RUX6Y' : device 'MPF300T' : Executing action PROGRAM PASSED.
programmer 'E2001RUX6Y' : Chain programming PASSED.
Chain Programming Finished: Fri Feb 01 11:44:56 2019 (Elapsed time 00:01:44)

                         o - o - o - o - o - o
```

11. Close **FlashPro Express** or in the **Project** tab, click **Exit**.

6 Appendix 2: Running the TCL Script

TCL scripts are provided in the design files folder under directory TCL_Scripts. If required, the design flow can be reproduced from Design Implementation till generation of job file.

To run the TCL, follow the steps below:

1. Launch the Libero software
2. Select **Project > Execute Script....**
3. Click Browse and select `script.tcl` from the downloaded TCL_Scripts directory.
4. Click **Run**.

After successful execution of TCL script, Libero project is created within TCL_Scripts directory.

For more information about TCL scripts, refer to **mpf_dg0852_df/TCL_Scripts/readme.txt.**

Refer to *Libero® SoC TCL Command Reference Guide* for more details on TCL commands. Contact Technical Support for any queries encountered when running the TCL script.