

# Programming the NVM of PMBus Buck Converters to Support Mass-Production Needs

Shawn Nie

## 1 Introduction

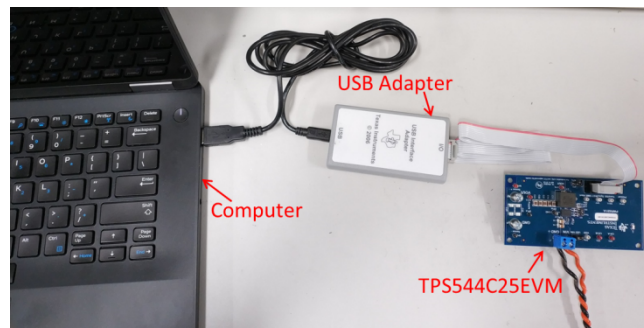
For mass production, it is necessary to select suitable software and program devices through simpler steps. This application report shows you how to program integrated PMBus converters with customized configurations to support mass production in various applications.

Some of the devices to which this guide applies include the following:

- [TPS544B25](#): A 4.5-V to 18-V VIN, 20-A SWIFT™ synchronous buck converter with PMBus and frequency synchronization.
- [TPS544C25](#): A 4.5-V to 18-V VIN, 20/30-A voltage-mode PMBus SWIFT™ step-down DC/DC converter with FSYNC.
- [TPS544B20](#): A 4.5-V to 18-V VIN, 20-A SWIFT synchronous buck converter with PMBus programmability and monitoring.
- [TPS544C20](#): A 4.5-V to 18-V VIN, 30-A SWIFT with PMBus programmability and voltage, current and temperature monitoring.
- [TPS546C20A](#): A 4.5-V to 18-V VIN, 35-A, 2x stackable synchronous buck converter with PMBus supporting telemetry.
- [TPS546C23](#): A 4.5-V to 18-V VIN, 35-A, 2x stackable synchronous buck converter with PMBus supporting telemetry.

## 2 Programming Hardware Requirements

The [Fusion Digital Power](#) (GUI) configures the devices. The application uses PMBus protocol to communicate with the device over a serial bus by using a TI USB adapter. The GUI launches into online mode if the devices are connected to a PC with a USB adapter as [Figure 1](#) shows. Otherwise, the GUI launches into offline mode.

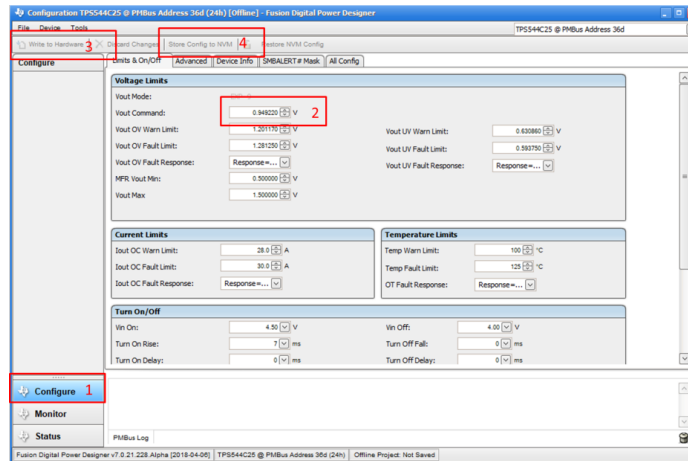


**Figure 1. Hardware Requirements**

To use any third party programmer or other I2C master to write device configurations to nonvolatile memory of above devices, a corresponding adapter that is compatible with a third party programmer must be used.



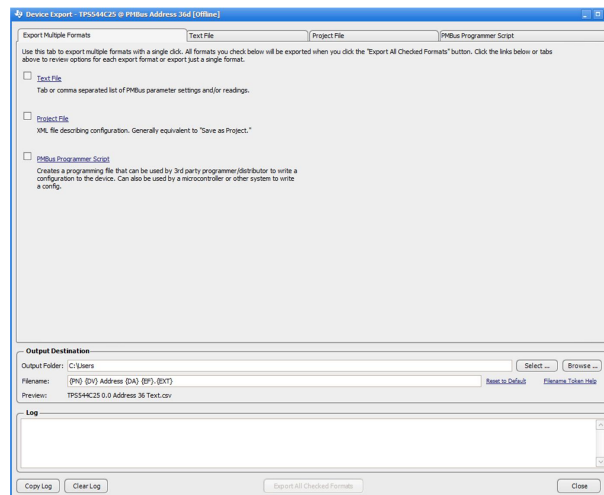
2. Adjust the voltage limits for the Vout Command.
3. Click Write to Hardware.
4. Click Store Configuration to NVM.
5. When the configuration is completed, save the configuration to the file.



**Figure 3. Labeled Steps to How to Configure A Device**

### 3.1.2 Exporting A File

To export a file, click on File, then Export. There are multiple export format options to choose from. To save a configuration, click in one or multiple boxes for the following options you want the file to be saved as: Text File, Project File, and PMBus Programmer Script.



**Figure 4. Device Export**

Figure 4 shows three different export file formats:

- **Text File (.csv):** This exports the device configuration in a text file that can be opened with Microsoft Excel or a text editor. You can personalize the format of the text file. Once the format of the text file is established, click Export Text File (shown in Figure 5) to save the text file.
  - **File Format:** Select CSV format to save the file as a Microsoft Excel spreadsheet. Select "Tab Separated" to save the file as a text file.
  - **Hex Format:** Three hex formats (0xAABB, AABB, AA-BB) can be selected.
  - **Parameters to Include:** Select "Read and Writable, Read Only, Writable Only" to decide which will

be saved.

- **Embedded Device Address:** Use “current device address” or enter a decimal number between 1 and 127 as the address.
- **Fields to Include:** Select the fields to be saved in every column.
  - **SAA Number (Dec):** The number of the USB adapters this device is attached to. This is useful if the user is exporting device data for two devices with the same address but on different adapters.
  - **Device Address:** (hexadecimal or decimal) The PMBus address the device is using (or was using at the time of project export in offline mode).
  - **Part Number:** The Texas Instruments' device part number.
  - **Command ID (with phase):** The same command ID in the advanced configure form of the GUI. This is a command ID with the phase number optionally appended. For example, IOUT\_CAL\_GAIN\_1 or READ-TEMPERATURE\_2MAX. Phase numbering starts at 1.
  - **Command ID:** A phase number is not appended to the Command ID. For example, there are multiple entries for IOUT\_CAL\_GAIN\_1 for each phase. The phase field determines which phase the record corresponds to . The phase field uses zero-based numbering.
  - **Command Code (Hex):** The PMBus or SMBus command code for the parameter. If the parameter is a meta command, the record does not correspond directly with a PMBus command that the device supports. In this case, this field is empty.
  - **Encoding:** The data type for this parameter. Select "Byte", "Word", or "Block".
  - **Decoded Value:** A textual summary of the value of the parameter. For numerical parameters such as VOUT\_COMMAND, this includes units. An example is “1.25 V”.
  - **PAGE (Hex):** The page that the parameter corresponds to. Page numbering starts at 0. If the parameter is global to the device and not PAGED, then the field is empty.
  - **PHASE (Hex):** The phase that the parameter corresponds to. Phase numbering starts at 0. If the parameter never varies when PHASE changes, the field is empty. Some parameters support a special PHASE = 0xFF, such as READ\_IOUT.
  - **Encoded Value (Hex):** This is raw data sent to and from the device when the device last read the parameter. This value is displayed in hex.

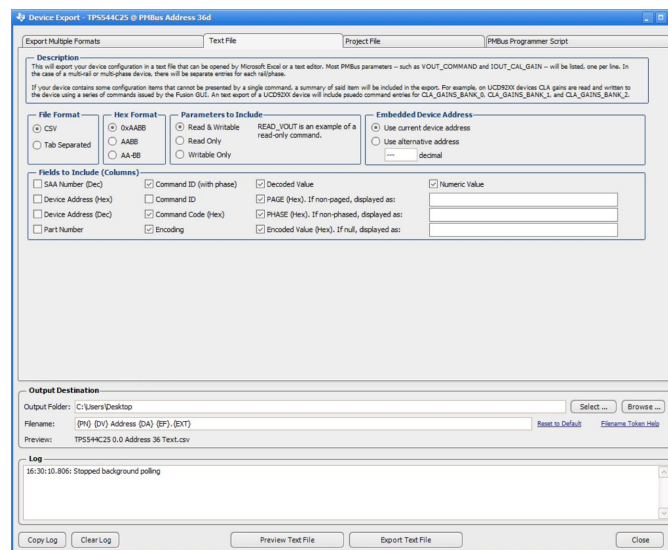


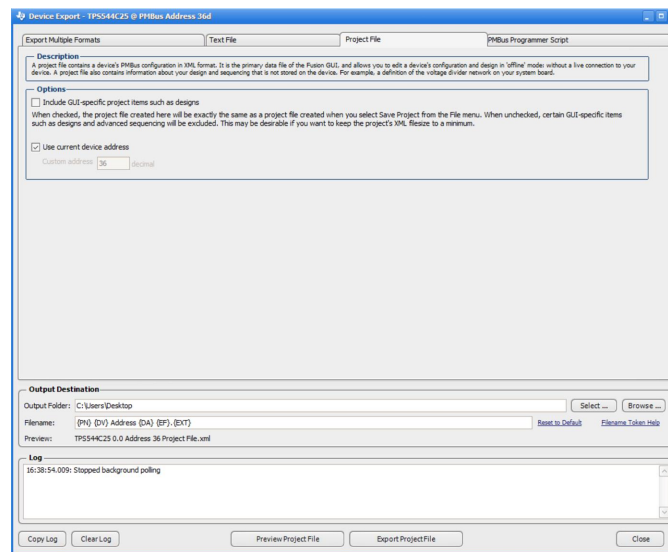
Figure 5. Exporting Text File

Figure 5 is a saved .csv file. Most PMBus parameters such as VOUT\_MODE and IOUT\_OC\_FAULT\_LIMIT are listed one parameter per line. In the case of a multirail or multiphase device, there are separate entries for each rail or phase. For example, “IOUT\_OC\_FAULT\_LIMIT” is the Command ID and 0x46 is the register address of this command. 0xF854 is the code data written to this register, which corresponds to 42A. The "Encoded Type" column lists the type of 0xF854.

	A	B	C	D	E	F
1	CmdIDWithPhase	CmdCodeHex	EncodedType	EncodedHex[HiByte LoByte]	Decoded	DecodedNumeric
2	VOUT_MODE	0x20	Byte	0x17	EXP -9	
3	CAPABILITY	0x19	Byte	0xB0	Max Bus: 400 Khz; PEC: Yes; SMBALERT#: Yes	
4	IC_DEVICE_ID	0xAD	Block	0x2346	0x2346 (TP5546C23)	
5	IC_DEVICE_REV	0xAE	Block	0x0100	0x0100	
6	IOUT_CAL_OFFSET	0x39	Word	0xE000	0.0000 A	0
7	IOUT_OC_FAULT_LIMIT	0x46	Word	0xF854	42.0 A	42
8	IOUT_OC_FAULT_RESPONSE	0x47	Byte	0xFF	Response=3,Restart=7,Delay=7	
9	IOUT_OC_WARN_LIMIT	0x4A	Word	0xF84A	37.0 A	37

**Figure 6. Text File**

- **Project File (.xml):** A project file contains the PMBus configuration of a device in XML format. It is the primary data file of the fusion GUI, and allows the user to edit the configuration of the device and design in offline mode without a live connection to the device. A project file contains information about the design and sequencing that is not stored on the device. For example, a definition of the voltage divider network on the system board. Click “Export Project File” to save the project file (see [Figure 7](#)).


**Figure 7. Exporting Project File**

When you check “Include GUI-specific project items such as designs”, the project file created here can be exactly the same as a project file created when you select “Save Project” from the “File” menu. When unchecked, GUI-specific items such as designs and advanced sequencing, can be excluded.

[Figure 8](#) is a part of project file content. The following numbers correspond to specific project information.

1. **Version:** XML version= “1.0”
2. **Timestamp:** The time the project file was created.
3. **Creator:** The software that created the project file.
4. **Part ID:** The part number
5. **Address:** The device address
6. **Is offline:** Whether it is online or offline
7. **Write Protected:** If the device is write protected, the data cannot be written.

```
<?xml version="1.0" encoding="UTF-8"?>
- <ProjectData xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Version>1</Version>
  <Timestamp>2018-02-15T16:12:25.9634432-06:00</Timestamp>
  <Creator>Texas Instruments Fusion Digital Power Designer v7.0.21.0 [2017-12-01]</Creator>
- <Devices>
  - <Device>
    <PartID>TPS544C25</PartID>
    <Address>36</Address>
    <Is_Offline>false</Is_Offline>
    <SAA_Number>1</SAA_Number>
    <Write_Protected>false</Write_Protected>
    <Package>0</Package>
    <Num_Outputs>1</Num_Outputs>
  - <Parameters>
    - <Parameter>
      <ID>VOUT_MODE</ID>
      <Code>32</Code>
      <IDAndCode>VOUT_MODE [0x20]</IDAndCode>
      <ValueText>EXP -9</ValueText>
      <ValueEncoded Hex="0x17" xsi:type="PMBusByte"/>
      <ParameterType>Custom</ParameterType>
      <ParameterCategory>Configuration</ParameterCategory>
      <Page>255</Page>
      <Phase xsi:nil="true"/>
      <CustomIndex xsi:nil="true"/>
    
```

Figure 8. Project File

- **Script File:** A setup of I2C, SMBus and PMBus commands that configure the device. These commands are shown in the script file. The formats are text (.txt) and Excel (.csv). Fusion GUI and third party programmers can use a script file. The file contains steps to program a single device using PMBus commands. Click “Export PMBus Programmer Script” to save the file as Figure 9 shows.

Figure 9 shows four options that can be selected when the script file is saved.

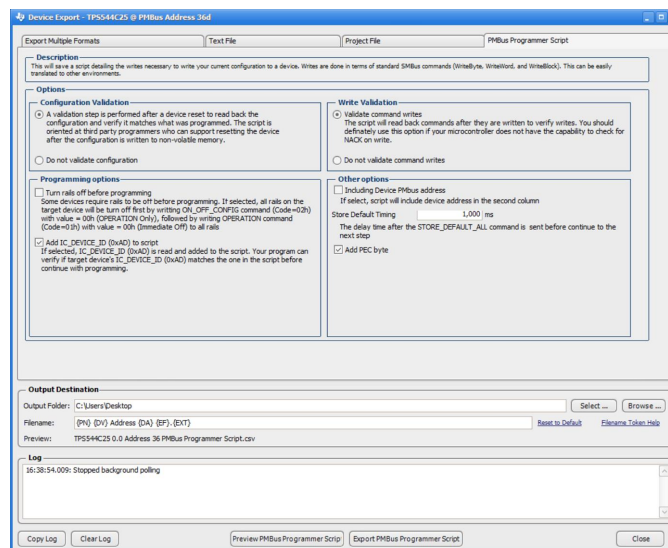


Figure 9. Exporting Script File

- **Configuration Validation:** A validation step is performed after a device reset to read back the configuration and verify that the configuration matches what was programmed. The script is designed for third party programmers who can support resetting the device after the configuration is written to nonvolatile memory.
- **Write Validation:** When checking the validate command write, the script reads back commands after they are written to verify writes. Use this option if the microcontroller cannot check NACK on write.

When checking the validate command write, the user can view the read back commands in the saved script file. The type of the saved code in the script file is changed from 0xABCD to 0xCDAB. Figure 10 shows the IOUT\_OC\_FAULT\_LIMIT command. The code of 42A is 0xF854, but the code is saved as 0x54F8.



54	Comment	Write IOUT_OC_FAULT_LIMIT 42.0 A	
55	WriteWord	0x46	0x54F8
56	ReadWord	0x46	0x54F8
57	Comment	Write IOUT_OC_WARN_LIMIT 37.0 A	
58	WriteWord	0x4A	0x4AF8
59	ReadWord	0x4A	0x4AF8

**Figure 10. Checking “Validate Command Write”**

If unchecked, the readback commands are not saved in the script file as [Figure 11](#) shows.

39	Comment	Write IOUT_OC_FAULT_LIMIT 42.0 A	
40	WriteWord	0x46	0x54F8
41	Comment	Write IOUT_OC_WARN_LIMIT 37.0 A	
42	WriteWord	0x4A	0x4AF8

**Figure 11. Unchecking “Validate Command Write”**

– **Programming Option:**

- Turn rails off before programming. Some devices require that the rails are turned off before programming. If the rails are selected, all rails on the target devices are turned off first by writing ON\_OFF\_CONFIG command (Code = 02h) with value = 00h (Operation Only), followed by writing OPERATION command (Code = 01h) with value = 01h (Immediate Off) to all rails.
- Add IC\_DEVICE\_ID (0xAD) to script. If selected, IC\_DEVICE\_ID (0xAD) is read and added to the script. The program can verify if the IC\_DEVICE\_ID (0xAD) of the target device matches the script before continuing with programming.

For example, [Figure 12](#) shows the IC\_DEVICE\_ID (0xAD) of the TPS544C25 device.

7	Comment	Verifying IC_DEVICE_ID is 0x2700 (TPS544C25)		
8	BlockRead	0xAD	0x022700	

**Figure 12. Verifying IC\_DEVICE\_ID**

– **Other Options:**

- **Including Device PMBus Address:** If this option is selected, the script includes the device address in the second column. The red 'w' text in [Figure 13](#) is the device address.

55	Comment	Write IOUT_OC_FAULT_LIMIT 42.0 A		
56	WriteWord	0x1B	0x46	0x54F898
57	ReadWord	0x1B	0x46	0x54F8
58	Comment	Write IOUT_OC_WARN_LIMIT 37.0 A		
59	WriteWord	0x1B	0x4A	0x4AF8E3
60	ReadWord	0x1B	0x4A	0x4AF8

**Figure 13. Including Device PMBus Address**

- **Store Default Timing:** The delay time after the STORE\_DEFAULT\_ALL command is sent before continuing to the next step.
- **Add PEC Byte:** The PEC byte can be added of the code as [Figure 14](#) shows.

64	Comment	Write OT_FAULT_LIMIT 145 C		
65	WriteWord	0x1B	0x4F	0x9100E8
66	ReadWord	0x1B	0x4F	0x9100
67	Comment	Write OT_WARN_LIMIT 120 C		
68	WriteWord	0x1B	0x51	0x780098
69	ReadWord	0x1B	0x51	0x7800

**Figure 14. Adding PEC Byte**

Figure 15 shows an example of a saved script file. Rows 1-6 show important information about script files such as formats, options, PEC, version, and time.

```

1 Comment Format=CSV; Hex=CoderUpper; BreakOutBytes=False; PEC=True; IncludeBlockLength=True; HasAddress=False [DO NOT REMOVE THIS LINE IF YOU WANT TO IMPORT USING A FUSION TOOL]
2 Comment PEC byte is calculated on all message bytes, including addresses and Read/Write bit.
3 Comment SMBus Fields are Request_Address,Command,Data with PEC byte
4 Comment For reads, the last field is what is expected back from the device
5 Comment Creator: Texas Instruments Fusion Digital Power Designer v7.0.21
6 Comment Date: Thursday, February 15, 2019
7 Comment Verifying IC_DEVICE_ID is 0x2700 (TP5544C25)
8 BlockRead 0xAD 0x022700
9 Comment Write configuration to volatile memory
10 Comment Write VIN_OFF 4.00 V: Minimum allowed value for VIN_OFF [0x36]
11 WriteWord 0x36 0x10F03E
12 ReadWord 0x36 0x10F0
13 Comment Write VIN_ON 4.50 V
14 WriteWord 0x35 0x12F0A9
15 ReadWord 0x35 0x12F0
16 Comment Write VIN_OFF 4.00 V
17 WriteWord 0x36 0x10F03E
18 ReadWord 0x36 0x10F0
19 Comment Write VOUT_SCALE_LOOP 1.00 : Maximum allowed value for VOUT_SCALE_LOOP [0x29]
20 WriteWord 0x29 0x04F0D8
21 ReadWord 0x29 0x04F0

```

- (1) Row 1 shows the format and options you have when the script file is saved.
- (2) Row 2 describes the PEC byte calculation.
- (3) Row 3 describes the SMBus Fields.
- (4) Row 4 is for reads.
- (5) Row 5 shows the software and version used to create the script file.
- (6) Row 6 records the time of when the script file is created.

Figure 15. Script File

The configuration can be saved as an individual device configuration in a project file (.xml), but also can be saved as a system file (.tifsp) by clicking “File -> Save System File” in the system window (shown in Figure 16).

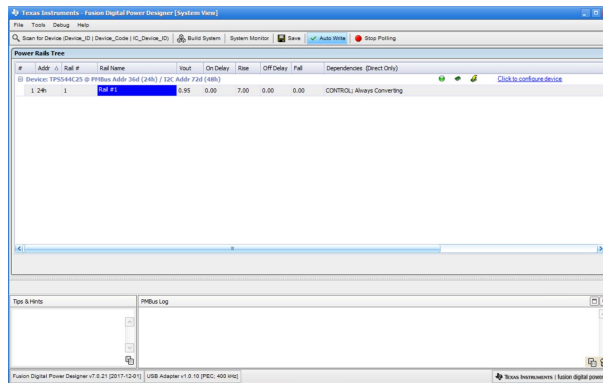


Figure 16. Export System File in System Window

There are some differences between system files and project files. For example, only a system file can be used with TI digital power manufacturing tools.

**System file (.tifsp):** stores multiple device configurations in a single file. System files can be saved in the system configuration window by using the Save button or File menu.

**Project file (.xml):** stores a single configuration definition of a device and non-device, such as rail names. Project files can be saved in the configuration window of the device by clicking File > Save Project As.

### 3.1.3 Importing A File

The system file (.tifsp) or the configuration file of the device saved in the previous step can be imported to the device.

To import the project file or script file, follow these steps:

1. Click File, then click Import to Device. Choose the preferred file type (as shown in Figure 17) and click the Next button to continue.



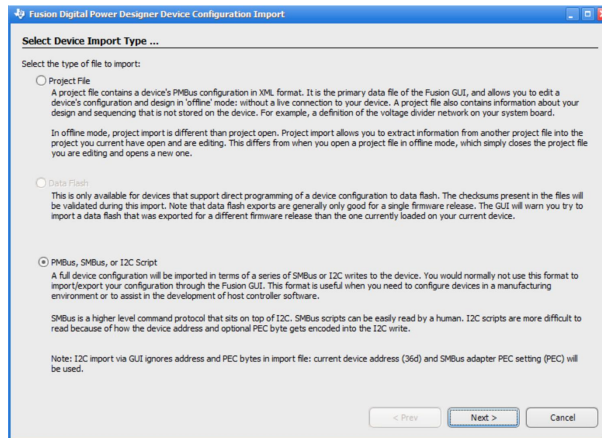


Figure 17. Selecting Device Import Type

2. Click Select File, and select the file you want to import. Click Next to continue (as shown in Figure 18).

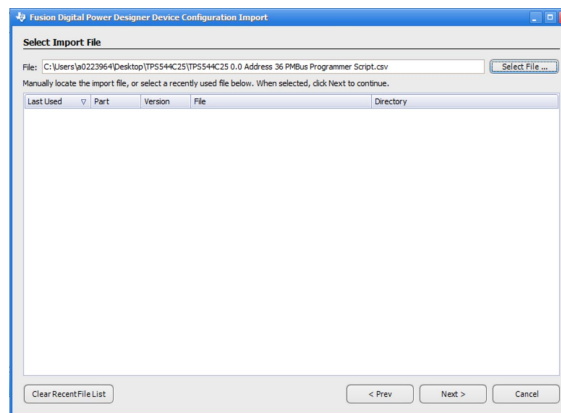


Figure 18. Selecting Import File

3. Set the address by clicking Use Address(es) found in script, and click the Next button (as shown in Figure 19).

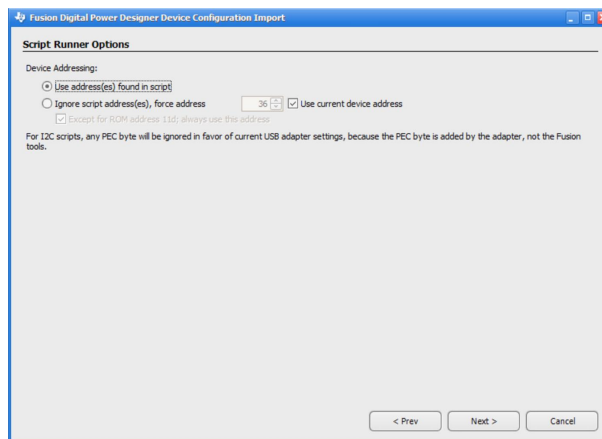


Figure 19. Script Runner Options

4. The Fusion GUI then begins to import the script. After the import is completed, the text **Script import 100% successful** appears and is highlighted in green (as shown in Figure 20)

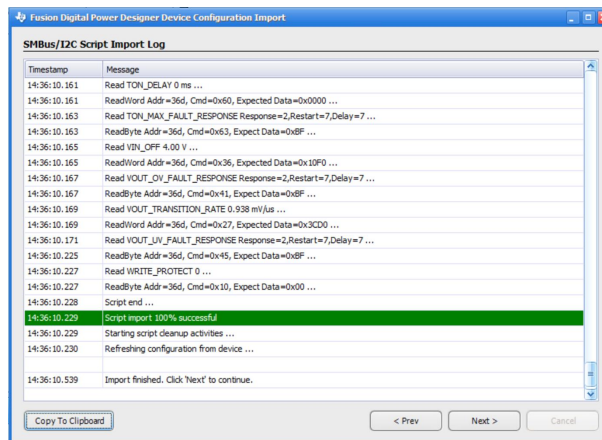


Figure 20. SMBus/I2C Script Import Log

**NOTE:** When there are errors in the script file during an import, the errors are indicated by the text **0x3CF833** (as shown in Figure 21), followed by an error parsing script message (as shown in Figure 22), indicating that there are errors in the script during a file import.

79	Comment	Write IOU_OC_FAULT_LIMIT 30.0 A		
80	WriteWor	0x46	0x3CF8	
81	ReadWorc	0x46	0x3CF8	
82	Comment	Write IOU_OC_WARN_LIMIT 28.0 A		
83	WriteWor	0x4A	0x38F89D	
84	ReadWorc	0x4A	0x38F8	

Figure 21. Script File With An Error

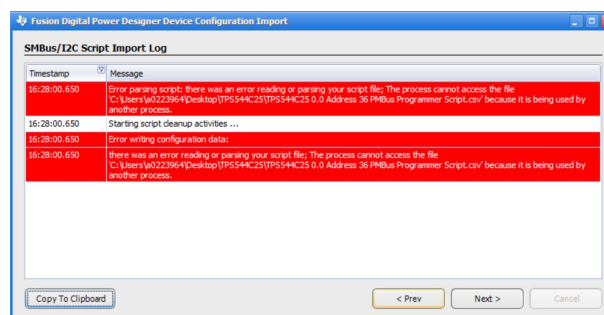
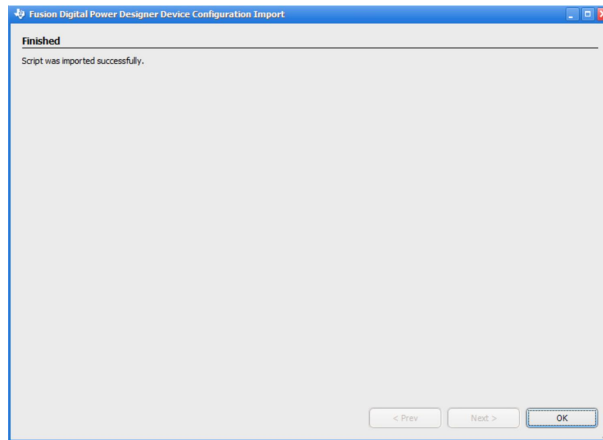


Figure 22. Reporting The Error

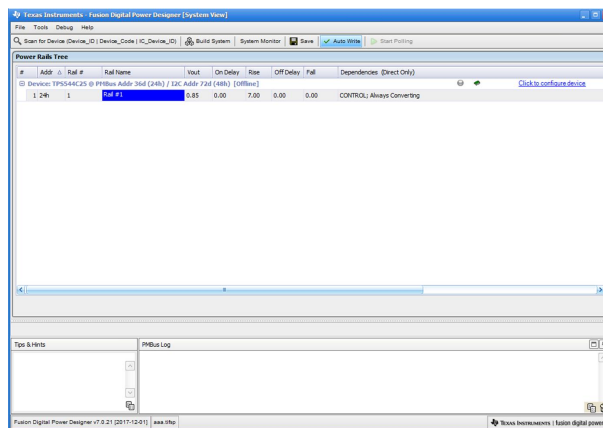
- Click the OK button to complete the importing (shown in Figure 23).



**Figure 23. Finishing Importing A Script File**

A system file can be imported to a device in the system window.

1. Click File then click Import System File to select the file you want to import (as shown in [Figure 24](#)).



**Figure 24. System Window**

2. The import dialog box appears after you select a file to import. You can review the parameters using the import dialog box by clicking on Show Update Parameters Only. Once you are done reviewing the updated parameters, click the Import button to import the system file to the device (as shown in [Figure 25](#)).

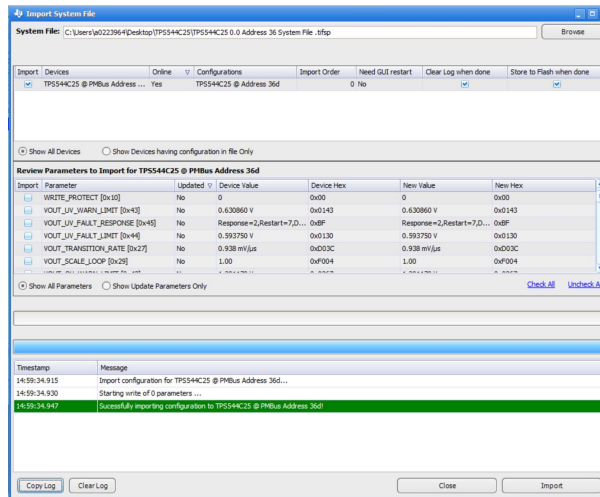


Figure 25. .Importing System File

### 3.2 Applications of Fusion Digital Power Manufacturing Tool

The Fusion Digital Power Manufacturing Tool (MFR GUI) is a Graphical User Interface used to download firmware, upgrade firmware, and to configure, calibrate, and test firmware. The MFR GUI supports the same devices as the Fusion Digital Power Designer software. This tool is typically used in manufacturing environments.

With the Fusion Manufacturing Tool, mass production is very convenient. After the device is connected to a PC, click the Scan button to find a device, then click the Start button so the system file can be loaded to the device.

#### 3.2.1 Import System File

Once the manufacturing tool is launched, three tabs appear within the Digital Power Manufacturing App window. The tabs are titled Edit Script, Load Script, and Run Script (as shown in Figure 26). Proceed with the following steps:

1. Click on the Load Script tab, and under that tab, click on the Browse button to find a script to load.
2. Select a script.
3. Click the Load button once the script is displayed.

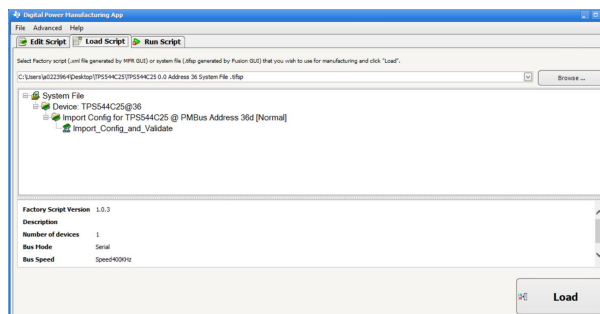


Figure 26. Loading Script

4. After the “Run Script” tab appears, click the Scan button to find the device (shown in Figure 27).

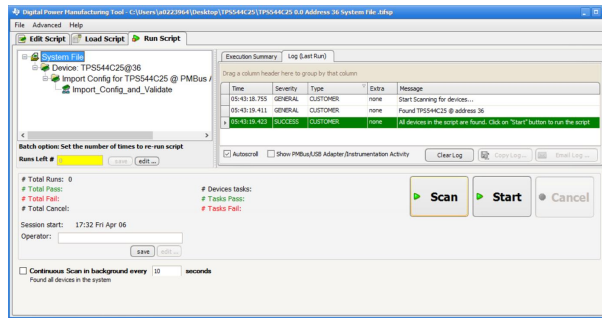


Figure 27. Running Script

5. After the scan is finished, click the Start button to run the script.

### 3.2.2 Creating A Script

Sometimes, you just want to change a few parameters, so there is no need to export a system file from the Fusion Digital Power GUI and load the script to the device using the manufacturing tool. It is more convenient to create script directly using the manufacturing tool.

In the Edit Script window shown in Figure 28, click on File, then click on New Script Wizard. When the New Script Wizard dialog box appears, you have the option of selecting some factory script templates:

- **Basic:** Contains the device and three generally used activities (initialization, normal and shutdown)
- **Blank:** Empty Device
- **Configure\_With\_Project\_File:** Configures a device based on a project file
- **Firmware\_Download\_via\_Rom:** Downloads firmware to a device in read-only memory mode and sends it to program mode. The activity type of downloading firmware to a blank device is the Initialization type.
- **Firmware\_Update:** Updates firmware on a device in program mode. The activity type is Normal, since it is in program mode.

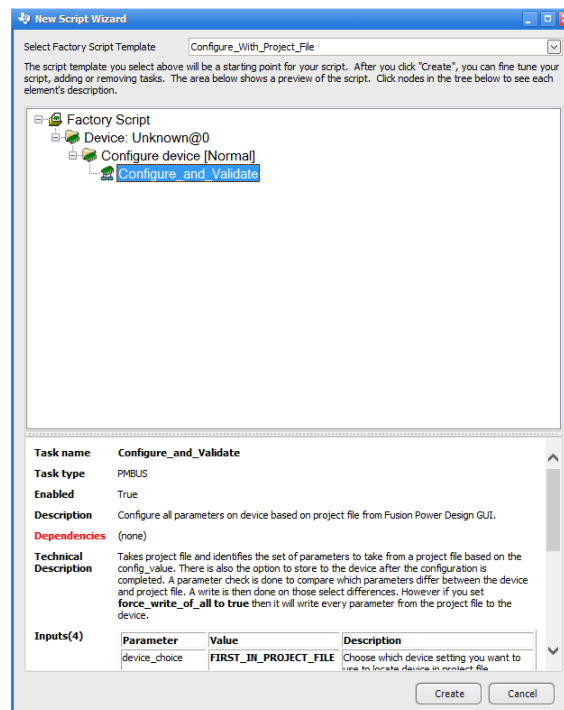
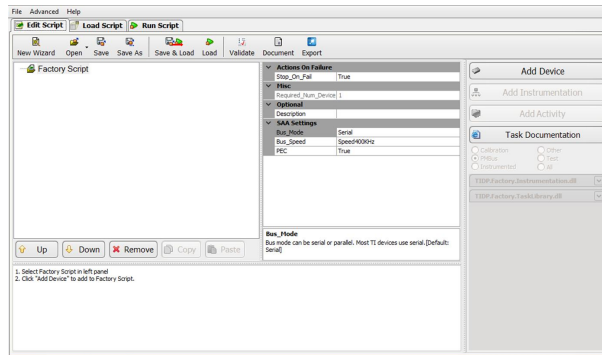


Figure 28. New Script Wizard

You can click nodes in the tree to see each element description. After selecting the factory script template, click Create to finish script creation.

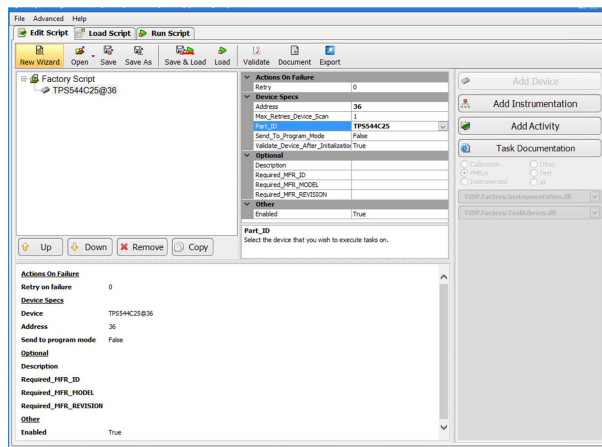
Alternatively, you can create the script directly from the Edit Script window instead of selecting the factory script template to Blank. For example, you can change the IOUT\_OC\_FAULT\_LIMIT from 30 A to 20 A by following these seven steps:

1. After launching the manufacturing tool, the Load Script screen appears, as shown in [Figure 29](#). To create a new script or edit an existing script, you must enable a special script editor mode. Click File in the top menu and then click Admin; enter the password “texas” A new Edit Script tab appears in the GUI.



**Figure 29. Load Script Screen**

2. Click Add Device and select Part\_ID TPS544C25. Then write the device address 36d (shown in [Figure 30](#)).



**Figure 30. Adding The Device**

3. Click Add Activity and change the activity name to IOUT\_OC\_FAULT\_LIMIT. Select activity type Normal (shown in [Figure 31](#)).



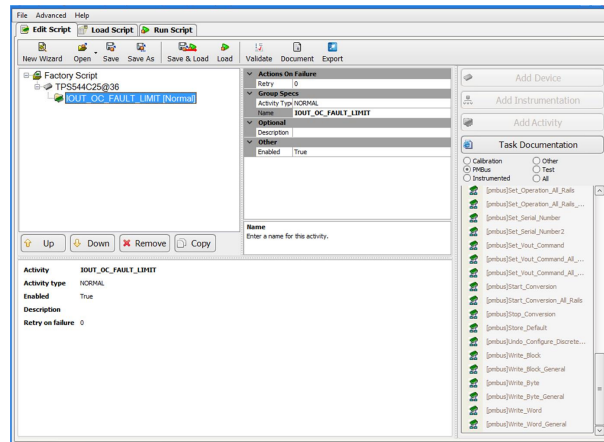


Figure 31. Adding An Activity

- To add a task to modify the value of IOUT\_OC\_FAULT\_LIMIT, click Write Word and write cmd\_byte=0x46, hex\_data=0xF828 (shown in Figure 32).

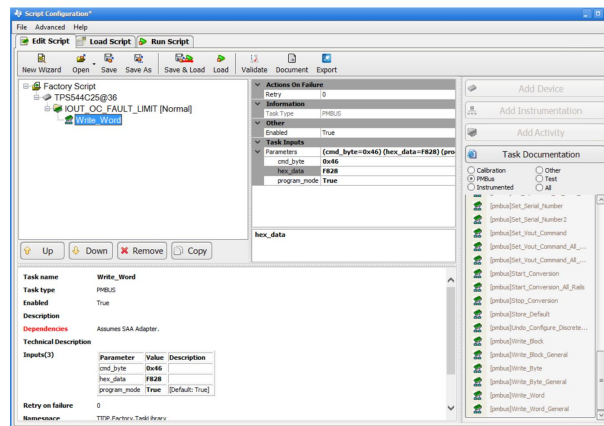


Figure 32. Adding A Task to Modify The Value of IOUT\_OC\_FAULT\_LIMIT

- To add a task to store all data to a non-volatile User Store memory, click Send Byte and write cmd\_byte=0x15 (shown in Figure 33).

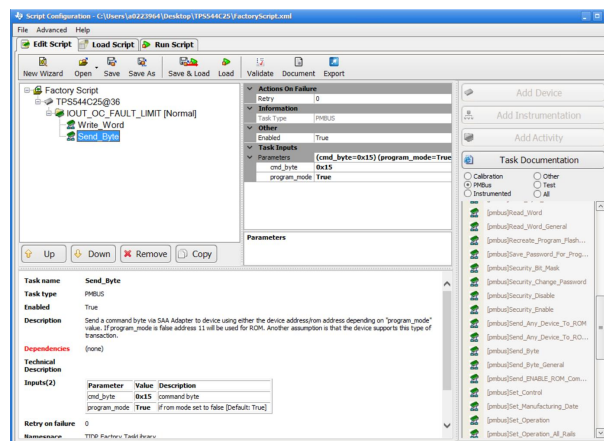


Figure 33. Adding A Task to Store All Data to NVM

- Click Save and Load to save the file and load it to the device (shown in Figure 34).

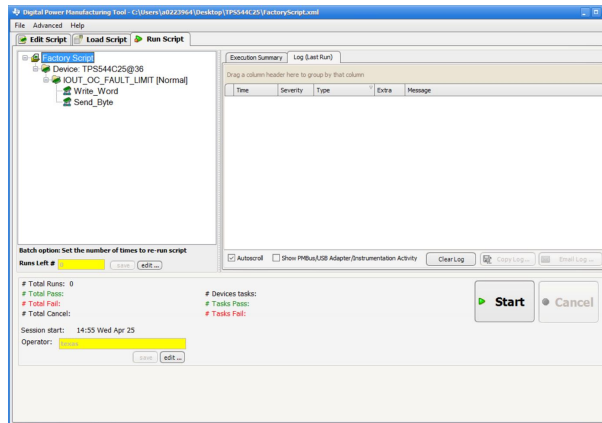


Figure 34. Saving and Loading Script

7. When the Run Script tab appears, click the Start button to begin script execution (shown in Figure 35).

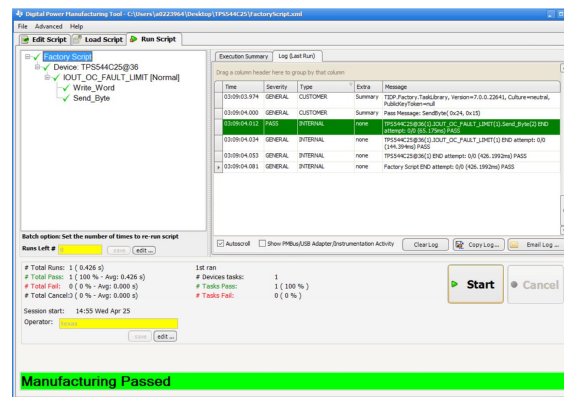


Figure 35. Running Script

## 4 Examples to Support Mass Production

The following examples show how to export a system file from the Fusion Digital Power GUI and load that file to the TPS544C25 using a manufacturing tool to support mass production. The purpose is to configure the IOUT\_OC\_WARN\_LIMIT to 28 A and IOUT\_OC\_FAULT\_LIMIT to 30 A.

### 4.1 Configuring The Device

1. After launching the Fusion Digital Power GUI, click the Configure button. Change the IOUT\_OC\_WARN\_LIMIT to 28 A and the IOUT\_OC\_FAULT\_LIMIT to 30 A, as shown in Figure 36.

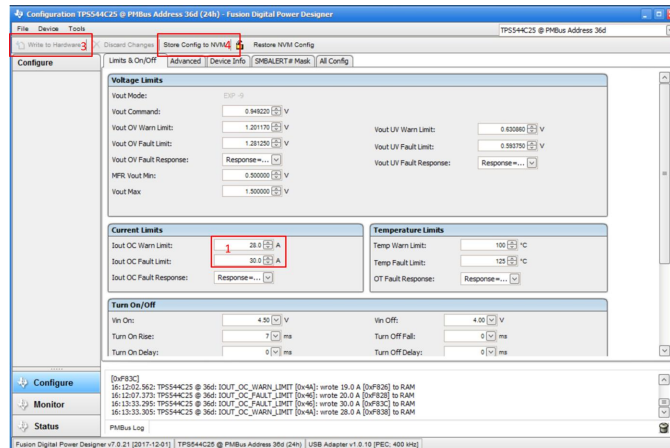


Figure 36. Configuring The Device

2. Click Write to Hardware.
3. Click Write Store Configuration to NVM.

## 4.2 Exporting A File

### 4.2.1 Exporting A Text File

1. Click File > Export > Text File and choose the file format (shown in Figure 37).

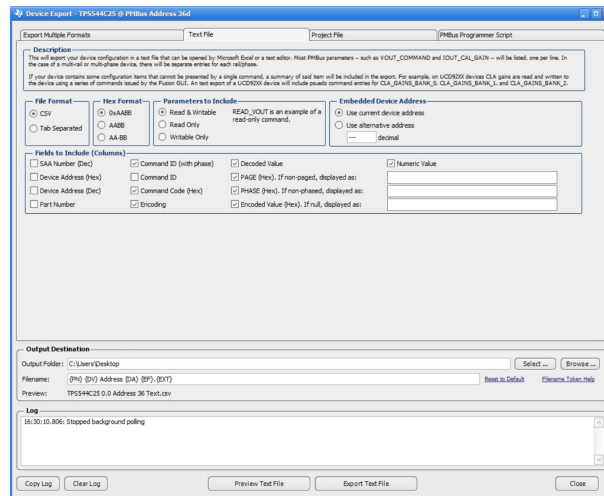


Figure 37. Exporting A Text File

2. Click Export Text File to create a file named TPS544C25 0.0 Address 36 Text.csv (shown in Figure 38).

CmdIDWithPhase	CmdCode	PageHexIfPaged	PhaseHex	EncodedType	EncodedHex(HiByte LoByte)	Decoded	DecodedNumeric
1 CmdIDWithPhase							
2 VOUT_MODE	0x20			Byte	0x17	EXP -9	
3 CAPABILITY	0x19			Byte	0x80	Max Bus: 400 kHz; PEC: Yes; SMBALERT: Yes	
4 IC_DEVICE_ID	0x4D			Block	0x2700	0x2700 (TPS544C25)	
5 IC_DEVICE_REV	0x4E			Block	0x0000	0x0000	
6 IOUT_CAL_OFFSET	0x39			Word	0x0000	0.0000 A	0
7 IOUT_OC_FAULT_LIMIT	0x46			Word	0xF83C	30.0 A	30
8 IOUT_OC_FAULT_RESPONSE	0x47			Byte	0x8F	Response=2,Restart7,Delay=7	
9 IOUT_OC_WARN_LIMIT	0x4A			Word	0xF838	28.0 A	28
10 MFR_06 (FOR USER)	0x40			Word	0x0000	0000000000000000	
11 MFR_23 (OPTIOXN)	0x43			Word	0x00C7	PM6_VTH1.55 DTT EN3 DR AC CNTL3 DTC CI	

Figure 38. Text file

### 4.2.2 Exporting A Project File

1. Click File > Export > Project File and choose the file format (shown in Figure 39).

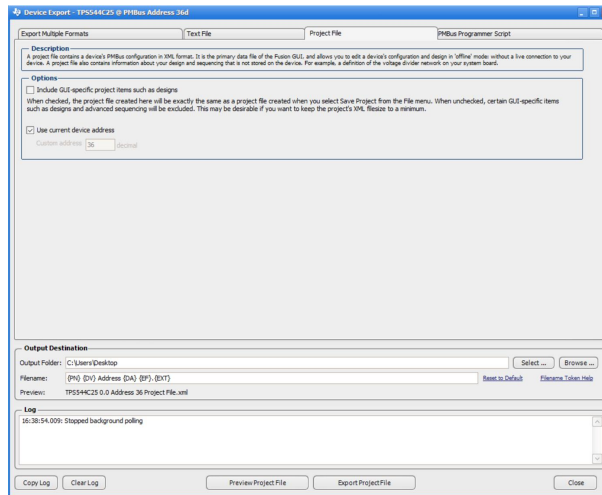


Figure 39. Exporting A Project File

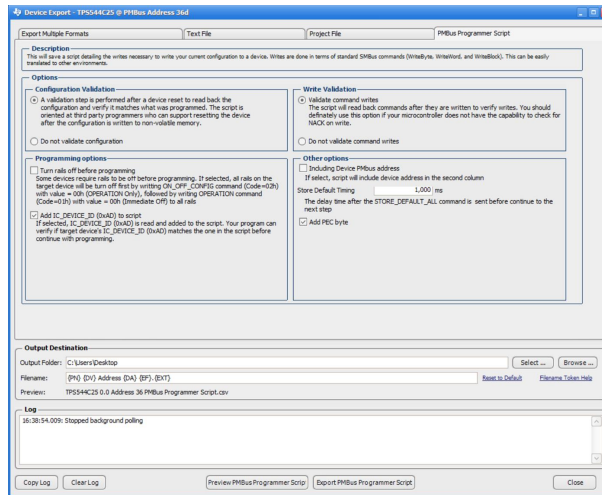
- Click Export Project File to create a file named TPS544C25 0.0 Address 36 Project File.xml (shown in Figure 40).

```
<?xml version="1.0" encoding="UTF-8"?>
- <ProjectData xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Version>1</Version>
  <Timestamp>2018-04-06T16:46:18.8173725-05:00</Timestamp>
  <Creator>Texas Instruments Fusion Digital Power Designer v7.0.21.0 [2017-12-01]</Creator>
- <Devices>
  - <Device>
    <PartID>TPS544C25</PartID>
    <Address>36</Address>
    <Is_Offline>false</Is_Offline>
    <SAA_Number>1</SAA_Number>
    <Write_Protected>false</Write_Protected>
    <Package>0</Package>
    <Num_Outputs>1</Num_Outputs>
  - <Parameters>
    - <Parameter>
      <ID>VOUT_MODE</ID>
      <Code>32</Code>
      <IDAndCode>VOUT_MODE [0x20]</IDAndCode>
      <ValueText>EXP -9</ValueText>
      <ValueEncoded Hex="0x17" xsi:type="PMBusByte"/>
      <ParameterType>Custom</ParameterType>
      <ParameterCategory>Configuration</ParameterCategory>
      <Page>255</Page>
      <Phase xsi:nil="true"/>
      <CustomIndex xsi:nil="true"/>
    - <Parameter>
      <ID>IOUT_OC_FAULT_LIMIT</ID>
      <Code>70</Code>
      <IDAndCode>IOUT_OC_FAULT_LIMIT [0x46]</IDAndCode>
      <ValueText>30.0 A</ValueText>
      <ValueEncoded Hex="0xF83C" xsi:type="PMBusWord"/>
      <ValueNumeric>30</ValueNumeric>
      <ParameterType>Current</ParameterType>
      <ParameterCategory>Limits</ParameterCategory>
      <Page>255</Page>
      <Phase xsi:nil="true"/>
      <CustomIndex xsi:nil="true"/>
    - <Parameter>
      <ID>IOUT_OC_WARN_LIMIT</ID>
      <Code>74</Code>
      <IDAndCode>IOUT_OC_WARN_LIMIT [0x4A]</IDAndCode>
      <ValueText>28.0 A</ValueText>
      <ValueEncoded Hex="0xF83B" xsi:type="PMBusWord"/>
      <ValueNumeric>28</ValueNumeric>
      <ParameterType>Current</ParameterType>
      <ParameterCategory>Limits</ParameterCategory>
      <Page>255</Page>
      <Phase xsi:nil="true"/>
      <CustomIndex xsi:nil="true"/>
    </Parameter>
  </Devices>
</ProjectData>
```

Figure 40. Project File

#### 4.2.3 Exporting A Script File

- Click File > Export > PMBus Programmer Script and choose the file format (shown in Figure 41).



**Figure 41. Exporting A Script File**

2. Click Export PMBus Programmer Script to create a script file named TPS544C25 0.0 Address 36 PMBus Programmer Script.csv (shown in Figure 42).

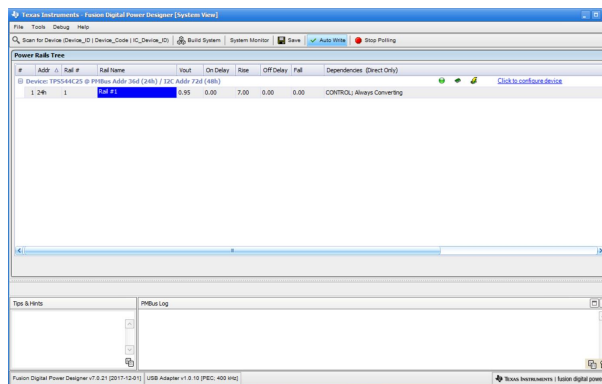
```

1 Comment Format=CSV; Hex=CoderUpper; BreakOutBytes=False; PEC=True; IncludeBlockLength=True; HasAddress=False [DO NOT REMOVE THIS LINE IF YOU WANT TO IMPORT USING A FUSION TOOL]
2 Comment PEC byte is calculated on all message bytes, including addresses and Read/Write bit.
3 Comment SMBus Fields are Request_Address,Command,Data with PEC byte
4 Comment For reads, the last field is what is expected back from the device
5 Comment Creator: Texas Instruments Fusion Digital Power Designer v7.0.21
6 Comment Date: Friday, April 6, 2018
7 Comment Verifying IC_DEVICE_ID is 0x2700 (TPS544C25)
8 BlockRead 0xAD 0x022700
9 Comment Write configuration to volatile memory
10 Comment Write VIN_OFF 4.00 V: Minimum allowed value for VIN_OFF [0x36]
11 WriteWord 0x36 0x10F03E
12 ReadWord 0x36 0x10F0
13 Comment Write VIN_ON 4.50 V
14 WriteWord 0x35 0x12F0A9
15 ReadWord 0x35 0x12F0
179 Comment Write IOU_T_OC_FAULT_LIMIT 30.0 A
80 WriteWord 0x46 0x3CF833
81 ReadWord 0x46 0x3CF8
82 Comment Write IOU_T_OC_WARN_LIMIT 28.0 A
83 WriteWord 0x4A 0x38F89D
84 ReadWord 0x4A 0x38F8
  
```

**Figure 42. Script File**

#### 4.2.4 Exporting A System File

In the system window shown in Figure 43, click File > Save System File.



**Figure 43. Exporting A System File In System Window**

### 4.3 Loading A System File

After launching the manufacturing tool, the Edit Script tab appears, as shown in Figure 44.

1. Click Load Script. The Load Script tab appears.
2. Click Browse and select the system file TPS544C25 0.0 Address 36 System File.tifsp. After the script is displayed in the Load Script tab, click Load.

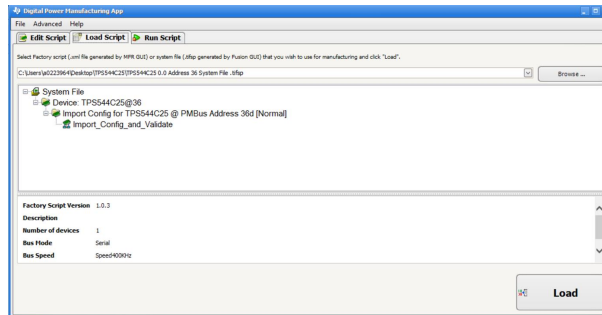


Figure 44. Loading A Script

3. After the Run Script tab appears, click the Scan button (shown in Figure 45).

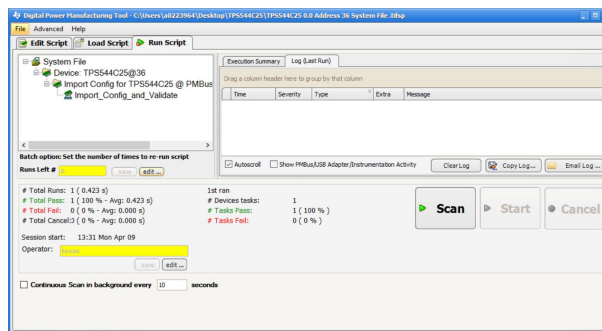


Figure 45. Scanning For Devices

4. After the scan is finished, click the Start button to run the script (shown in Figure 46).

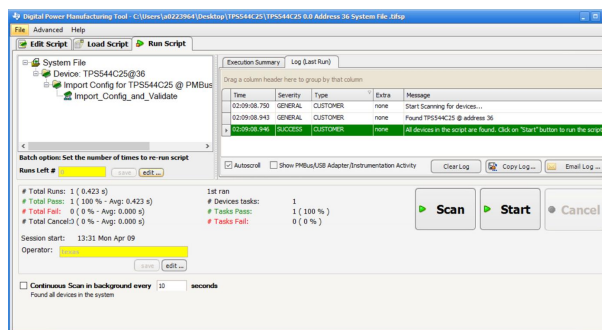
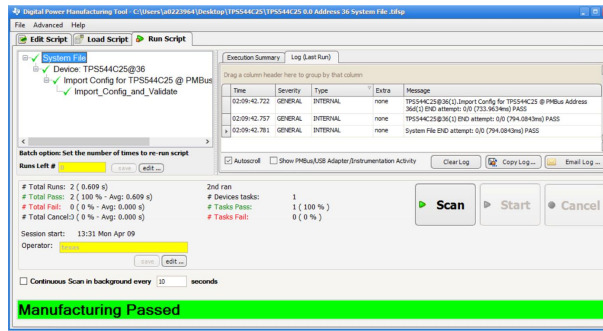


Figure 46. Running The Script

5. When the script is finished running, you see Manufacturing Passed on the bottom of the screen (shown in Figure 47).





**Figure 47. Manufacturing Passed Alert**

**5 References**

1. Fusion Digital Power Manufacturing GUI User’s Guide, Script Runner, February 2014.
2. Fusion Digital Power Manufacturing Tool User’s Guide, Script Writer, April 2010.

**Revision History**

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

**Changes from A Revision (March 2019) to B Revision**

**Page**

- Replaced figure 2..... 2

**Changes from Original (May 2018) to A Revision**

**Page**

- Edited application report for clarity..... 1
- Changed images to be more uniform in size. .... 1

## IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale ([www.ti.com/legal/termsofsale.html](http://www.ti.com/legal/termsofsale.html)) or other applicable terms available either on [ti.com](http://ti.com) or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2019, Texas Instruments Incorporated