AN 035: SPI Passive Programming with Raspberry Pi

AN035-v1.2
December 2022
www.efinixinc.com

Contents

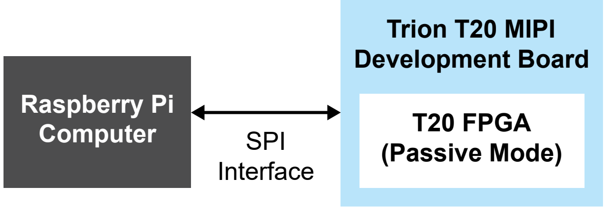
Introduction

One of the methods to configure a Trion® FPGA is using the SPI passive configuration mode. In SPI passive mode, an external microprocessor or microcontroller sends the bitstream to the Trion® FPGA using the SPI interface.

A Raspberry Pi computer can act as a microcontroller in the passive mode configuration mode. Efinix provides example designs that demonstrate a Raspberry Pi computer configuring the Trion® development board using passive SPI mode. This example design includes C and Python Raspberry Pi software examples.

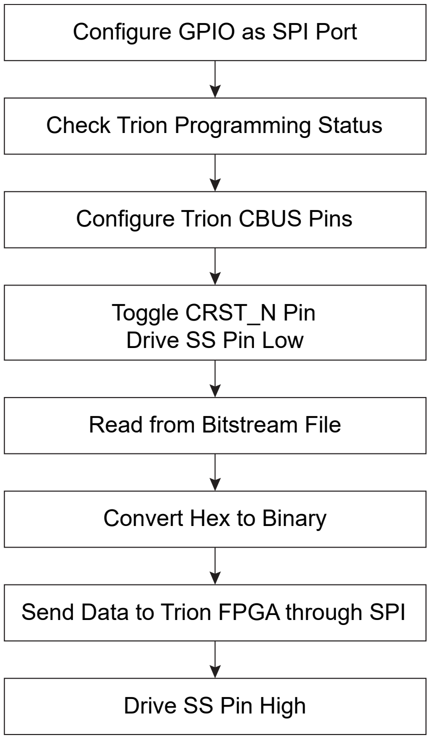
The software examples provided support up to x32 SPI passive mode, but the example targets the Trion® T20 MIPI Development Board that only supports up to x4 mode. You can use SPI passive modes of up to x32 when using the software examples with boards that support higher SPI passive modes. See Extending the Example to Your Own Board on page 7 for more information on extending the example to your own board.

Figure 1: SPI Passive Configuration using Raspberry Pi Block Diagram



The following flow chart describes the Raspberry Pi software example flow.

Figure 2: Raspberry Pi Software Flow


Configure GPIO as SPI Port
Check Trion Programming Status
Configure Trion CBUS Pins
Toggle CRST_N Pin
Drive SS Pin Low
Read from Bitstream File
Convert Hex to Binary
Send Data to Trion FPGA through SPI
Drive SS Pin High

Required Hardware and Software

- Raspberry Pi computer:
  - Raspberry Pi 3 or 4 running Raspbian v9.13 or later
  - Display and input device to view and run commands
  - `wiringpi` Python library installed[1]
  - Your Trion® bitstream file, **.hex**, stored in the Raspberry Pi
- Trion® T20 MIPI Development Board
- Jumper cables depending on the SPI passive mode used

Note: Refer to Extending the Example to Your Own Board on page 7 for more information about using the example design with other Trion® boards.

Example Design Files

There are two Raspberry Pi software examples included in the example design:

- *raspberry_pi_programmer.py*—Raspberry Pi software using Python
- *c_programmer*—Raspberry Pi software using C

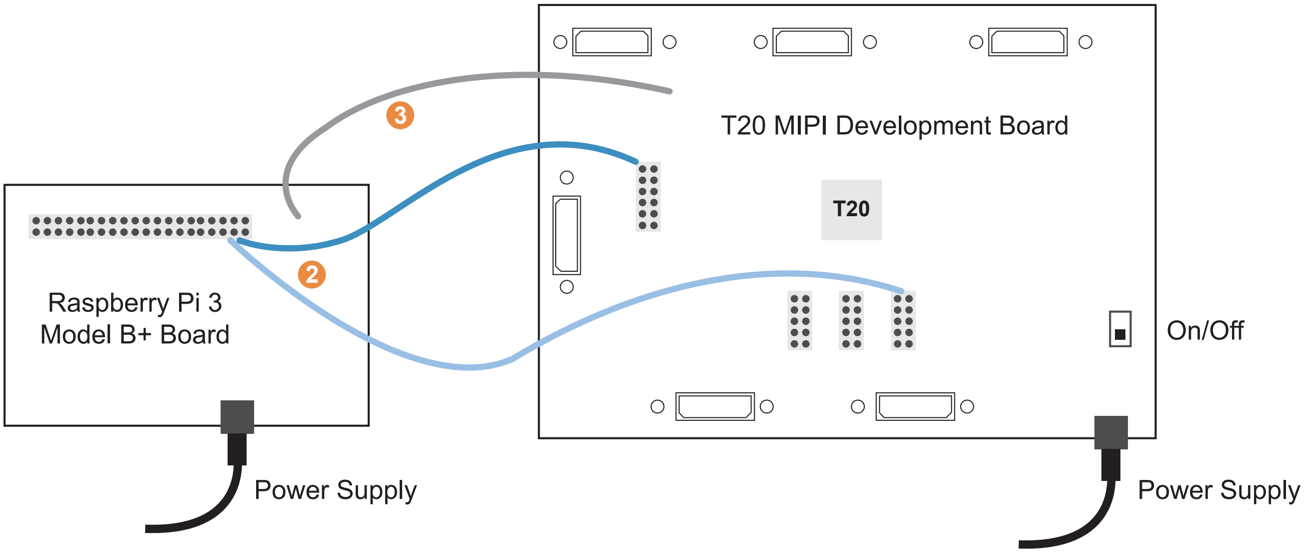
Note: This example design does not include a bitstream file. You have to generate a bitstream file to configure the Trion® FPGA.

[1] Refer to https://github.com/WiringPi/WiringPi-Python for more information.

Set Up the Hardware

The following figure shows the hardware setup steps:

Figure 3: Hardware Setup



1. Attach standoffs to the board if you have not already done so.
2. Connect the pins of the Raspberry Pi board to the Trion® T20 MIPI Development Board using jumpers as shown in **Trion T20 MIPI Development Board SPI Interface Pin Mapping** on page 6.
3. Connect any of the ground pin between the boards.
4. Connect the Trion® T20 MIPI Development Board's and Raspberry Pi board's power supply and turn on the boards.

Trion® T20 MIPI Development Board SPI Interface Pin Mapping

The following tables list the Raspberry Pi board GPIO pins and Trion® T20 MIPI Development Board pins required for the jumper connections.

Note: Refer to SPI Interface Pin Mapping on page 8 for the generic SPI interface pin mapping to be used with other Trion® boards.

Table 1: Trion® T20 MIPI Development Board SPI Interface Pin Mapping

Raspberry Pi Board Pin Name	Signal Name	Trion® T20 MIPI Development Board	
		Header Name	Pin Name
GPIO24	SCK	J5	SPI_CLK
GPIO25	SS	J5	SPI_SS
GPIO0	CDATA0	J5	SPI_MOSI
GPIO1	CDATA1	J5	SPI_MISO
GPIO2	CDATA2	J8	GPIOL_12_3V3
GPIO3	CDATA3	J8	GPIOL_13_3V3

Important: In addition to the SPI interface signals, you also must connect a jumper between the ground pins of the boards.

Run the Python Software Example Design

1. Run the following command to install the `wiringpi` library on the Raspberry Pi board:
   `pip install wiringpi`
2. Run the following command to start the configuration:

```
python raspberry_pi_programmer.py <input_file> --width=<programming_width>
```

- <*input_file*> is the bitstream file path to configure the board
- <*programming_width*> is the SPI passive configuration width

Run the C Software Example Design

1. Run the following command to install the `wiringpi` library on the Raspberry Pi board: `sudo apt-get install wiringpi`
2. Run `make` to build programmer
3. Run the following command to start the configuration:

```
./programmer -i <input_file> -w <programming_width>
```

- <*input_file*> is the bitstream file path to configure the board
- <*programming_width*> is the SPI passive configuration width

Extending the Example to Your Own Board

You can use the software examples with any Trion® board that supports the SPI interface. The steps targeting the Trion® T20 MIPI Development Board are applicable when using a different Trion board. However, the jumper connection required depends on the board and SPI passive mode used. Refer to **SPI Interface Pin Mapping** on page 8 for the Raspberry Pi's SPI pins definition.

The number of onboard GPIO available on the Raspberry Pi 3 Model B+ board allows SPI passive of up to x16 only. To use the x32 mode, the Raspberry Pi 3 Model B+ board must be attached with an MCP23017 HAT GPIO expander. Using the GPIO expander reduces the configuration speed. Setups other than the Raspberry Pi 3 Model B+ board with sufficient I/O pins available for the x32 interface can preserve configuration speed.

Note: Trion® FPGAs uses SPI polarity and phase mode 3. You must use mode 3 when using other microcontrollers to configure Trion® FPGAs.

SPI Interface Pin Mapping

The following tables list all Raspberry Pi board GPIO pins and their respective SPI signal names. Connect the Raspberry Pi Board Pin pins to the corresponding pins of your board based on the signal names. Refer to your board schematics for the actual board pin names for each signal.

Table 2: SPI Interface Pin Mapping

Raspberry Pi Board Pin Name	Signal Name	Applicable Mode
GPIO24	SCK	All
GPIO25	SS	All
GPIO0	CDATA0	All
GPIO1	CDATA1	x2, x4, x8, x16, x32
GPIO2	CDATA2	x4, x8, x16, x32
GPIO3	CDATA3	x4, x8, x16, x32
GPIO4	CDATA4	x8, x16, x32
GPIO5	CDATA5	x8, x16, x32
GPIO12	CDATA6	x8, x16, x32
GPIO13	CDATA7	x8, x16, x32
GPIO6	CDATA8	x16, x32
GPIO14	CDATA9	x16, x32
GPIO10	CDATA10	x16, x32
GPIO11	CDATA11	x16, x32
GPIO30	CDATA12	x16, x32
GPIO31	CDATA13	x16, x32
GPIO21	CDATA14	x16, x32
GPIO22	CDATA15	x16, x32
GPIO15	CRST_N	All
GPIO16	CDONE	All
GPIO27	CBUS0	Optional(2)
GPIO28	CBUS1	Optional(2)
GPIO29	CBUS2	Optional(2)

(2) You can drive the `CBUS[2:0]` pins from the Raspberry Pi board. Connect these jumpers and uncomment the `set_cbus_width()` in the software source file.

Table 3: Additional SPI Interface Pin Mapping through GPIO Expander

MCP23017 HAT GPIO Expander Pin Name	Signal Name	Applicable Mode
1	CDATA16	x32
2	CDATA17	x32
3	CDATA18	x32
4	CDATA19	x32
5	CDATA20	x32
6	CDATA21	x32
7	CDATA22	x32
8	CDATA23	x32
9	CDATA24	x32
10	CDATA25	x32
11	CDATA26	x32
12	CDATA27	x32
13	CDATA28	x32
14	CDATA29	x32
15	CDATA30	x32
16	CDATA31	x32

Important: In addition to the SPI interface signals, you also must connect a jumper between the ground pins of the boards.

Revision History

Table 4: Revision History

Date	Version	Description
December 2022	1.2	Corrected the titles for Run the Python Software Example Design and Run the C Software Example Design topics (DOC-1039).
August 2021	1.1	Added note about using SPI mode 3 when configuring Trion FPGAs with microcontroller. (DOC-492)
May 2021	1.0	Initial release.