GateMate™ FPGA Programmer Datasheet

Generic Programmer Board for GateMate

DS1002

Datasheet August 2023

Cologne Chip AG
Eintrachtstr. 113
50668 Köln

Tel.: +49 (0) 221 / 91 24-0
Fax: +49 (0) 221 / 91 24-100

https://colognechip.com
info@colognechip.com

The information presented can not be considered as assured characteristics. Data can change without notice. Parts of the information presented may be protected by patent or other rights. Cologne Chip products are not designed, intended, or authorized for use in any application intended to support or sustain life, or for any other application in which the failure of the Cologne Chip product could create a situation where personal injury or death may occur.

Contents

Contents of Sub-Document DS1002 – Programmer Board Version 1.3 Schematics ↗

Contents of Sub-Document DS1002 – Programmer Board Version 1.4 Schematics ↗

About this Document

This datasheet is the main document in an ensemble of GateMate^{TM} FPGA Programmer documents, which include the following sub-documents:

- **DS1002** – GateMate^{TM} FPGA Programmer Version 1.3 Schematics ↗
- **DS1002** – GateMate^{TM} FPGA Programmer Version 1.4 Schematics ↗

All documents of the programmer are always updated together. Please make sure that you use these documents with the same date.

For more information please refer to the following documents:

- Technology Brief of GateMate^{TM} FPGA ↗
- **DS1001** – GateMate^{TM} FPGA CCGM1A1 Datasheet ↗
- **UG1002** – GateMate^{TM} FPGA Toolchain Installation User Guide ↗

Cologne Chip provides a comprehensive technical support. Please visit our website for more information or contact our support team.

Revision History

This datasheet is constantly updated. The latest version of the document can be found following the link below:

DS1002 – GateMate^{TM} FPGA Programmer Datasheet

Date	Remarks
June 2023	• In the course of the release of the PCB version 1.4, the previous datasheet was split into several documents. The main document is valid for all versions. For each version, the schematics, BOM and other board-specific information are available in a separate sub-document. • Some minor changes have been made to board version 1.4, see details in Table 1.1 on page 9.
March 2022	Initial release.

Chapter 1

Introduction

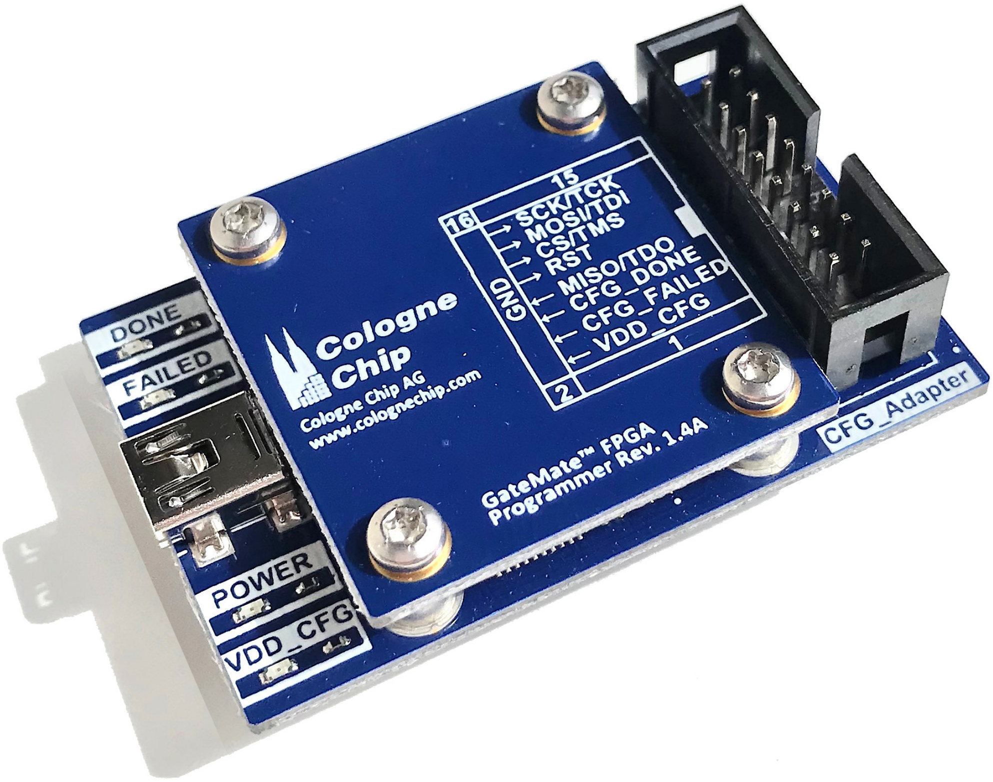


Figure 1.1: *Top view of the GateMate^{TM} FPGA Programmer 1.4 (including module cover)*

The GateMate^{TM} FPGA Programmer allows an easy and flexible access to the GateMate^{TM} FPGA. The configuration can be loaded from any host computer via USB interface into the FPGA. Furthermore, the flash memory, which is typically available at the user application can be accessed as well.

The main features of the programmer are

- `FTDI` `FT232HQ` USB bridge
- USB 2.0 to JTAG and SPI bridge provided by FTDI's Multi-Protocol Synchronous Serial Engine (MPSSE)
- Mini-USB B connector
- USB-powered
- Configuration voltage `VDD_CFG` required from user application
- 4 status LEDs
- Form factor 35×55 mm

This document describes how to set up the GateMate^{TM} FPGA Programmer and explains the different use cases.

☝ **Please note!**

This datasheet covers the following versions of the GateMate^{TM} FPGA Programmer:

- Version 1.3
- Version 1.4 with minor changes

Differences between these two boards are listed in Table 1.1 on page 9.

Table 1.1: *Differences between programmer versions 1.3 and 1.4*

Ver.	Changes
1.3	C19 is 6.3V type.
1.4	C19 is 10V type.
1.3	C22 is 6.3V type.
1.4	C22 is 10V type.
1.3	U3 is dual NAND gate SN74AUP2G00DCUR (8 pins).
1.4	U3 is quad NAND gate SN74AUC00RGYR (15 pins).
1.4	Inverter U5 (SN74AUP1G04DRYR, 6 pins) added.
1.3	U4 is octal bi-directional transceiver SN74AXC8T245PWR (24 pins).
1.4	U4 is octal FET switch SN74CB3T3245PWR (20 pins).
1.4	Pull-up resistors R13 and R14 removed at U4 input pins.
1.4	Removed port `FF_RESET` from module `Programmer (FTDI 232H)`.
1.3	`DONE` signal from the FPGA is directly feed to host controller via connector `CFG_ADAPTER`, level shifter U4 and USB interface U1.
1.4	`DONE` signal from the FPGA is latched in flip-flop U3 between `CFG_ADAPTER` connector and level shifter U4.
1.3	`FAILED` signal from the FPGA is feed to host controller via connector `CFG_ADAPTER`, level shifter U4, flip-flop U3 and USB interface U1.
1.4	`FAILED` signal from the FPGA is latched between connector `CFG_ADAPTER` and level shifter U4.
1.3	Latched `FAILED` signal in flip-flop U3 is reset from signal `FF_RESET` which is a separat reset signal from the host controller.
1.4	Latched `FAILED` and `DONE` signals in flip-flops U3 are reset from signal `T_RST` which is the common reset signal from the host controller.

see schematics in GateMate$^{\text{TM}}$ FPGA Programmer Version 1.3 Schematics
GateMate$^{\text{TM}}$ FPGA Programmer Version 1.4 Schematics

Chapter 2

Block Diagram

Figure 2.1 shows the block diagram of the GateMate^{TM} FPGA Programmer and its embedding between the host controller and the user application.

Typically, the host controller is any computer that comes with an USB interface. The programmer software runs on both `Microsoft Windows` and `Linux`.

Four LEDs display the current state of the programmer device as described in Table 2.1.

Table 2.1: *GateMate^{TM} FPGA Programmer LED signaling*

LED	color	Description
D1	green	USB power available from host controller
D3	green	`Configuration Done` signaling
D4	red	`Configuration Failed` signaling
D5	green	Power from FPGA application (configuration GPIO bank) available

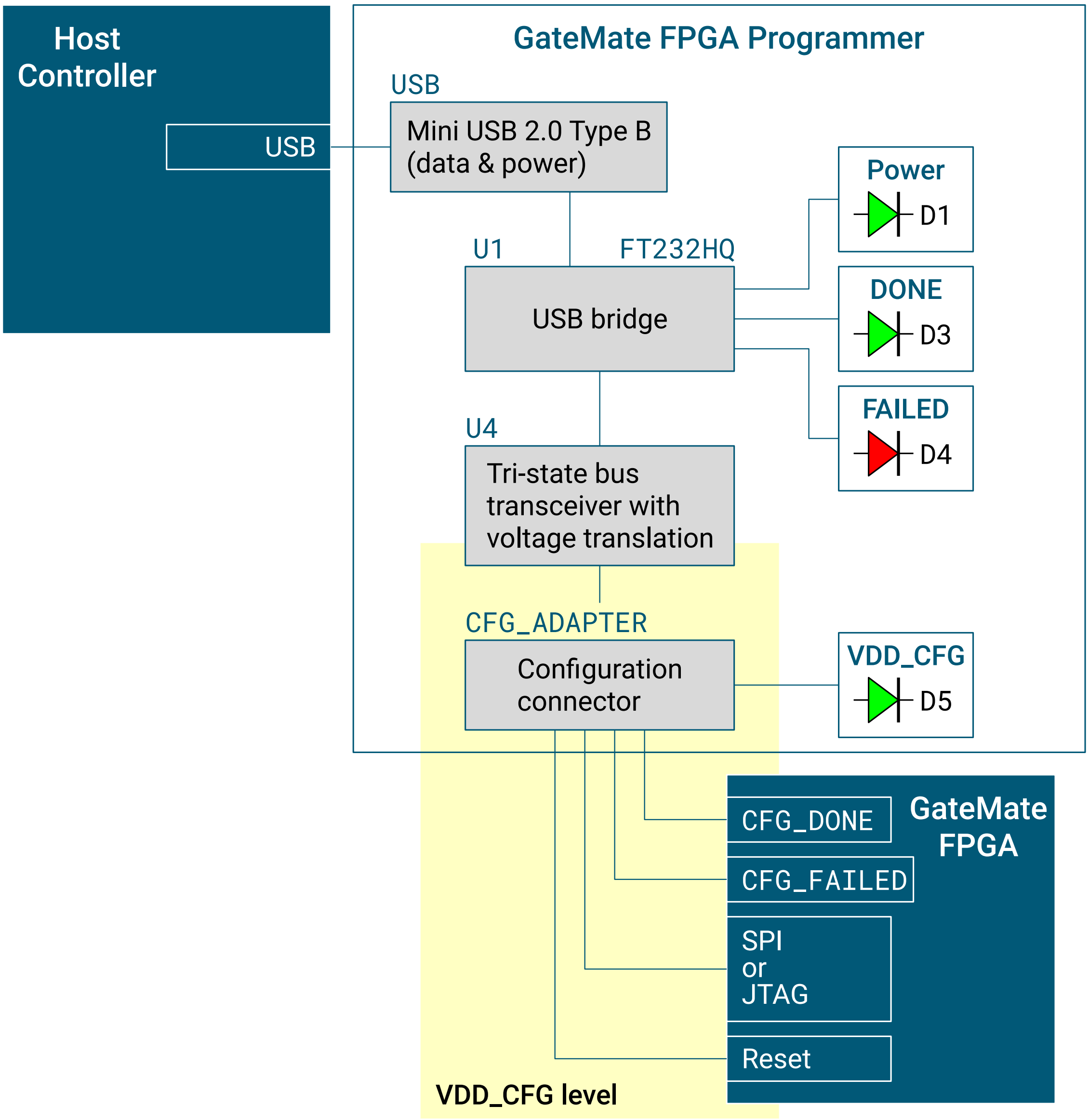


Figure 2.1: *Block diagram of the GateMate^{TM} FPGA Programmer*

Chapter 3

Startup

3.1 Connection to the User Application

To connect the GateMate^{TM} FPGA Programmer to the user application, the connector CFG_Adapter must be used. Figure 3.1 shows the signals of this connector. Depending on the user's requirements, it can act as JTAG or SPI bus. The detailed BGA connection is described in Table 3.1 for both interfaces.

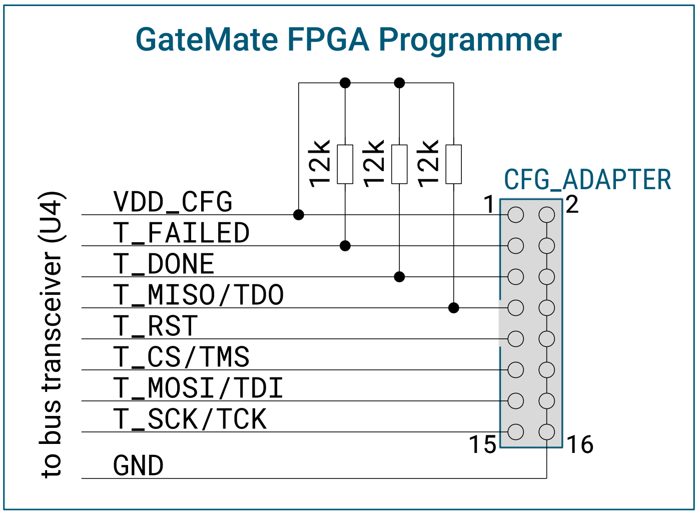


Figure 3.1: *Connecting the GateMate^{TM} FPGA Programmer with the GateMate FPGA*

The GateMate^{TM} FPGA Programmer is USB bus powered. The connector CFG_Adapter requires power supply from the FPGA configuration GPIO bank (signal VDD_CFG) for operation. The pull-up resistors shown in Figure 3.1 are populated on the programmer's printed circuit board (PCB).

No further hardware setup is required.

Table 3.1: *GateMate^{TM} FPGA Programmer configuration adapter*

GateMate^{TM} FPGA Programmer		GateMate FPGA		GateMate FPGA	
CFG_ADAPTER		**JTAG interface**		**SPI interface**	
pin	**signal name**	**ball**	**signal name**	**ball**	**signal name**
1	VDD_CFG	U3[1]	VDD_WA	U3[1]	VDD_WA
3	T_FAILED	V2	CFG_FAILED_N	V2	CFG_FAILED_N
5	T_DONE	V3	CFG_DONE	V3	CFG_DONE
7	T_MISO / TDO	U2	JTAG_TDO	P1	SPI_D1
9	T_RST	T15	RST_N	T15	RST_N
11	T_CS / TMS	T3	JTAG_TMS	N3	SPI_CS_N
13	T_MOSI / TDI	T2	JTAG_TDI	P2	SPI_D0
15	T_SCK / TCK	R3	JTAG_TCK	N4	SPI_CLK
2,4,6,. . .,16	GND	V18[1]	GND	V18[1]	GND

[1] and others

3.2 Connection to the Host Controller

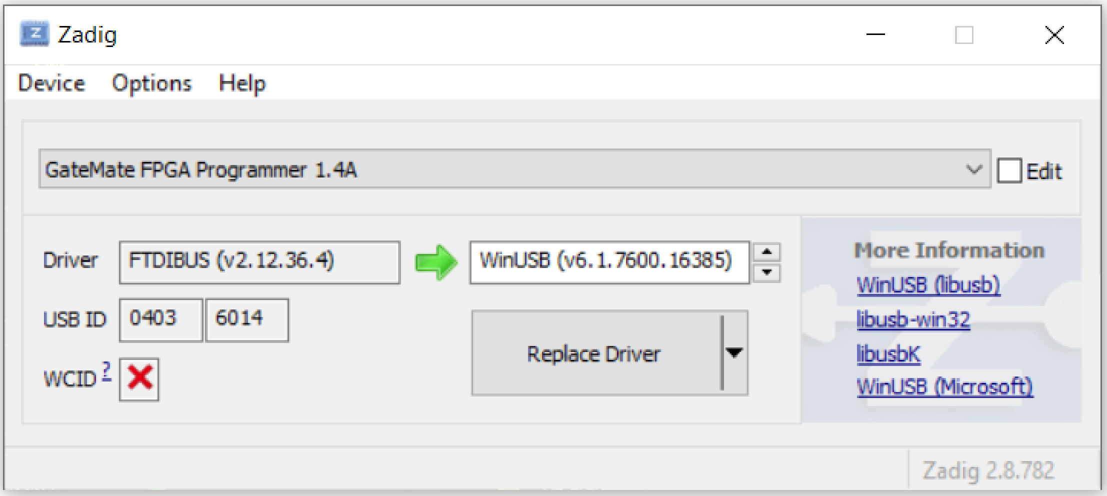
The GateMate^{TM} FPGA Programmer requires a computer with Linux or Windows operating system as follows:

- Supported Linux environments:
  - Debian-based Linux (Debian, Ubuntu, ...) with `apt` package manager
  - Arch-based Linux (Arch, Manjaro, ...) with `pacman` package manager
  - Red Hat-based Linux (Fedora, ...) with `dnf` or `yum` package manager
- Windows environments:
  - Windows 7 or later, 64 bit
  - Zadig USB driver installer
    `https://zadig.akeo.ie/`

When first plugged into the computer's USB port, drivers should load by default.

In Windows environments, however, it is necessary to change the default USB driver using Zadig ↗. Download the software and connect the GateMate^{TM} FPGA Progammer to any USB port. In the Zadig Window, select **Options > List All Devices** to refresh the device list. From the drop-down list, select **GateMate^{TM} FPGA Programmer 1.4**. Now select **libusb-win32 (any version)** from the driver list and replace the drivers (see Figure 3.2).

Replacing drivers might take a moment. Your GateMate^{TM} FPGA Programmer should then be listed as **libusb-win32** device in the Device Manager as shown in Figure 3.3.



GateMate FPGA Programmer 1.4A

Driver FTDIBUS (v2.12.36.4) → WinUSB (v6.1.7600.16385)

USB ID 0403 6014

WCID ? ✗

Figure 3.2: *Zadig Window with selected GateMate^{TM} FPGA Programmer*

In Linux environments, no driver installation is required. The device should enumerate as `FT232H` using the `lsusb` command.

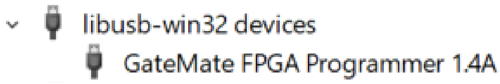

libusb-win32 devices
GateMate FPGA Programmer 1.4A

Figure 3.3: *GateMate^{TM} FPGA Programmer in Device Manager*

The tool setup for the Cologne Chip GateMate^{TM} series is described in the user guide **UG1002** – GateMate^{TM} FPGA Toolchain Installation User Guide

Chapter 4

Loading the FPGA Configuration

GateMate FPGA configuration can be loaded either via the JTAG or SPI interfaces. Both options are described in this chapter.

It is assumed that the user application has, among other things, a GateMate FPGA and a flash memory from which the configuration is loaded during normal operation. In this chapter it is described, how configuration bit files can be handled alternatively with the GateMate^{TM} FPGA Programmer during the application development.

4.1 Loading the FPGA Configuration via the JTAG Interface

In this mode, the programmer is configured to send the configuration bitstream directly to the FPGA via JTAG as shown in Figure 4.1. The FPGA configuration mode pins `CFG_MD[3:0]` must be set to `0x C` (JTAG).

An active `CFG_DONE` and inactive `CFG_FAILED` signal indicates successfull configuration. The `CFG_FAILED` signal may send a pulse during configuration in case of any error. Note that the `CFG_DONE` and `CFG_FAILED` pins can be configured as user GPIO. In this case no configuration status can be read back.

4.2 Loading the FPGA Configuration via the SPI Interface

In this mode, the programmer is configured to send the configuration bitstream directly to the FPGA via SPI as shown in Figure 4.2. The FPGA configuration mode pins `CFG_MD[3:0]` must be set to `0x 4` (SPI slave mode).

An active `CFG_DONE` and inactive `CFG_FAILED` signal indicates successfull configuration. The `CFG_FAILED` signal may send a pulse during configuration in case of any error. Note

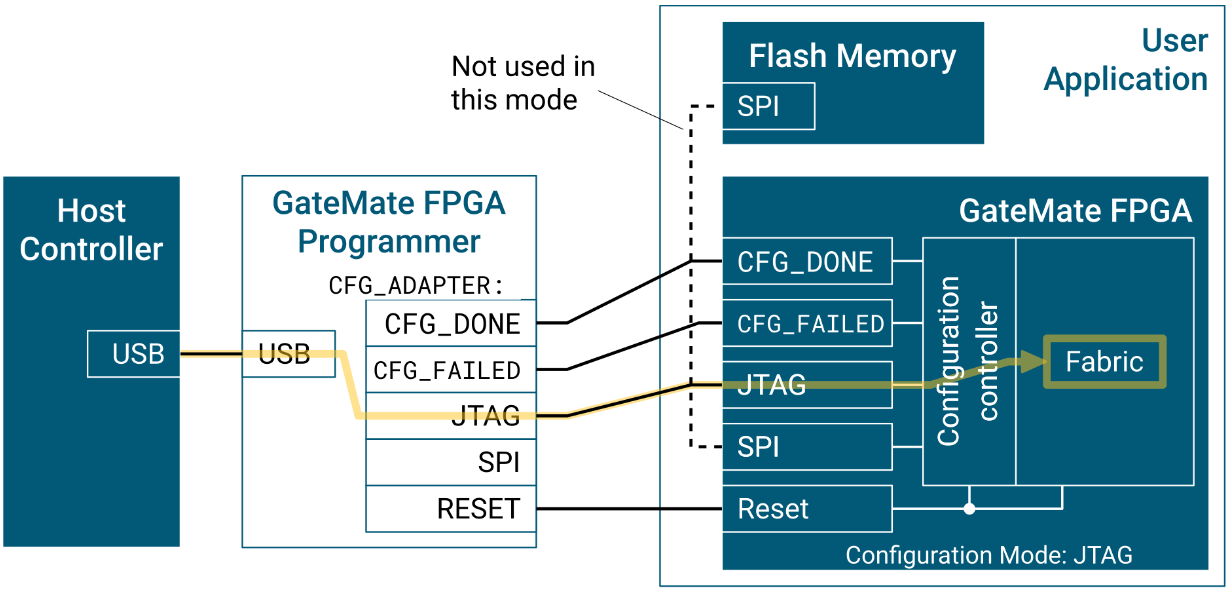


Figure 4.1: *Loading the FPGA configuration via the JTAG interface*

that the `CFG_DONE` and `CFG_FAILED` pins can be configured as user GPIO. In this case no configuration status can be read back.

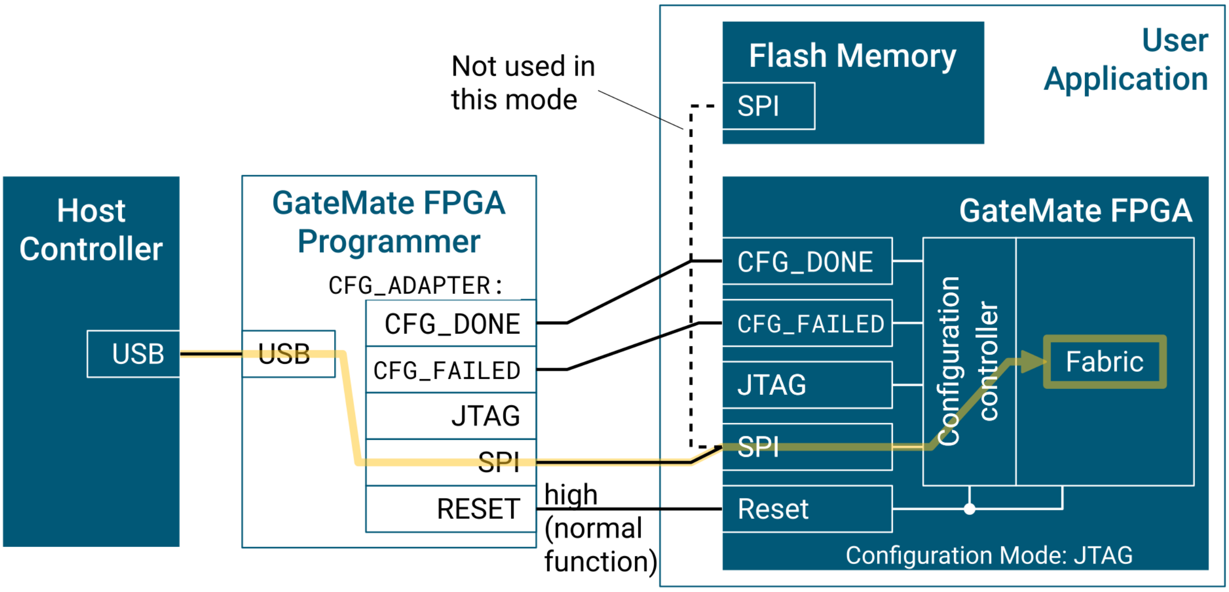


Figure 4.2: *Loading the FPGA configuration via the SPI interface*

Chapter 5

Access to the Flash Memory

The GateMate$^{\text{TM}}$ FPGA Programmer offers an easy way to access the user application's flash memory. Again, two ways are possible as described in this chapter.

5.1 Access to the Flash Memory via the JTAG Interface

The GateMate$^{\text{TM}}$ FPGA configuration controller has a built-in JTAG-SPI-bridge to access an external SPI flash memory using the JTAG interface.

Handling of the JTAG-SPI-bridge is described in the GateMate$^{\text{TM}}$ FPGA datasheet.

In this mode, the programmer is configured to send the configuration bitstream or any other data to the external SPI flash memory via JTAG as shown in Figure 5.1. The FPGA configuration mode pins `CFG_MD[3:0]` must be set to `0x C` (JTAG). Further prerequisite is that the FPGA configuration bank is not configured as user GPIO.

In this configuration, both `CFG_DONE` and `CFG_FAILED` signals don't output any status information.

5.2 Access to the Flash Memory via the SPI Interface

The following configuration requires a direct connection of the SPI signals to the external flash memory. Both external flash memory and GateMate$^{\text{TM}}$ FPGA may share the same signals.

In this mode, the programmer is configured to send the configuration bitstream or any other data directly to the external SPI flash memory as shown in Figure 5.2. Prerequisite is that the FPGA acts passively on the SPI bus. If both FPGA and external flash memory share

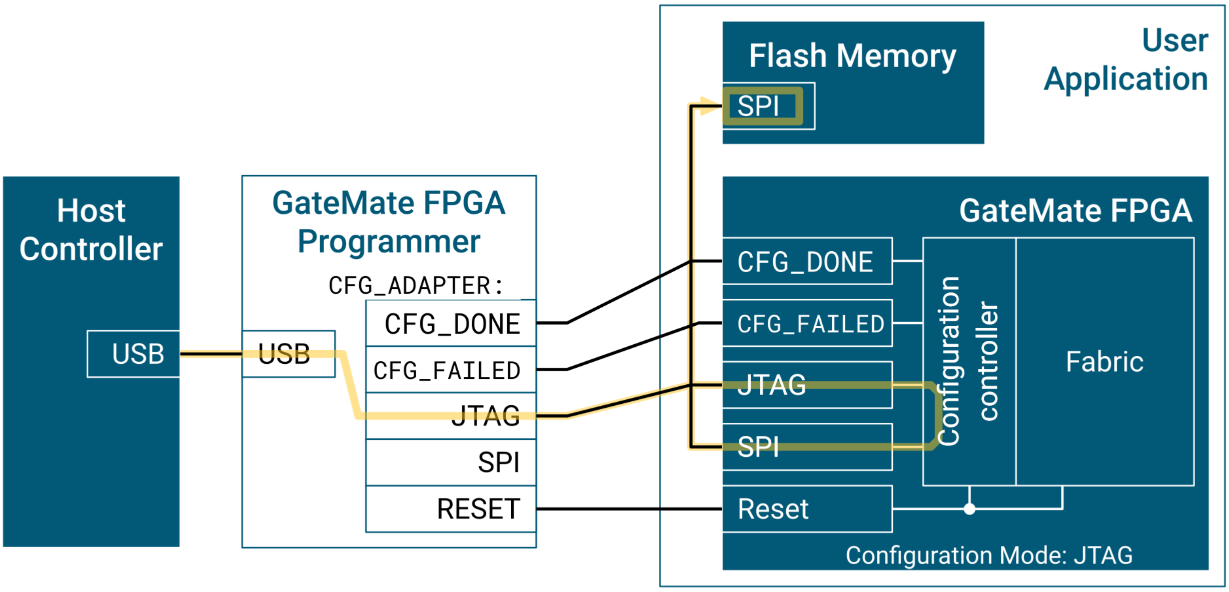


Figure 5.1: *Access to the flash memory via the JTAG interface*

the same bus signals, it is required to keep the FPGA in active reset while programming the external flash memory.

After successfull programming, the reset signal may be released to start the FPGA configuration e.g. in SPI master mode.

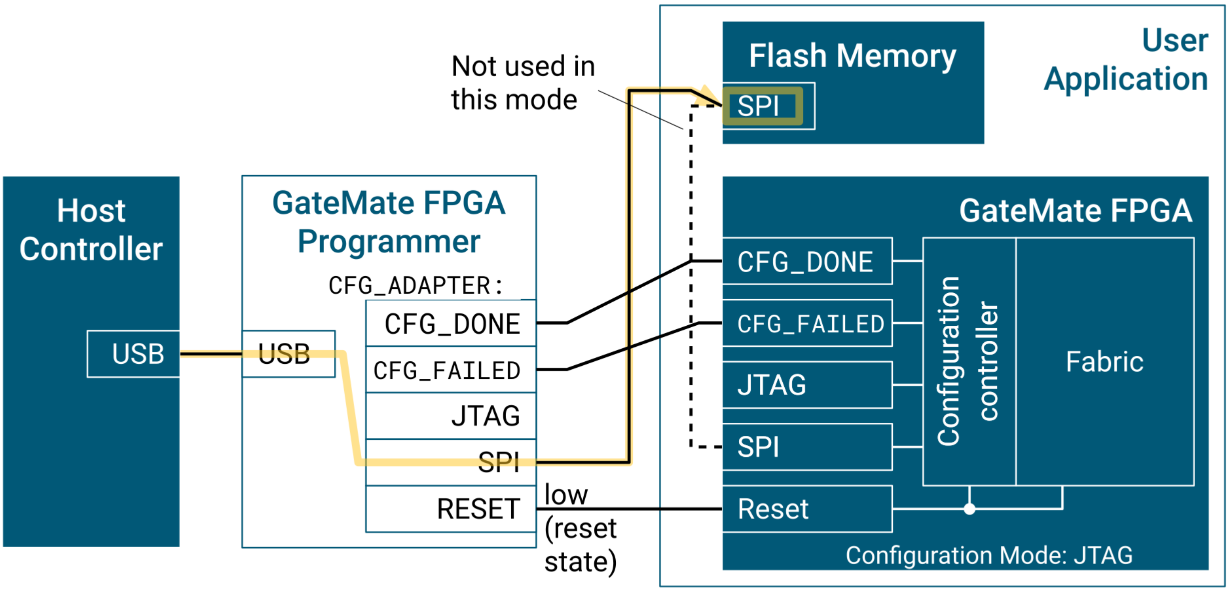


Figure 5.2: *Access to the flash memory via the SPI interface*

Chapter 6

Electrical Characteristics

Table 6.1: *Absolute maximum characteristics of the GateMate^{TM} FPGA Programmer*

Symbol	Min	Typ	Max	Unit	Description
	-40		125	°C	Junction temperature
V_{BUS}	-0.3		5.8	V	USB Voltage
VDD_{CFG}	-0.5		4.2	V	Target Supply Voltage
V_i	-0.5		4.2	V	Input Signals on CFG_ADAPTER
V_o	-0.5		4.2	V	Output Signals on CFG_ADAPTER
T_{stg}	-55		150	°C	Storage temperature

Table 6.2: *Operating characteristics of the GateMate^{TM} FPGA Programmer*

Symbol	Min	Typ	Max	Unit	Description
	-40		85	°C	Operating temperature
V_{BUS}	4.75	5.0	5.25	V	USB Voltage
VDD_{CFG}	1.4		2.75	V	Target Supply Voltage
V_i	0		2.75	V	Input Signals
V_o	0		2.75	V	Output Signals Active
	0		VDD_{CFG}	V	Output Signals Tri-State

GateMate^{TM} FPGA
Programmer Datasheet
Generic Programmer Board for GateMate
DS1002
August 2023

Cologne Chip