terasIC
www.terasic.com

Atum-A3-Nano

Powerful performance in a compact platform !

User Manual

CONTENTS

Chapter 1

Atum A3 Nano Board

The Atum A3 Nano board delivers powerful performance in a compact, affordable platform. Powered by Altera's largest Agilex 3 FPGA with 135K LEs, the board features 64MB of SDRAM, an on-board USB-Blaster III with USB Type-C connection, HDMI output, Gigabit Ethernet and MicroSD storage—all within 85mm x 70mm form factor.

Designed for versatility, the Atum A3 Nano includes multiple user I/Os (LEDs, buttons, switches), a 2x20 GPIO connector, and dual 2x6 TMD GPIO headers. Whether you are developing robotics, aut omotive solutions, smart city, consumer electronics, or advanced image processing applications, the Atum A3 Nano provides the performance and expandability you need.

1.1 Package Contents

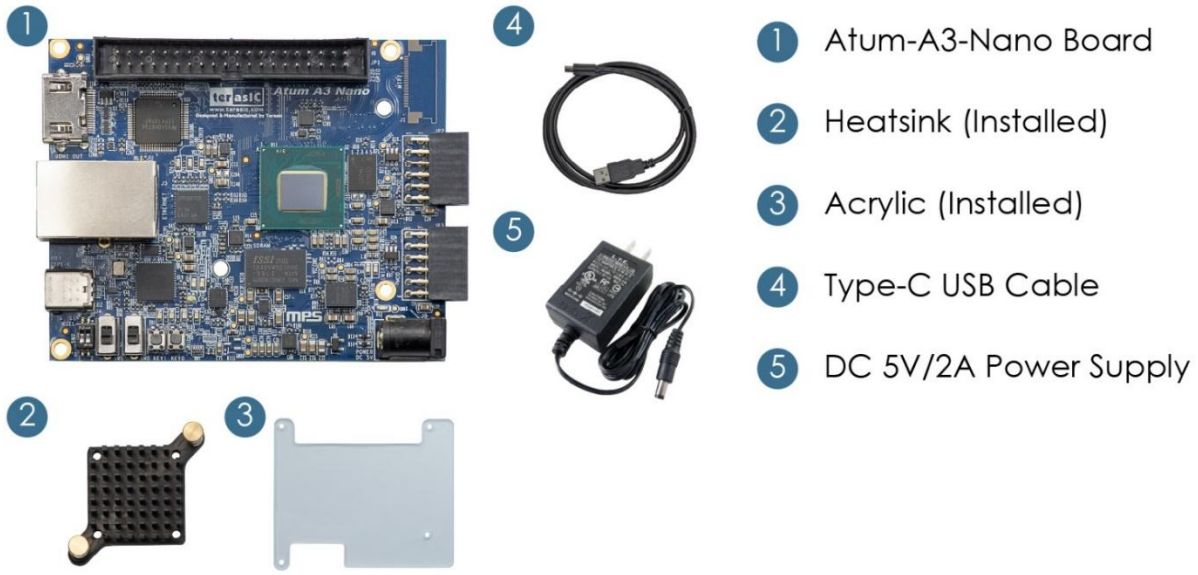

Figure 1-1 The Atum A3 Nano package contents

The Atum A3 Nano package includes:

1. Atum A3 Nano Board
2. Heatsink (Installed)
3. Acrylic (Installed)
4. Type-C USB Cable
5. DC 5V/2A Power Supply

1.2 Atum A3 Nano System CD

The Atum A3 Nano System CD contains all the documents and supporting materials associated with Atum A3 Nano, including the user manual, reference designs and device datasheets. Users can download this system CD from the link: http://atum-a3-nano.terasic.com/cd.

The developers can create their Quartus project based on the **golden_top** Quartus project included in this CD. The **golde_top** Quartus project is placed in the folder: *Demonstrations/golden_top* .

1.3 Getting Help

Here are the addresses where you can get help if you encounter any problems:

- Terasic Technologies
- No.80, Fenggong Rd., Hukou Township, Hsinchu County, 303035 Taiwan

Email: support@terasic.com

Tel.: +886-3-575-0880

Website: atum-a3-nano.terasic.com

Chapter 2

Introduction to the Atum A3 Nano Board

This chapter provides an introduction to the design and features of the board.

2.1 Layout and Components

Figure 2-1 and Figure 2-2 shows a photograph of the board that illustrates its layout and the location of connectors and key components.

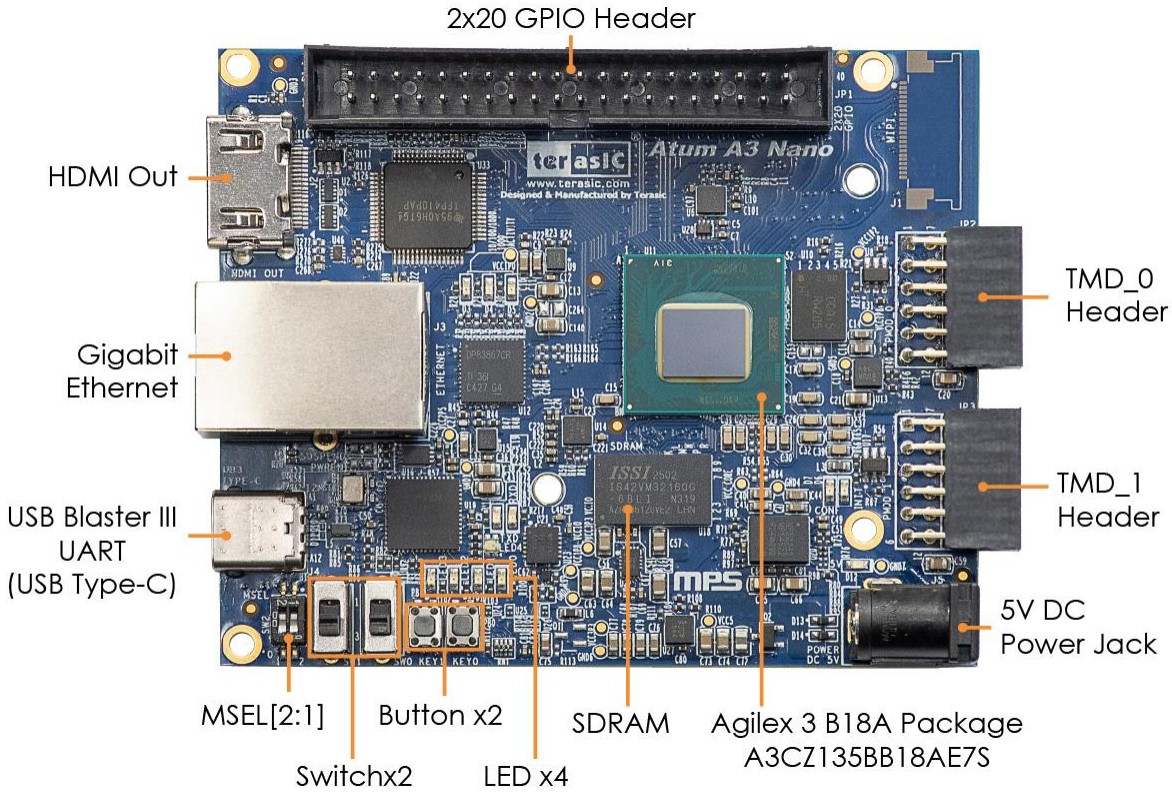

Figure 2-1 Atum A3 Nano board (top view)

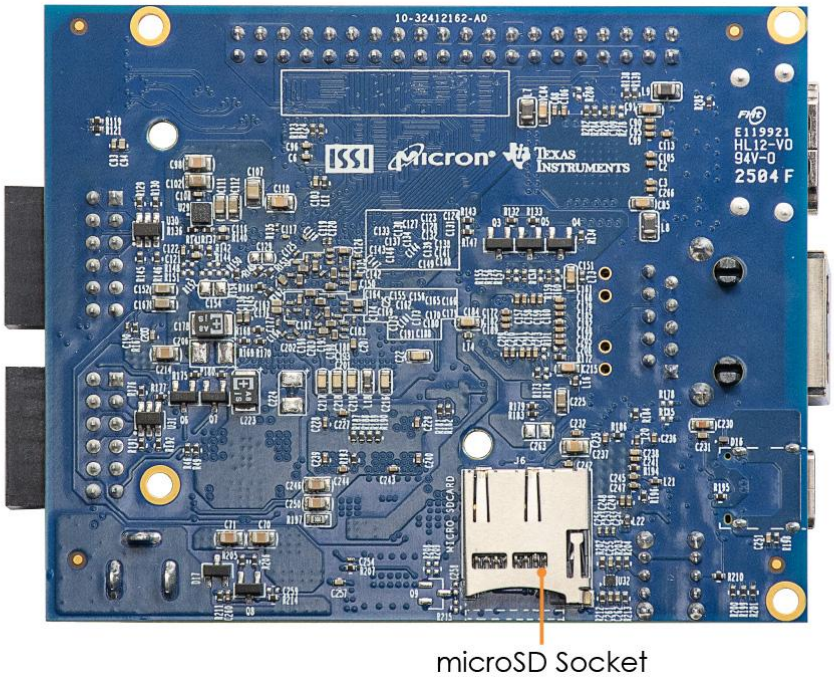

Figure 2-2 Atum A3 Nano board (bottom view)

The Atum A3 Nano board has many features that allow users to implement a wide range of designed circuits, from simple circuits to multimedia projects.

The following hardware is provided on the board:

- **FPGA: Agilex 3 A3CZ135BB18AE7S Device**
  - 135,110 Logic Elements
  - 6.89 Mbit M20K, 1.4 Mbit MLAB
  - 368 18x19 Multipliers
  - 4 IO PLL, 8 Fabric I/O PLL
- **Programming/Debug and Configuration**
  - On-Board USB Blaster III (Type C USB connector)
  - ASx4 Mode with 128Mbit QSPI Flash
- **Memory Device**
  - 64MB SDRAM, x32 bits data bus
  - MicorSD Card Socket
- **Communiation**
  - Gigabit Etherent
  - UART
- **Display**

- HDMI Output

- **Expansion Connectors**
  - One 2x20 GPIO Connector (voltage level: 3.3V)
  - Two 2x6 TMD GPIO Connectors (voltage level: 3.3V; PMOD compatible)

- **Switches/Buttons/LEDs**
  - 4 LEDs
  - 2 Slide Switches
  - 2 Push Buttons

- **Power**
  - 5V DC input

2.2 Block Diagram of the Atum A3 Nano Board

Figure 2-3 is the block diagram of the board. All the connections are established through the Agilex 3 FPGA device to provide maximum flexibility for users. Users can configure the FPGA to implement any system design.

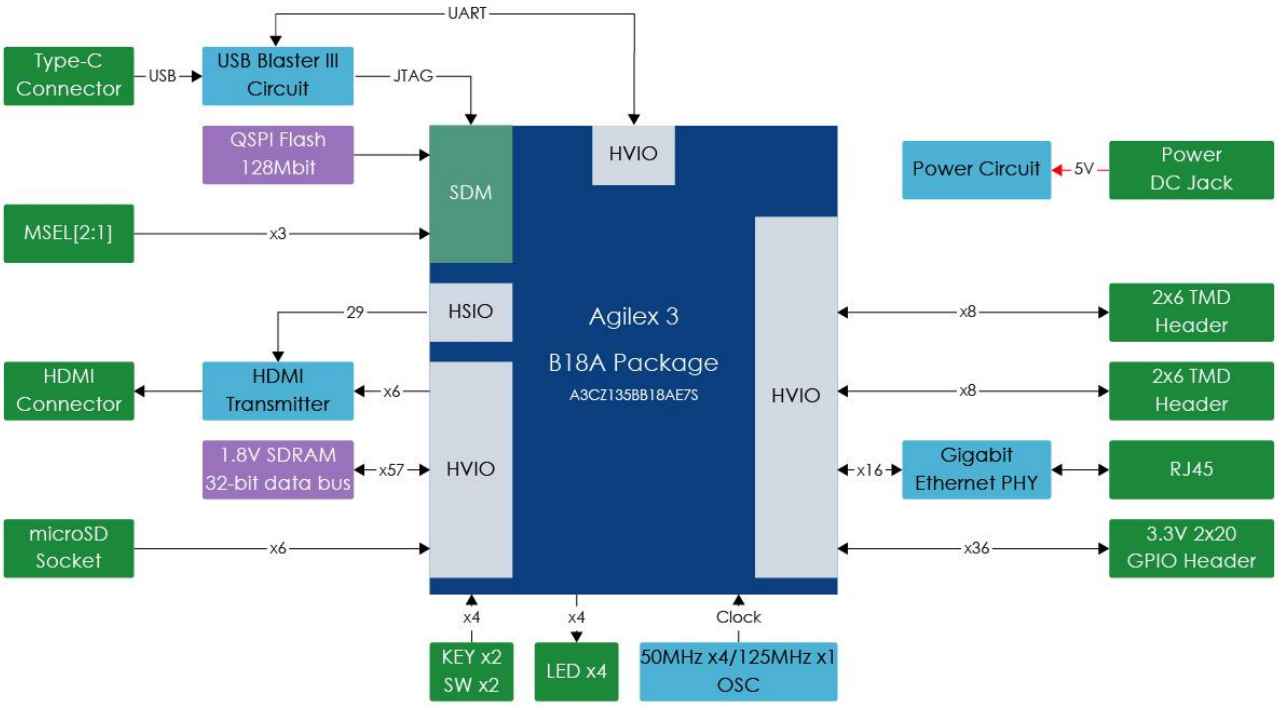

Figure 2-3 Block diagram of Atum A3 Nano

The components shown in Figure 2-3 are described in detail in Section 2.1.

Chapter 3

Using the Atum A3 Nano Board

This chapter provides instructions for using the board and describes its peripherals.

3.1 Settings of FPGA Configuration Mode

When the Atum A3 Nano board is powered on, the FPGA is configured from QSPI Flash. The MSEL[2:1] switches are used to select the configuration scheme, implemented as a 2-pin DIP switch **SW2** on the Atum A3 Nano board, as shown in Figure 3-1.

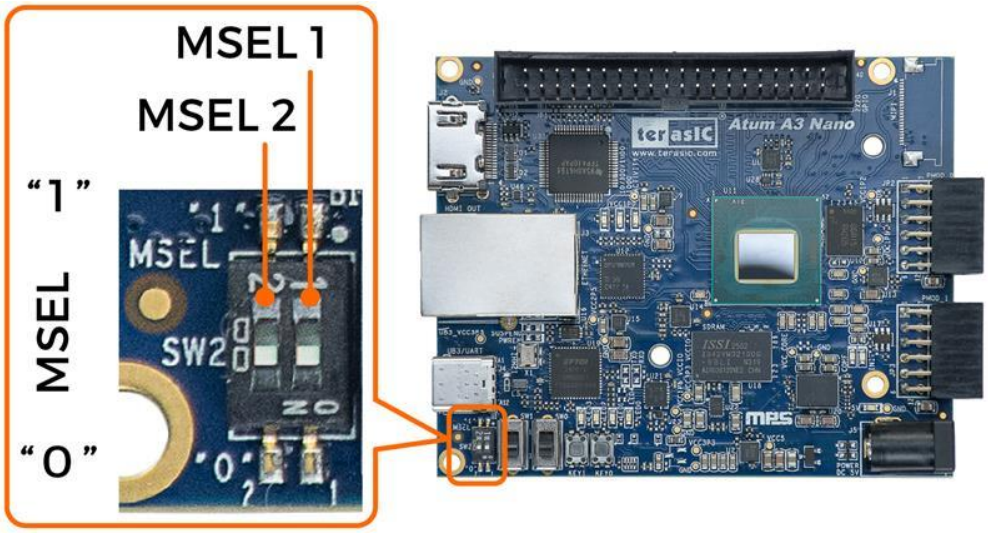

Figure 3-1 DIP switch (SW2) setting of Active Serial (AS) mode on Atum A3 Nano board

Table 3-1 shows the relation between MSEL[2:1] and DIP switch (SW2).

Table 3-1 FPGA Configuration Mode Switch (SW2)

Board Reference	*Signal Name*	*Description*	*Default AS Mode*
SW2.1	MSEL1	Use these pins to set the FPGA Configuration scheme	ON ("0")
SW2.2	MSEL2		ON ("0")

Figure 3-1 shows MSEL[2:1] setting of Active Serial (AS) Fast mode, which the default setting on the Atum A3 Nano. When the board is powered on, the FPGA is configured from QSPI Flash, which is pre-programmed with a default configuration.

Table 3-2 MSEL Pin Settings for FPGA Configuration of Atum A3 Nano

MSEL[2:1]	*Configuration Scheme*	*Description*
00	AS Fast	FPGA configured from QSPI Flash (default)
11	JTAG	You can configure the FPGA using the dedicated JTAG interface and circuit.

3.2 Configuration of Agilex 3 FPGA on Atum A3 Nano

There are two programming methods supported by Atum A3 Nano:

1. JTAG programming: This is named after the IEEE standards Joint Test Action Group. The configuration bitstream is downloaded directly to the Agilex 3 FPGA. The FPGA will retain its status as long as the power is applied to the board; the configuration information will be lost when the power is turned off.
2. AS programming: The other programming method is Active Serial configuration. The configuration bitstream is written to the quad serial configuration device (QSPI Flash), which provides non-volatile storage for the bit stream. The information is retained within QSPI Flash even if the Atum A3 Nano board is turned off. When the board is powered on, the configuration data in the QSPI Flash device is automatically loaded into the Agilex 3 FPGA.

■ JTAG Chain on Atum A3 Nano Board

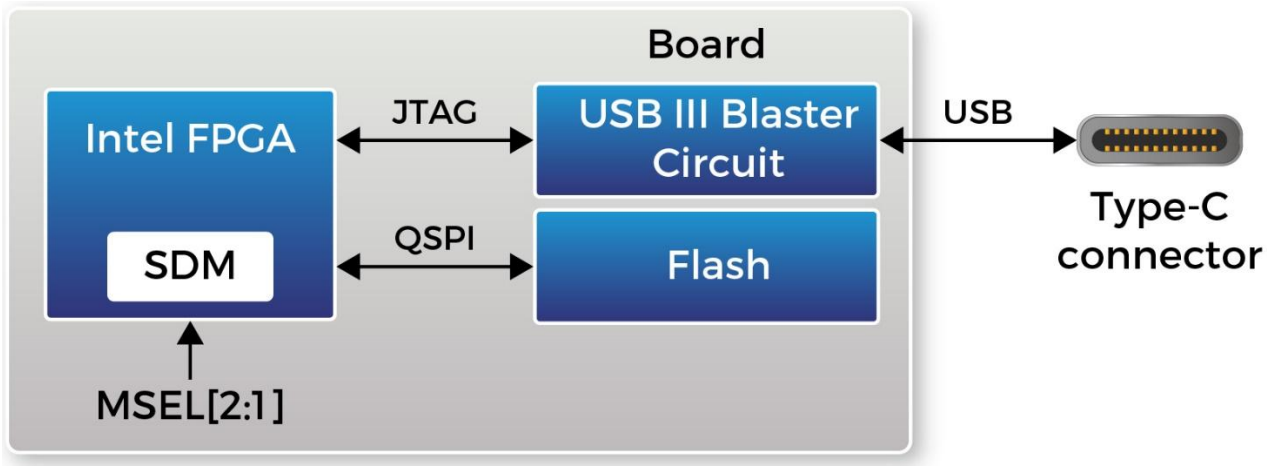

Figure 3-2 shows the JTAG interface of Atum A3 Nano board, which uses the USB Blaster III circuit to connect to the host PC. Users can configure or debug the Agilex 3 FPGA on the board through the USB Type-C interface and Quartus software on the host PC.

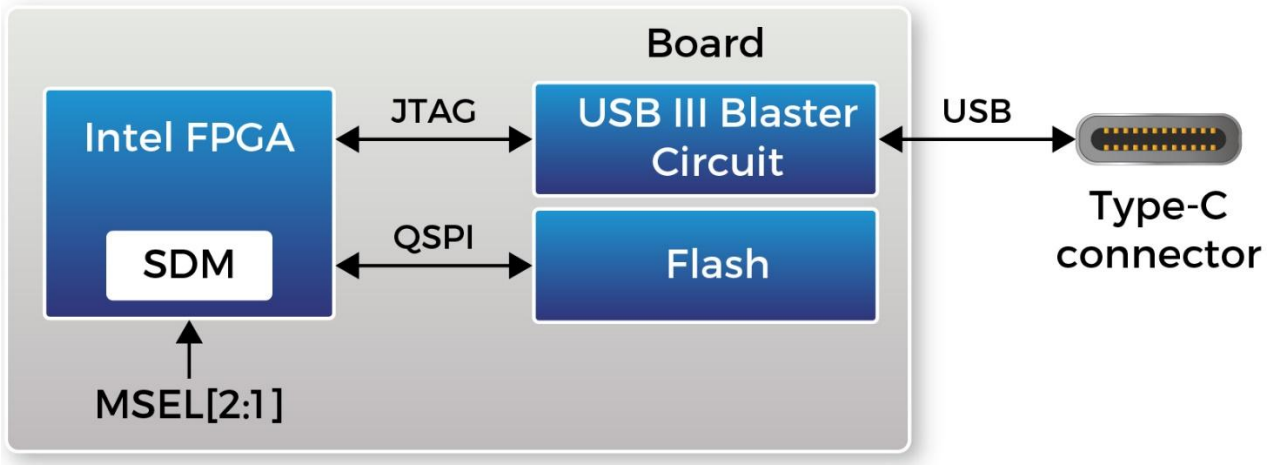

Figure 3-2 Block diagram of the JTAG chain

■ Configuring the FPGA in JTAG Mode

The following explains step-by-step how to program the FPGA in JTAG mode.

1. Make sure the Quartus Pro v25.1 and the driver of USB Blaster III are installed on your Host.
2. Open the Quartus Programmer tool, make sure the USB blaster III ("Atum A3 Nano[USB-x]") is found in "Hardware Setup.." tab.

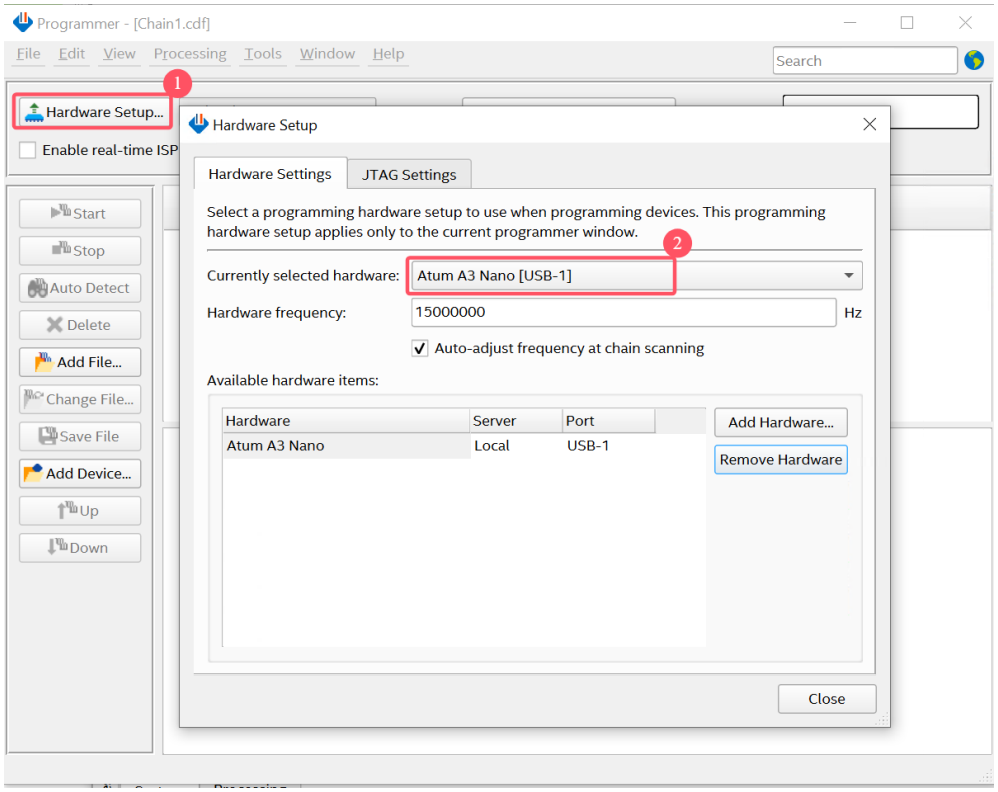

Figure 3-3 USB blaster III is found in Programmer

3. Open the Programmer and click "**Auto Detect**", as shown in Figure 3-4.

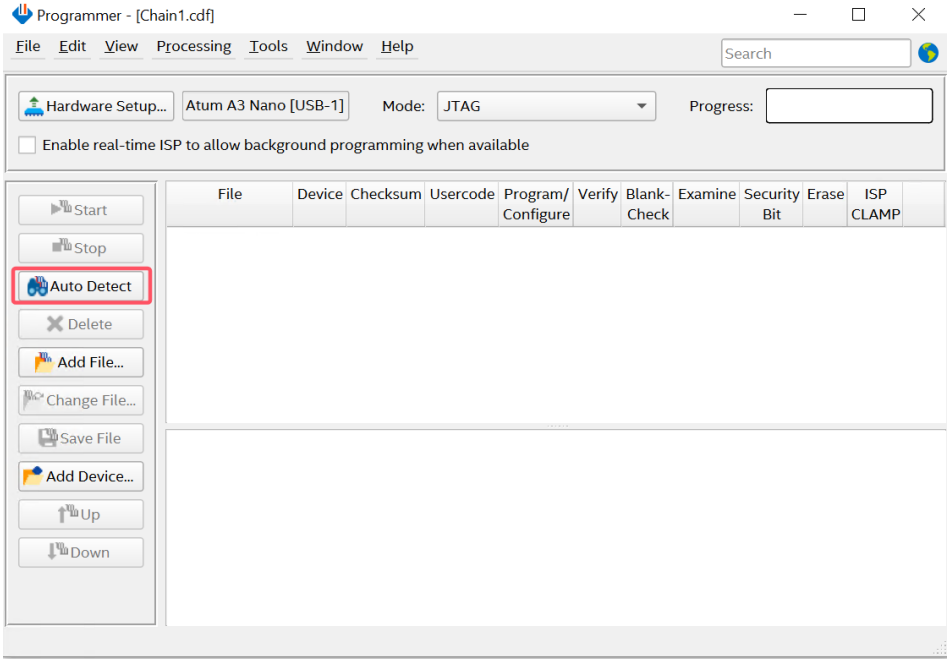

Figure 3-4 Detect FPGA device in JTAG mode

4. The Agilex 3 FPGA should be detected, as shown in Figure 3-5.

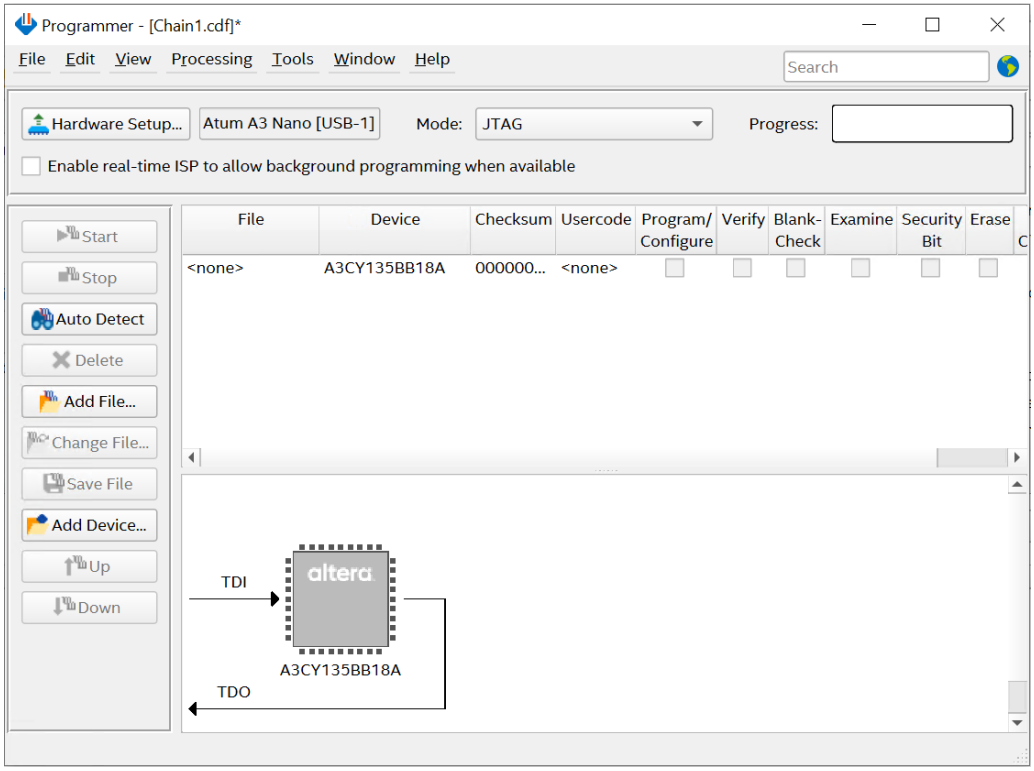

Figure 3-5 FPGA detected in Quartus programmer

5. Right click on the FPGA device and open the .sof file to be programmed, as shown in Figure 3-6.

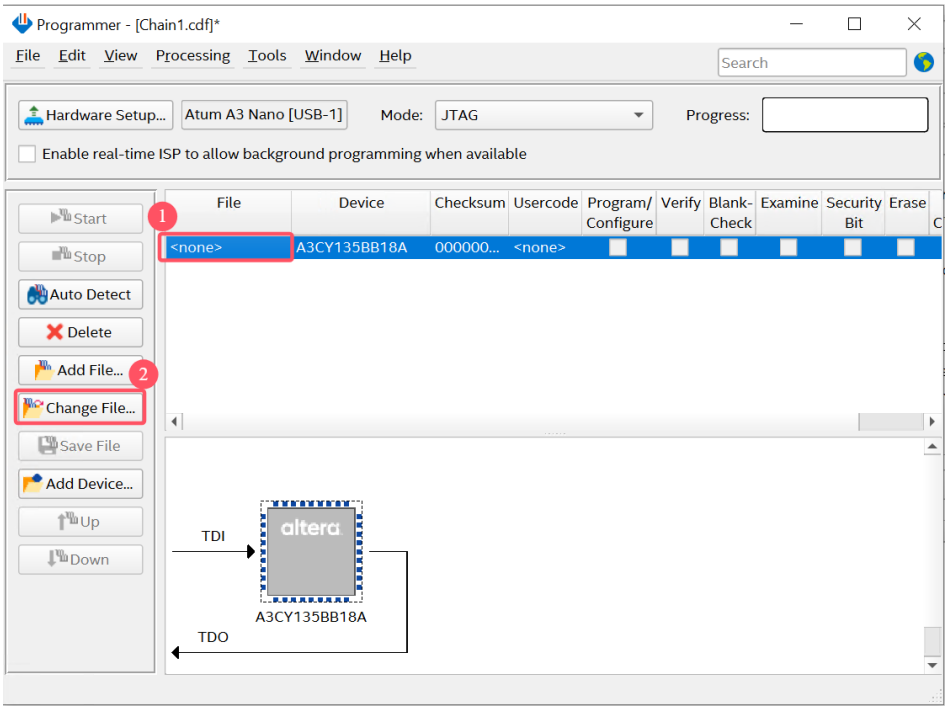

Figure 3-6 Open the .sof file to be programmed into the FPGA device

6. Select the .sof file to be programmed, as shown in Figure 3-7.

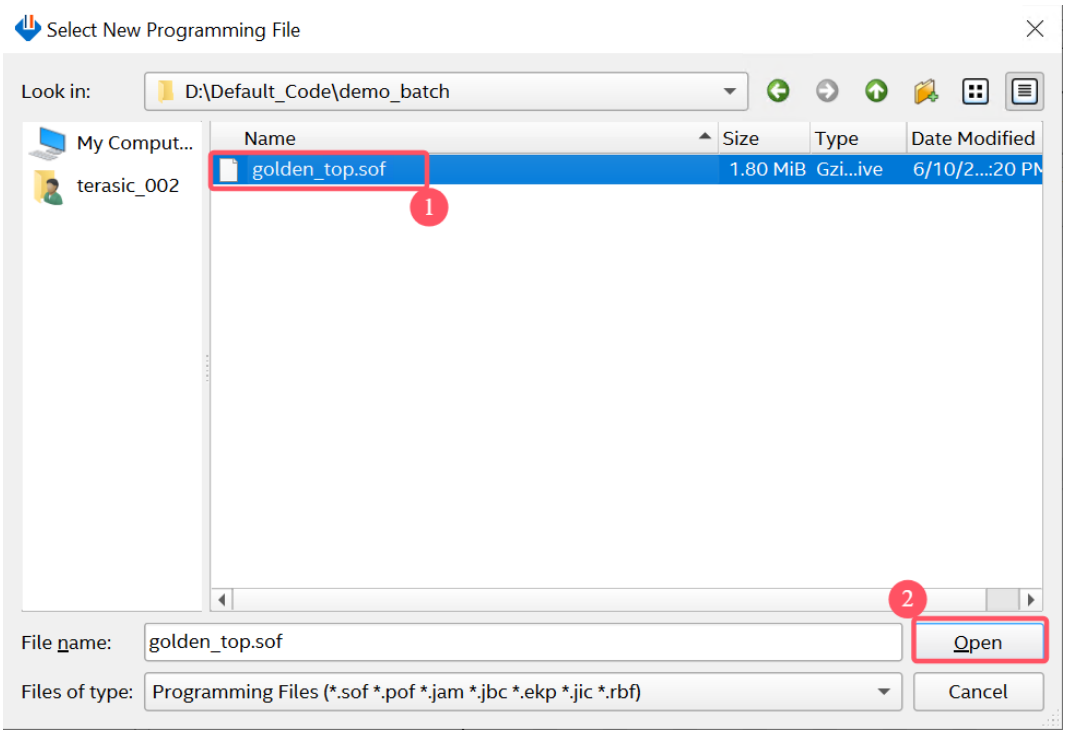

Figure 3-7 Select the .sof file to be programmed into the FPGA device

7. Click the "Program/Configure" checkbox and click "Start" button to download the .sof file into the FPGA device, as shown in Figure 3-8 and Figure 3-9.

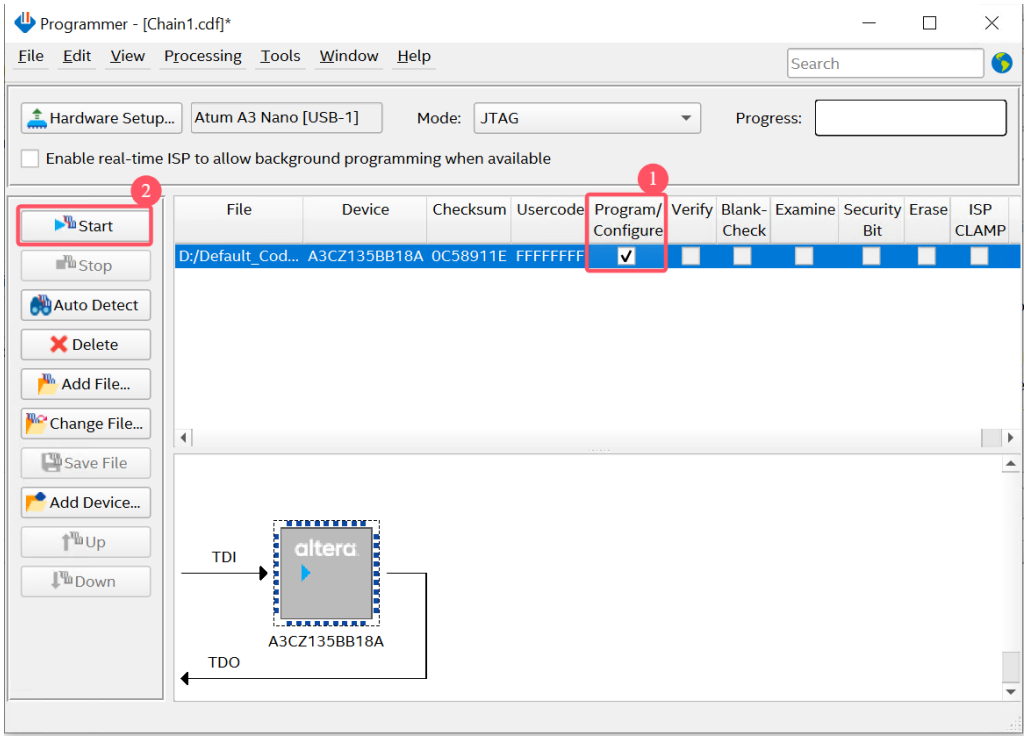

Figure 3-8 Program .sof file into the FPGA device

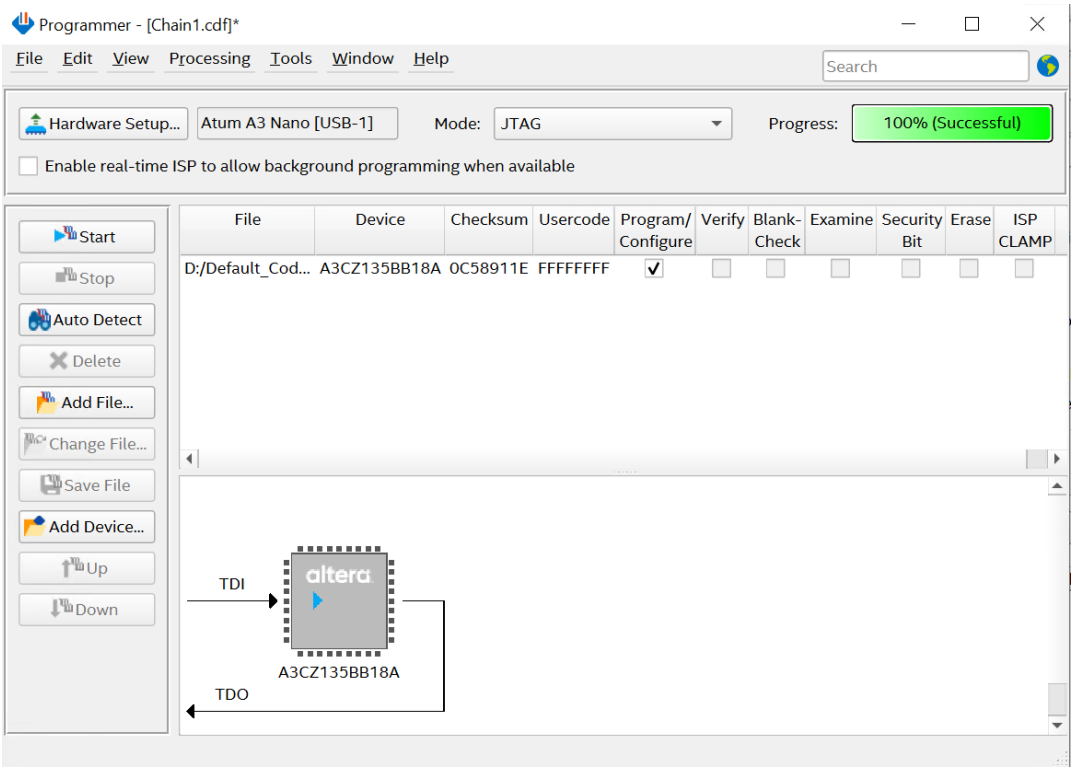

Figure 3-9 Program .sof file successfully

3.3 Board Status Elements

In addition to the four LEDs that the FPGA can control, there are three indicators that indicate the

board status (see Figure 3-10). Table 3-3 lists the details.

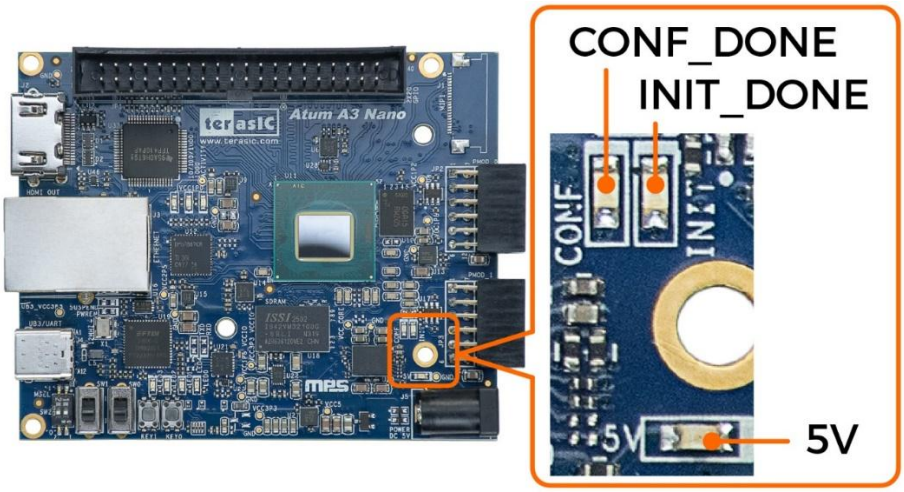

Figure 3-10 LED Indicators on the Atum A3 Nano

Table 3-3 LED Indicators

Board Reference	*LED Name*	*Description*
D12	5V	Illuminates when 5V power is active.
D6	INIT_DONE	Illuminates when the board initialization process finished for user model.
D7	CONF_DONE	Illuminates when the FPGA is successfully configured.

3.4 Clock Circuitry

Figure 3-11 shows the default frequency of all external clocks fed to the Agilex 3 FPGA. Two programmable BAW clock generators are used to distribute clock signals with low jitter. The four 50MHz clock signals connected to the FPGA are used as clock sources for user logic. A 125MHz clock is used for FPGA configuration bank (OSC_CLK1). For peripheral devices, one 25MHz clock is fed to the clock input of Gigabit Ethernet PHY. The pin assignments of clock inputs to FPGA I/O pins are listed in Table 3-4.

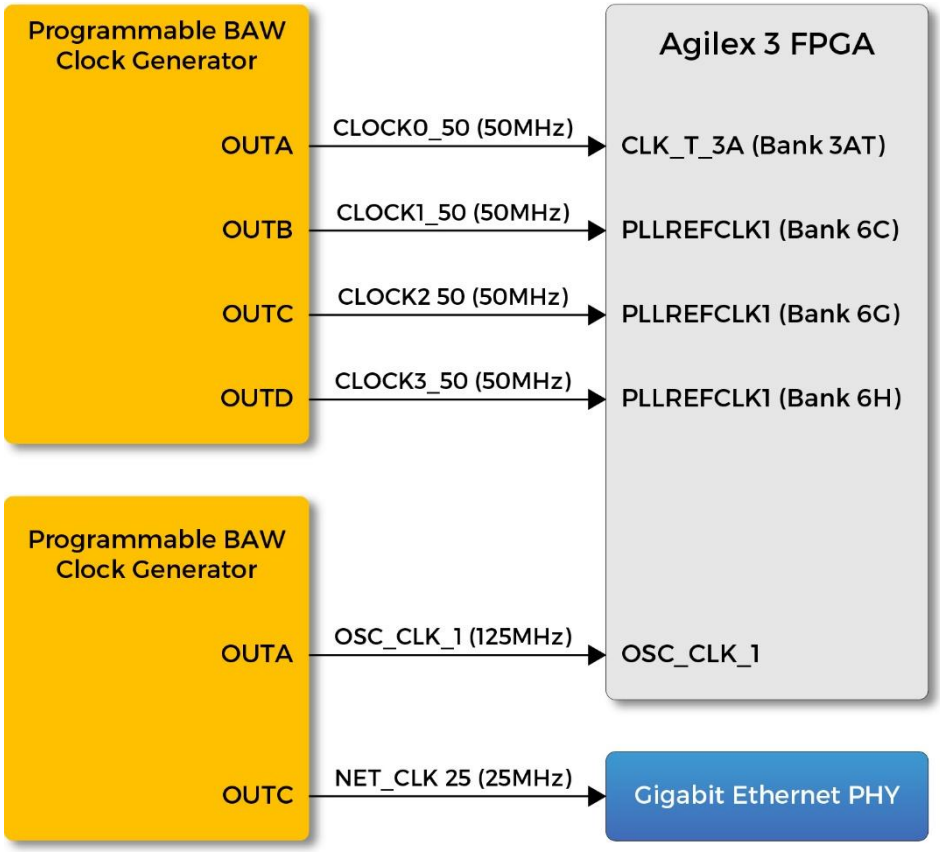

Figure 3-11 Block diagram of the clock distribution on the Atum A3 Nano

Table 3-4 Pin Assignment of Clock Inputs

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
CLOCK0_50	PIN_K43	50 MHz clock input	1.2V
CLOCK1_50	PIN_A8	50 MHz clock input	3.3-V LVCMOS
CLOCK2_50	PIN_AH9	50 MHz clock input	3.3-V LVCMOS
CLOCK3_50	PIN_R2	50 MHz clock input	3.3-V LVCMOS

3.5 User Push-buttons, Switches and LEDs

The board has two push-buttons connected to the FPGA, as shown in Figure 3-12. A Schmitt trigger circuit acts as a switch debouncer for the connected buttons. The two buttons named KEY0 and KEY1 coming out of the Schmitt trigger device are connected directly to the Agilex 3 FPGA. Each push-button generates a low logic level when it is pressed (Active low). Since the push-buttons are debounced, they can be used as reset inputs in a circuit.

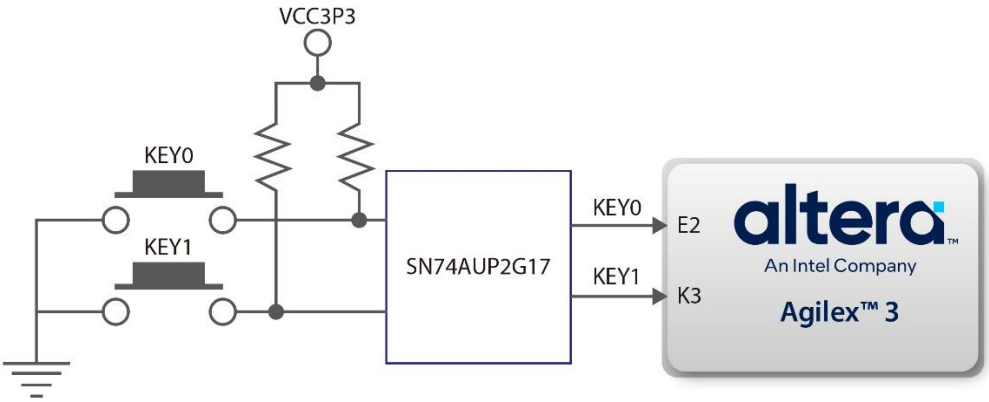

Figure 3-12 Connections between the push-buttons and the Agilex 3 FPGA

There are two slide switches connected to the Agilex 3 FPGA, as shown in Figure 3-13. These switches are not debounced and may be used as level-sensitive data inputs to a circuit. Each switch is connected directly and individually to the FPGA. When the switch is set to the DOWN position (towards the edge of the board), it sends a low logic level to the FPGA. When the switch is set to the UP position, a high logic level is sent to the FPGA.

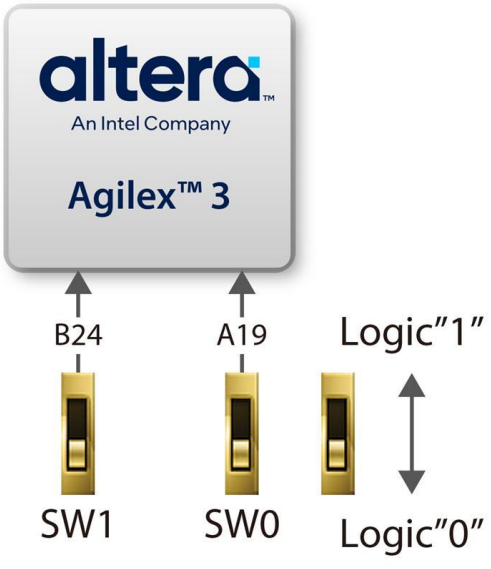

Figure 3-13 Connections between the slide switches and the Agilex 3 FPGA

There are also four user-controllable LEDs connected to the FPGA. Each LED is driven directly and individually by the Agilex 3 FPGA; driving its associated pin to a "**low**" logic level turn the LED "**on**". Figure 3-14 shows the connections between LEDs and Agilex 3 FPGA. Table 3-5, Table 3-6 and Table 3-7 list the pin assignment of user push-buttons, switches, and LEDs.

Figure 3-14 Connections between the LEDs and the Agilex 3 FPGA

Table 3-5 Pin Assignments of Slide Switches

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
SW[0]	PIN_A19	Slide Switch[0]	1.2V
SW[1]	PIN_B24	Slide Switch[1]	1.2V

Table 3-6 Pin Assignments of Push-buttons

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
KEY[0]	PIN_E2	Push-button[0]	3.3V
KEY[1]	PIN_K3	Push-button[1]	3.3V

Table 3-7 Pin Assignments of LEDs

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
LED[0]	PIN_AG2	LED [0]	3.3V
LED[1]	PIN_AM6	LED [1]	3.3V
LED[2]	PIN_AF1	LED [2]	3.3V
LED[3]	PIN_AF2	LED [3]	3.3V

3.6 2x20 GPIO Expansion Header

The board has one 40-pin expansion header. The header has 36 user pins connected directly to the Agilex 3 FPGA. It also includes DC +5V (VCC5), DC +3.3V (VCC3P3), and two GND pins. The maximum power consumption allowed for a daughter card connected to one GPIO ports is shown in Table 3-8.

Table 3-8 Voltage and Max. Current Limit of Expansion Header

Supplied Voltage	*Max. Current Limit*
5V	1A
3.3V	1.5A

Figure 3-15 shows the I/O distribution of the GPIO connector, and the Figure 3-16 indicates the pin 1 location of the expansion headers. Table 3-9 shows the pin assignment of the GPIO header.

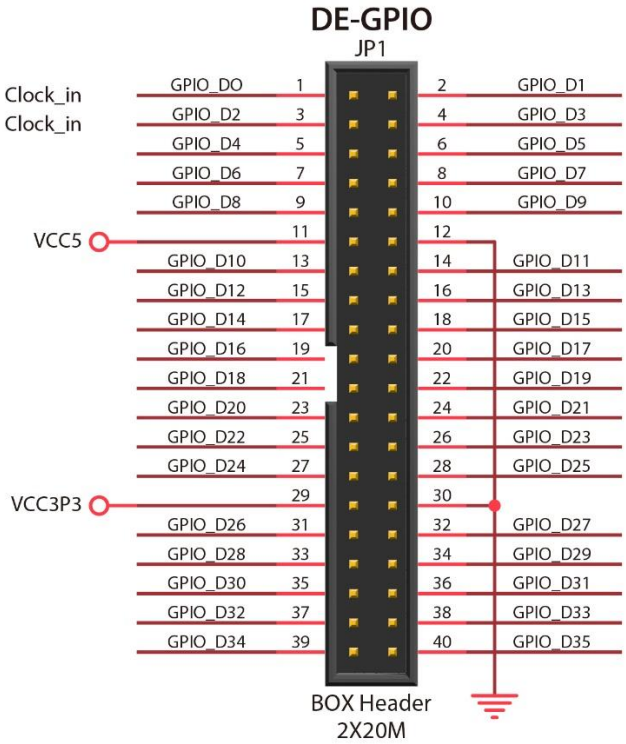

Figure 3-15 Pin arrangement of the GPIO expansion header

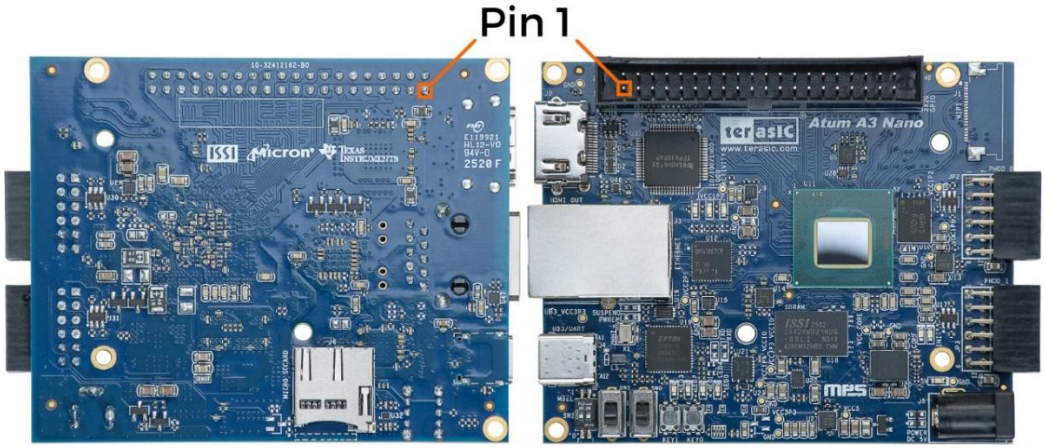

Figure 3-16 Pin1 location of the GPIO expansion header

Table 3-9 Pin Assignment of Expansion Header

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
GPIO_D[0]	PIN_B3	GPIO Connection 0[0]	3.3V
GPIO_D[1]	PIN_A11	GPIO Connection 0[1]	3.3V
GPIO_D[2]	PIN_D18	GPIO Connection 0[2]	3.3V
GPIO_D[3]	PIN_B11	GPIO Connection 0[3]	3.3V
GPIO_D[4]	PIN_H7	GPIO Connection 0[4]	3.3V
GPIO_D[5]	PIN_B5	GPIO Connection 0[5]	3.3V

GPIO_D[6]	PIN_W9	GPIO Connection 0[6]	3.3V
GPIO_D[7]	PIN_F3	GPIO Connection 0[7]	3.3V
GPIO_D[8]	PIN_U9	GPIO Connection 0[8]	3.3V
GPIO_D[9]	PIN_C2	GPIO Connection 0[9]	3.3V
GPIO_D[10]	PIN_U12	GPIO Connection 0[10]	3.3V
GPIO_D[11]	PIN_F7	GPIO Connection 0[11]	3.3V
GPIO_D[12]	PIN_P9	GPIO Connection 0[12]	3.3V
GPIO_D[13]	PIN_K7	GPIO Connection 0[13]	3.3V
GPIO_D[14]	PIN_K10	GPIO Connection 0[14]	3.3V
GPIO_D[15]	PIN_K13	GPIO Connection 0[15]	3.3V
GPIO_D[16]	PIN_D3	GPIO Connection 0[16]	3.3V
GPIO_D[17]	PIN_H13	GPIO Connection 0[17]	3.3V
GPIO_D[18]	PIN_B8	GPIO Connection 0[18]	3.3V
GPIO_D[19]	PIN_F10	GPIO Connection 0[19]	3.3V
GPIO_D[20]	PIN_A9	GPIO Connection 0[20]	3.3V
GPIO_D[21]	PIN_D10	GPIO Connection 0[21]	3.3V
GPIO_D[22]	PIN_F13	GPIO Connection 0[22]	3.3V
GPIO_D[23]	PIN_K18	GPIO Connection 0[23]	3.3V
GPIO_D[24]	PIN_F18	GPIO Connection 0[24]	3.3V
GPIO_D[25]	PIN_H21	GPIO Connection 0[25]	3.3V
GPIO_D[26]	PIN_D26	GPIO Connection 0[26]	3.3V
GPIO_D[27]	PIN_K21	GPIO Connection 0[27]	3.3V
GPIO_D[28]	PIN_H27	GPIO Connection 0[28]	3.3V
GPIO_D[29]	PIN_F26	GPIO Connection 0[29]	3.3V
GPIO_D[30]	PIN_B14	GPIO Connection 0[30]	3.3V
GPIO_D[31]	PIN_F27	GPIO Connection 0[31]	3.3V
GPIO_D[32]	PIN_B16	GPIO Connection 0[32]	3.3V
GPIO_D[33]	PIN_A14	GPIO Connection 0[33]	3.3V
GPIO_D[34]	PIN_F21	GPIO Connection 0[34]	3.3V
GPIO_D[35]	PIN_B19	GPIO Connection 0[35]	3.3V

3.7 HDMI Output

The development board provides a high performance DVI-compliant digital transmitter via the TI TFP410, which incorporates DVI v1.0 features and supports pixel rates up to 165MHz, including 1080p and WUXGA at 60Hz. The TFP410 is controlled via a serial I2C bus interface, which is connected to pins on the Agilex 3 FPGA and output via HDMI TX interface on Atum A3 Nano board. A schematic diagram of the circuitry is shown in Figure 3-17. Detailed information on using the TFP410 is available on the manufacturer's website, or under the Datasheets\HDMI folder in the Atum A3 Nano System CD.

Table 3-10 lists the HDMI Interface pin assignments and signal names relative to the FPGA.

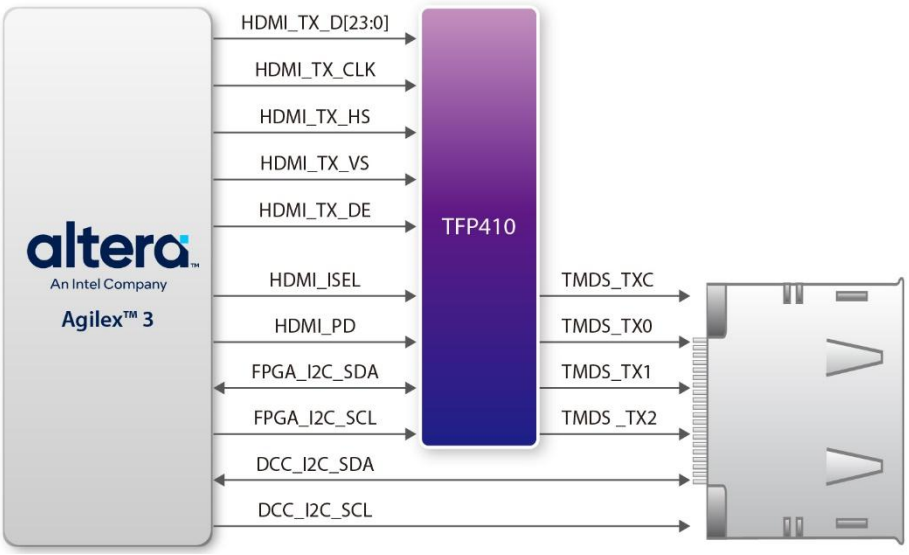

Figure 3-17 Connections between the FPGA and DVI Transmitter Chip

Table 3-10 Pin Assignment of HMDI

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
HDMI_TX_D0	PIN_G52	Video Data bus	1.2V
HDMI_TX_D1	PIN_J52	Video Data bus	1.2V
HDMI_TX_D2	PIN_G51	Video Data bus	1.2V
HDMI_TX_D3	PIN_J51	Video Data bus	1.2V
HDMI_TX_D4	PIN_M51	Video Data bus	1.2V
HDMI_TX_D5	PIN_H43	Video Data bus	1.2V
HDMI_TX_D6	PIN_A34	Video Data bus	1.2V
HDMI_TX_D7	PIN_A29	Video Data bus	1.2V
HDMI_TX_D8	PIN_B29	Video Data bus	1.2V
HDMI_TX_D9	PIN_B27	Video Data bus	1.2V
HDMI_TX_D10	PIN_A26	Video Data bus	1.2V
HDMI_TX_D11	PIN_B26	Video Data bus	1.2V
HDMI_TX_D12	PIN_A31	Video Data bus	1.2V
HDMI_TX_D13	PIN_H35	Video Data bus	1.2V
HDMI_TX_D14	PIN_A37	Video Data bus	1.2V
HDMI_TX_D15	PIN_F35	Video Data bus	1.2V
HDMI_TX_D16	PIN_F40	Video Data bus	1.2V
HDMI_TX_D17	PIN_F32	Video Data bus	1.2V
HDMI_TX_D18	PIN_V52	Video Data bus	1.2V
HDMI_TX_D19	PIN_K35	Video Data bus	1.2V
HDMI_TX_D20	PIN_V51	Video Data bus	1.2V
HDMI_TX_D21	PIN_K32	Video Data bus	1.2V
HDMI_TX_D22	PIN_K40	Video Data bus	1.2V
HDMI_TX_D23	PIN_A24	Video Data bus	1.2V
HDMI_TX_CLK	PIN_L51	HDMI transmitter clock	DIFFERENTIAL 1.2V SSTL
HDMI_TX_DE	PIN_R51	Data Enable Signal for Digital Video.	3.3V
HDMI_TX_HS	PIN_R52	Horizontal Synchronization	3.3V

HDMI_TX_VS	PIN_N52	Vertical Synchronization	3.3V
HDMI_TX_ISEL	PIN_J2	I²C interface select/I²C reset	3.3V LVCMOS
HDMI_PD	PIN_J1	Power down (active low)	3.3V LVCMOS
DDC_I2C_SCL	PIN_AV2	Serial Port Data Clock to Sink	3.3V LVCMOS
DDC_I2C_SDA	PIN_AN1	Serial Port Data Input/Output to Sink	3.3V LVCMOS
HDMI_I2C_SCL	PIN_M2	FPGA I2C Clock	3.3V LVCMOS
HDMI_I2C_SDA	PIN_N2	FPGA I2C Data	3.3V LVCMOS

3.8 TMD Header

The Atum A3 Nano board provides two 2x6 TMD (Terasic Mini Digital) expansion headers which are compatible with PMOD. Each TMD header has 8 digital GPIO user pins connected to the Agilex 3 FPGA, two 3.3V power pins and two GND pins.There are two Transient Voltage Suppressor diode arrays for each TMD header, they are used to implement ESD protection for the GPIO user pins.

Figure 3-18 shows the connections between the two TMD headers and the Agilex 3 FPGA. Table 3-11 lists the pin assignments of the headers.

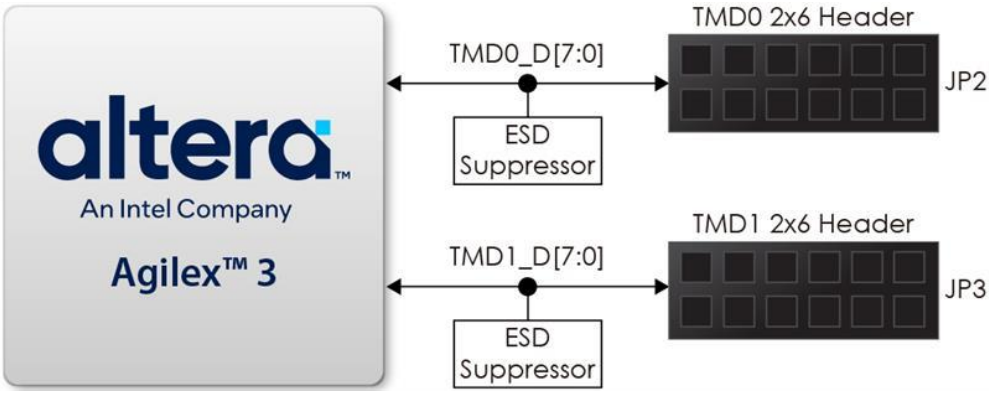

Figure 3-18 Connections between FPGA and TMD headers

Table 3-11 Pin Assignment of TMD headers

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
TMD0_D[0]	PIN_AD6	GPIO Connection [0]	3.3-V LVCMOS
TMD0_D[1]	PIN_P6	GPIO Connection [1]	3.3-V LVCMOS
TMD0_D[2]	PIN_AB9	GPIO Connection [2]	3.3-V LVCMOS
TMD0_D[3]	PIN_AE12	GPIO Connection [3]	3.3-V LVCMOS
TMD0_D[4]	PIN_P4	GPIO Connection [4]	3.3-V LVCMOS
TMD0_D[5]	PIN_AA4	GPIO Connection [5]	3.3-V LVCMOS
TMD0_D[6]	PIN_AH12	GPIO Connection [6]	3.3-V LVCMOS
TMD0_D[7]	PIN_AA6	GPIO Connection [7]	3.3-V LVCMOS
TMD1_D[0]	PIN_AR2	GPIO Connection [0]	3.3-V LVCMOS
TMD1_D[1]	PIN_AJ6	GPIO Connection [1]	3.3-V LVCMOS
TMD1_D[2]	PIN_AK1	GPIO Connection [2]	3.3-V LVCMOS
TMD1_D[3]	PIN_AR1	GPIO Connection [3]	3.3-V LVCMOS
TMD1_D[4]	PIN_AL12	GPIO Connection [4]	3.3-V LVCMOS

TMD1_D[5]	PIN_AJ4	GPIO Connection [5]	3.3-V LVCMOS
TMD1_D[6]	PIN_AV1	GPIO Connection [6]	3.3-V LVCMOS
TMD1_D[7]	PIN_AK2	GPIO Connection [7]	3.3-V LVCMOS

3.9 SDRAM Memory

The board features 64MB of SDRAM with a single 64MB (16M×32) SDRAM chip. The chip connects to the FPGA with 32 data lines, 13 address lines and control lines. This chip uses the 1.8V LVCMOS signaling standard. Connections between the FPGA and SDRAM are shown in Figure 3-19, and the pin assignment is listed in Table 3-12.

Figure 3-19 Connections between the FPGA and SDRAM

Table 3-12 Pin Assignment of SDRAM

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
DRAM_ADDR[0]	PIN_BN26	SDRAM Address[0]	1.8-V LVCMOS
DRAM_ADDR[1]	PIN_BH27	SDRAM Address[1]	1.8-V LVCMOS
DRAM_ADDR[2]	PIN_BM26	SDRAM Address[2]	1.8-V LVCMOS
DRAM_ADDR[3]	PIN_BH21	SDRAM Address[3]	1.8-V LVCMOS
DRAM_ADDR[4]	PIN_BH26	SDRAM Address[4]	1.8-V LVCMOS
DRAM_ADDR[5]	PIN_BH18	SDRAM Address[5]	1.8-V LVCMOS
DRAM_ADDR[6]	PIN_BK18	SDRAM Address[6]	1.8-V LVCMOS
DRAM_ADDR[7]	PIN_BH32	SDRAM Address[7]	1.8-V LVCMOS
DRAM_ADDR[8]	PIN_BM24	SDRAM Address[8]	1.8-V LVCMOS
DRAM_ADDR[9]	PIN_BH35	SDRAM Address[9]	1.8-V LVCMOS
DRAM_ADDR[10]	PIN_BN27	SDRAM Address[10]	1.8-V LVCMOS
DRAM_ADDR[11]	PIN_BN29	SDRAM Address[11]	1.8-V LVCMOS
DRAM_ADDR[12]	PIN_BK26	SDRAM Address[12]	1.8-V LVCMOS
DRAM_DQ[0]	PIN_BM50	SDRAM Data[0]	1.8-V LVCMOS
DRAM_DQ[1]	PIN_BM51	SDRAM Data[1]	1.8-V LVCMOS
DRAM_DQ[2]	PIN_BN47	SDRAM Data[2]	1.8-V LVCMOS
DRAM_DQ[3]	PIN_BM47	SDRAM Data[3]	1.8-V LVCMOS

DRAM_DQ[4]	PIN_BL51	SDRAM Data[4]	1.8-V LVCMOS
DRAM_DQ[5]	PIN_BH50	SDRAM Data[5]	1.8-V LVCMOS
DRAM_DQ[6]	PIN_BK50	SDRAM Data[6]	1.8-V LVCMOS
DRAM_DQ[7]	PIN_BH46	SDRAM Data[7]	1.8-V LVCMOS
DRAM_DQ[8]	PIN_BM37	SDRAM Data[8]	1.8-V LVCMOS
DRAM_DQ[9]	PIN_BN37	SDRAM Data[9]	1.8-V LVCMOS
DRAM_DQ[10]	PIN_BM42	SDRAM Data[10]	1.8-V LVCMOS
DRAM_DQ[11]	PIN_BN42	SDRAM Data[11]	1.8-V LVCMOS
DRAM_DQ[12]	PIN_BM44	SDRAM Data[12]	1.8-V LVCMOS
DRAM_DQ[13]	PIN_BN45	SDRAM Data[13]	1.8-V LVCMOS
DRAM_DQ[14]	PIN_BN39	SDRAM Data[14]	1.8-V LVCMOS
DRAM_DQ[15]	PIN_BM45	SDRAM Data[15]	1.8-V LVCMOS
DRAM_DQ[16]	PIN_BG2	SDRAM Data[16]	1.8-V LVCMOS
DRAM_DQ[17]	PIN_BA2	SDRAM Data[17]	1.8-V LVCMOS
DRAM_DQ[18]	PIN_BC2	SDRAM Data[18]	1.8-V LVCMOS
DRAM_DQ[19]	PIN_BJ1	SDRAM Data[19]	1.8-V LVCMOS
DRAM_DQ[20]	PIN_BE4	SDRAM Data[20]	1.8-V LVCMOS
DRAM_DQ[21]	PIN_BG1	SDRAM Data[21]	1.8-V LVCMOS
DRAM_DQ[22]	PIN_BC1	SDRAM Data[22]	1.8-V LVCMOS
DRAM_DQ[23]	PIN_BD1	SDRAM Data[23]	1.8-V LVCMOS
DRAM_DQ[24]	PIN_BN5	SDRAM Data[24]	1.8-V LVCMOS
DRAM_DQ[25]	PIN_BM9	SDRAM Data[25]	1.8-V LVCMOS
DRAM_DQ[26]	PIN_BM8	SDRAM Data[26]	1.8-V LVCMOS
DRAM_DQ[27]	PIN_BN11	SDRAM Data[27]	1.8-V LVCMOS
DRAM_DQ[28]	PIN_BN8	SDRAM Data[28]	1.8-V LVCMOS
DRAM_DQ[29]	PIN_BN14	SDRAM Data[29]	1.8-V LVCMOS
DRAM_DQ[30]	PIN_BM11	SDRAM Data[30]	1.8-V LVCMOS
DRAM_DQ[31]	PIN_BN16	SDRAM Data[31]	1.8-V LVCMOS
DRAM_BA[0]	PIN_BH43	SDRAM Bank Address[0]	1.8-V LVCMOS
DRAM_BA[1]	PIN_BM29	SDRAM Bank Address[1]	1.8-V LVCMOS
DRAM_DQM[0]	PIN_BE6	SDRAM byte Data Mask[0]	1.8-V LVCMOS
DRAM_DQM[1]	PIN_BM19	SDRAM byte Data Mask[1]	1.8-V LVCMOS
DRAM_DQM[2]	PIN_BJ2	SDRAM byte Data Mask[2]	1.8-V LVCMOS
DRAM_DQM[3]	PIN_BM14	SDRAM byte Data Mask[3]	1.8-V LVCMOS
DRAM_RAS_N	PIN_BM31	SDRAM Row Address Strobe	1.8-V LVCMOS
DRAM_CAS_N	PIN_BH40	SDRAM Column Address Strobe	1.8-V LVCMOS
DRAM_CKE	PIN_BK40	SDRAM Clock Enable	1.8-V LVCMOS
DRAM_CLK	PIN_BK32	SDRAM Clock	1.8-V LVCMOS
DRAM_WE_N	PIN_BM34	SDRAM Write Enable	1.8-V LVCMOS
DRAM_CS_N	PIN_BN34	SDRAM Chip Select	1.8-V LVCMOS

3.10 Gigabit Ethernet

The board supports Gigabit Ethernet communication through an external TI DP83867CR PHY chip. The DP83867CR chip with integrated 10/100/1000 Mbps Gigabit Ethernet transceiver only supports

an RGMII MAC interface. Figure 3-20 shows the connections between the FPGA, Gigabit Ethernet PHY, and RJ-45 connector.

The pin assignment associated with the Gigabit Ethernet interface is listed in Table 3-13. More information about the DP83867CR PHY chip and its datasheet, as well as the application notes, is available on the manufacturer's website.

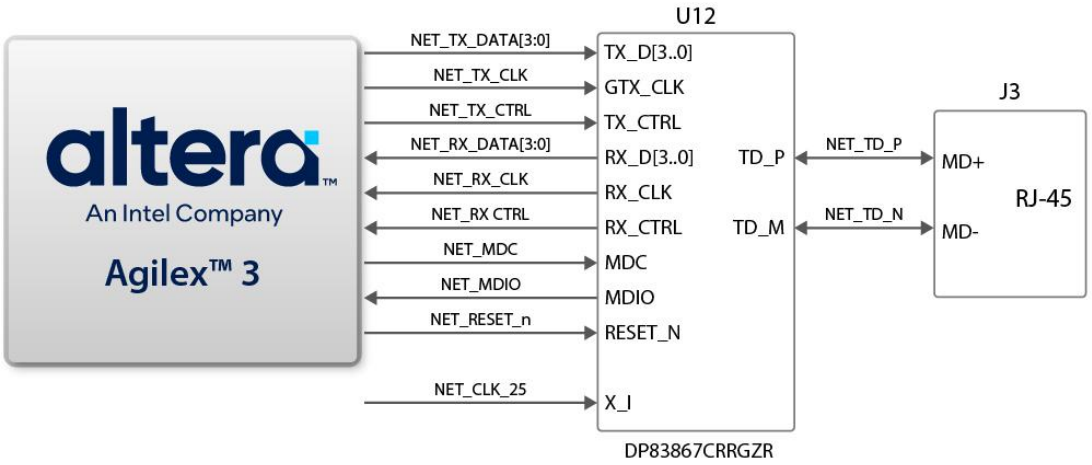

Figure 3-20 Connections between the FPGA and Gigabit Ethernet

Table 3-13 Pin Assignment of Gigabit Ethernet PHY

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
NET_TX_CTRL	PIN_AU6	RGMII transmit enable	1.8-V LVCMOS
NET_TX_DATA[0]	PIN_BH10	RGMII transmit data[0]	1.8-V LVCMOS
NET_TX_DATA[1]	PIN_BK10	RGMII transmit data[1]	1.8-V LVCMOS
NET_TX_DATA[2]	PIN_BM2	RGMII transmit data[2]	1.8-V LVCMOS
NET_TX_DATA[3]	PIN_BM3	RGMII transmit data[3]	1.8-V LVCMOS
NET_TX_CLK	PIN_AY6	RGMII transmit clock	1.8-V LVCMOS
NET_RX_CTRL	PIN_AU4	RGMII receive data valid	1.8-V LVCMOS
NET_RX_DATA[0]	PIN_BF12	RGMII receive data[0]	1.8-V LVCMOS
NET_RX_DATA[1]	PIN_BF9	RGMII receive data[1]	1.8-V LVCMOS
NET_RX_DATA[2]	PIN_BB12	RGMII receive data[2]	1.8-V LVCMOS
NET_RX_DATA[3]	PIN_BH3	RGMII receive data[3]	1.8-V LVCMOS
NET_RX_CLK	PIN_AT9	RGMII receive clock	1.8-V LVCMOS
NET_MDIO	PIN_BH7	Management Data	1.8-V LVCMOS
NET_MDC	PIN_BB9	Management Data Clock Reference	1.8-V LVCMOS
NET_RESET_n	PIN_AW12	Active low RESET initializes the PHY	1.8-V LVCMOS

There are three LEDs, blue LED (LEDB), green LED (LEDG) and yellow LED (LEDY), which report the status of Ethernet PHY (DP83867CR). The LED control signals are connected to Ethernet PHY and they are configurable. On Atum A3 Nano board, the three LEDs of DP83867CR are configured to the model and function listed in Table 3-14.

Table 3-14 State and Definition of LED Mode Pins

LED	Model	Function
LEDY	1	Port Mirror Disable, indicates the link is established
LEDB	2	ANEG_SEL = 0, Auto-Neg All Speed RGMII Clock Skew TX[2]= 1, indicates 1000BASE-T link is established
LEDG	1	RGMII Clock Skew TX[1:0]= 0x00, indicates receive or transmit activity

3.11 Micro SD Card Socket

The board supports a Micro SD card interface with 4 data lines. It serves as an external storage for the Atum A3 Nano board. Figure 3-21 shows the signals connected between the FPGA and the Micro SD card socket.

Table 3-15 lists the pin assignment of Micro SD card socket to the FPGA.

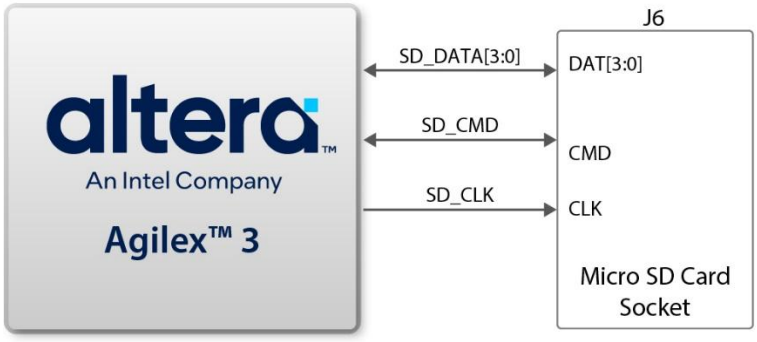

Figure 3-21 Connections between the FPGA and SD card socket

Table 3-15 Pin Assignment of Micro SD Card Socket

Signal Name	FPGA Pin No.	Description	I/O Standard
SD_CLK	PIN_Y1	SD Clock	3.3-V LVCMOS
SD_CMD	PIN_N1	SD Command Line	3.3-V LVCMOS
SD_DATA[0]	PIN_AC2	SD Data[0]	3.3-V LVCMOS
SD_DATA[1]	PIN_AC1	SD Data[1]	3.3-V LVCMOS
SD_DATA[2]	PIN_V2	SD Data[2]	3.3-V LVCMOS
SD_DATA[3]	PIN_V1	SD Data[3]	3.3-V LVCMOS

3.12 UART

The board provides a UART interface for users to communicate and transfer data with the Agilex 3 FPGA through the host. The interface is implemented via the FT2232H chip in the USB Blaster III circuit.

Connecting a USB cable between the Atum A3 Nano board's Type-C connector and the host enables both USB Blaster III and FPGA UART functions. Serial driver isn't required, users need to make sure the USB Blaster III driver is installed before using the UART function. After connecting the USB

cable, normally USB Blaster III and a COM port number will both show in PC Device Manager.

Figure 3-22 shows the connections between the Agilex 3 FPGA, FT2232H chip and the USB Type-C connector. Table 3-16 lists the pin assignments of the UART interface connected to the FPGA.

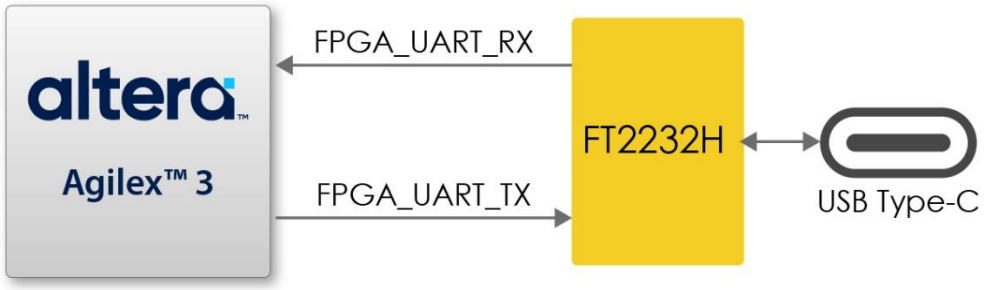

Figure 3-22 Connections between the FPGA and Type-C UART connector

Table 3-16 Pin Assignment of UART

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
FPGA_UART_TX	PIN_T6	UART Transmitter	3.3-V LVCMOS
FPGA_UART_RX	PIN_L1	UART Receiver	3.3-V LVCMOS

Chapter 4

Examples of Advanced Designs

This chapter provides examples of advanced designs implemented by RTL or Qsys on the **Atum A3 Nano board**. These reference designs cover the features of peripherals connected to the FPGA, such as SDRAM, HDMI, Micro SD card and UART. All the associated files can be found in the directory **\Demonstrations** of Atum A3 Nano System CD.

■ Installation of Demonstrations

To install the demonstrations on your computer:

Copy the folder Demonstrations to a local directory of your choice. It is important to make sure the path to your local directory contains NO space. Otherwise, it will lead to error in Nios.

Note: Quartus Pro v25.1 is required for all Atum A3 Nano demonstrations, including the Agilex 3 FPGA device.

4.1 Atum A3 Nano Factory Configuration

The Atum A3 Nano board has a default configuration bitstream pre-programmed, which demonstrates some of the basic features on board. The setup required for this demonstration and the location of its files are shown below.

■ Demonstration Setup, File Locations, and Instructions

- Project directory: \Default_Code
- Bitstream used: golden_top.sof or golden_top.jic
- Demo batch file : \Default_Code\demo_batch\test.bat
- Use Type-C USB cable to connected the board to PC, use HDMI cable to connect the board to HDMI monitor, then power on the Atum A3 Nano board. If necessary (that is, if the default factory configuration is not currently stored in the QSPI device), download the bit stream to the board via JTAG interface.
- You should now be able to observe the four user LEDs are blinking, and the HDMI monitor displays the board image.
- The test.bat is executed to program .sof to the FPGA, the flash_program.bat is executed to program the .jic to the flash, the convert.bat is executed to convert .sof to .jic file then program the .jic to the flash.
- If users want to program a new design into the QSPI flash device, the easiest method is to

copy the new .sof file to the demo_batch folder and execute the convert.bat to program the flash device.

4.2 SDRAM Test in Verilog

Atum A3 Nano system CD offers another SDRAM test with its test code written in Verilog HDL. The memory size of the SDRAM is 16Mx32bit.

■ System Block Diagram

Figure 4-1 shows the function block diagram of this demonstration. The SDRAM controller uses 50 MHz as a reference clock and generates 125MHz as the memory clock.

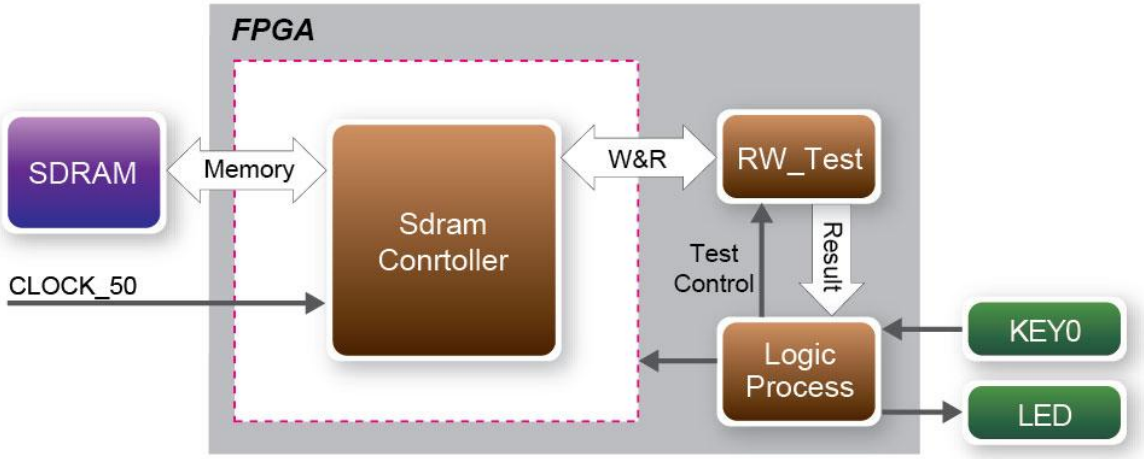

Figure 4-1 Block Diagram of the SDRAM test in Verilog

RW_test module writes the entire memory with a test sequence first before comparing the data read back with the regenerated test sequence, which is same as the data written to the memory. KEY0 triggers test control signals for the SDRAM, and the LEDs will indicate the test result.

■ Design Tools

- Quartus Prime 25.1 Pro Edition

■ Demonstration Source Code

- Quartus Project directory: SDRAM_RTL_test
- Bitstream used: golden_top.sof

■ Demonstration Batch File

Demo Batch File Folder: SDRAM_RTL_test\demo_batch
The demo batch file includes following files:

- Demo Batch File : test.bat
- FPGA Configure File: golden_top.sof

■ Demonstration Setup and Instructions

- Please make sure both Quartus II and UBIII driver are installed on the host PC.
- Power on the board.
- Execute the demo batch file " test.bat" from the directory SDRAM_RTL_Test\demo_batch.
- Press KEY0 on the board to start the verification process. When KEY0 is pressed, the LED [2:0] should turn on.Then release KEY0, LED1 and LED0 should start blinking.
- After approximately 6 seconds, LED0 should stop blinking and stay ON to indicate the test is PASS. Table 4-1 lists the status of LED indicators.
- If LED1 is not blinking, it means 50MHz clock source is not working.
- Press KEY0 again to repeat the SDRAM test.

Table 4-1 Status of LED Indicators

Name	Description
LED2	Reset
LED1	Blinks
LED0	ON if the test is PASS after releasing KEY0

4.3 HDMI_out RTL in Verilog

Atum A3 Nano system CD offers another HDMI(DVI 1.0-compliant) test with its test code written in Verilog HDL. That is, the output test for the TFP410 IC includes I2C configuration and the ability to output 1080p@60 with the specified Color Par.

■ System Block Diagram

Figure 4-2 shows the function block diagram of this demonstration. The vpg(vedio pattern generator) uses 50MHz in by PLL to generates 148.5MHz to do input The VPG generates a 1080p@60 video timing color bar, which is sent to the TFP410 and then displayed on an HDMI monitor.

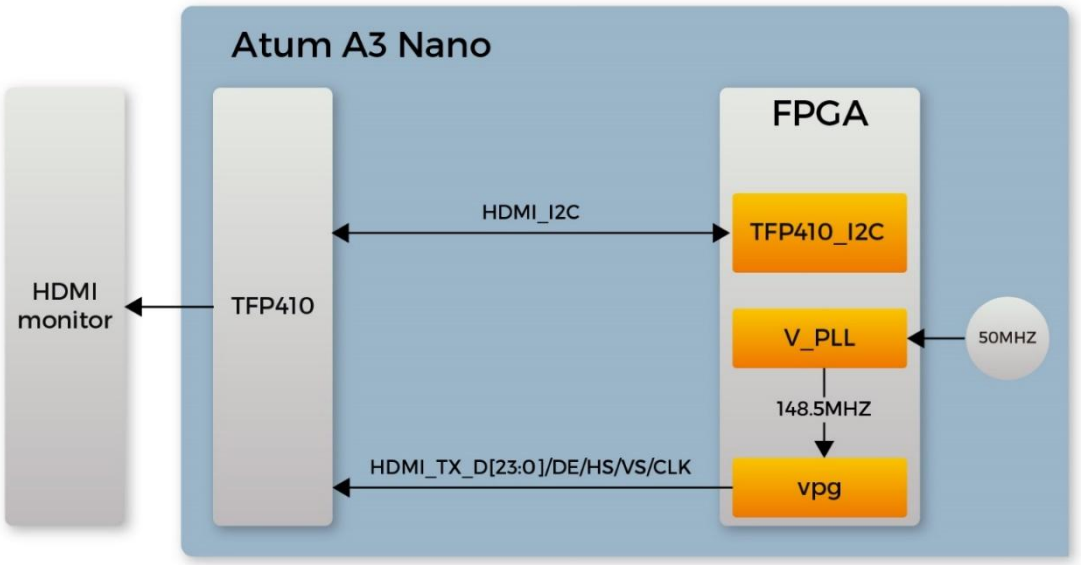

Figure 4-2 Block Diagram of the HDMI_out_RTL in Verilog

For the TFP410 to output video, the TFP410_I2C module must be used to configure the device via I2C by setting bit 0 of register 0x08 (CTL_1_MODE) to 1, enabling normal operation.

■ Design Tools

- Quartus Prime 25.1 Pro Edition

■ Demonstration Source Code

- Quartus Project directory: HDMI_out_RTL
- Bitstream used: golden_top.sof

■ Demonstration Batch File

Demo Batch File Folder: HDMI_out_RTL\demo_batch
The demo batch file includes following files:

- Demo Batch File : test.bat
- FPGA Configure File: golden_top.sof

■ Demonstration Setup and Instructions

- Please make sure both Quartus Pro and UBIII driver are installed on the host PC.
- Connect the HDMI cable to J2 (HDMI Out) on the Atum A3 Nano.
- Power on the board.
- Execute the demo batch file “ test.bat” from the directory
- HDMI_out_RTL\demo_batch.
- 1080p@60 color bar should be visible on the monitor.

4.4 UART Test in Verilog

Atum A3 Nano system CD offers UART loopback test code written in Verilog HDL.

■ System Block Diagram

Figure 4-3 shows the function block diagram of this demonstration. As shown in the block diagram, FPGA_UART_TX is the UART signal output from the FT2232 on the Atum A3 Nano and input to the FPGA. Conversely, FPGA_UART_RX is the UART signal transmitted from the FPGA and input to the FT2232 on the Atum A3 Nano. KEY0 functions as a cutoff switch for the UART loopback.

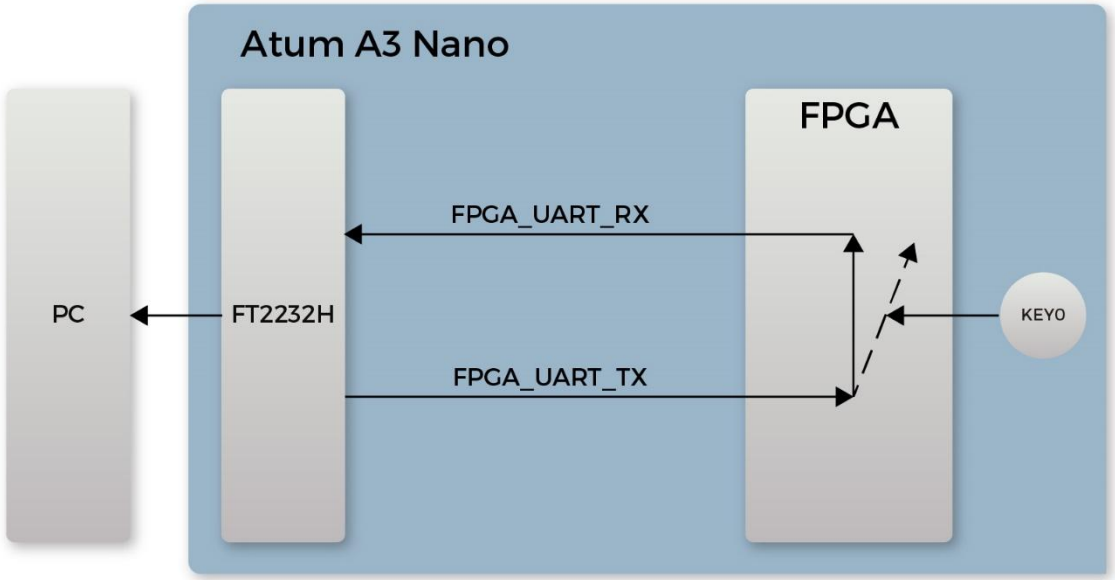

Figure 4-3 Block Diagram of the UART in Verilog

■ Design Tools

- Quartus Prime 25.1 Pro Edition

■ Demonstration Source Code

- Quartus Project directory: UART
- Bitstream used: golden_top.sof

■ Demonstration Batch File

Demo Batch File Folder: UART\demo_batch
The demo batch file includes following files:

- Demo Batch File : test.bat
- FPGA Configure File: golden_top.sof

■ Demonstration Setup and Instructions

- Please make sure both Quartus Pro and UBIII driver are installed on the host PC.
- Power on the board.
- Execute the demo batch file " test.bat" from the directory UART\demo_batch.
- Use a UART communication tool such as PuTTY on the PC. Select the correct COM port (the COM port number can be found in Windows Device Manager), set the baud rate to 115200, and open the UART terminal window. Typing characters from the PC keyboard should result in echoed characters appearing in the terminal. Press KEY0 to verify whether the UART transmission is interrupted.

4.5 SD Card Demonstration

Many applications use a large external storage device, such as an SD Card to store data. The FPGA

board provides the hardware and software needed for SD Card access. In this demonstration we will show how to read the file contents of a specific file and how to create a new file and write content into the file. The SD Card is required to be formatted as FAT32 File System in advance. Long file name is supported in this demonstration.

Figure 4-4 shows the hardware system block diagram of this demonstration. In the system platform designer, the terasic_sdio component (a Platform Designer Componen) is used to communicate with the SD Card by SDIO protocol. A Nios V processor is used to implement the filesystem. The Nios V program is running on the on-chip memory. The PLL generates a 100MHz clock for the Nios V processor and 50Mhz for the terasic_sdio component. The terasic_sdio component generate 25Mhz sd_clk. The terasic_sdio component is designed to communicate with SD card via SDIO protocol. It can execute SD command and single block (512 byte) read/write.

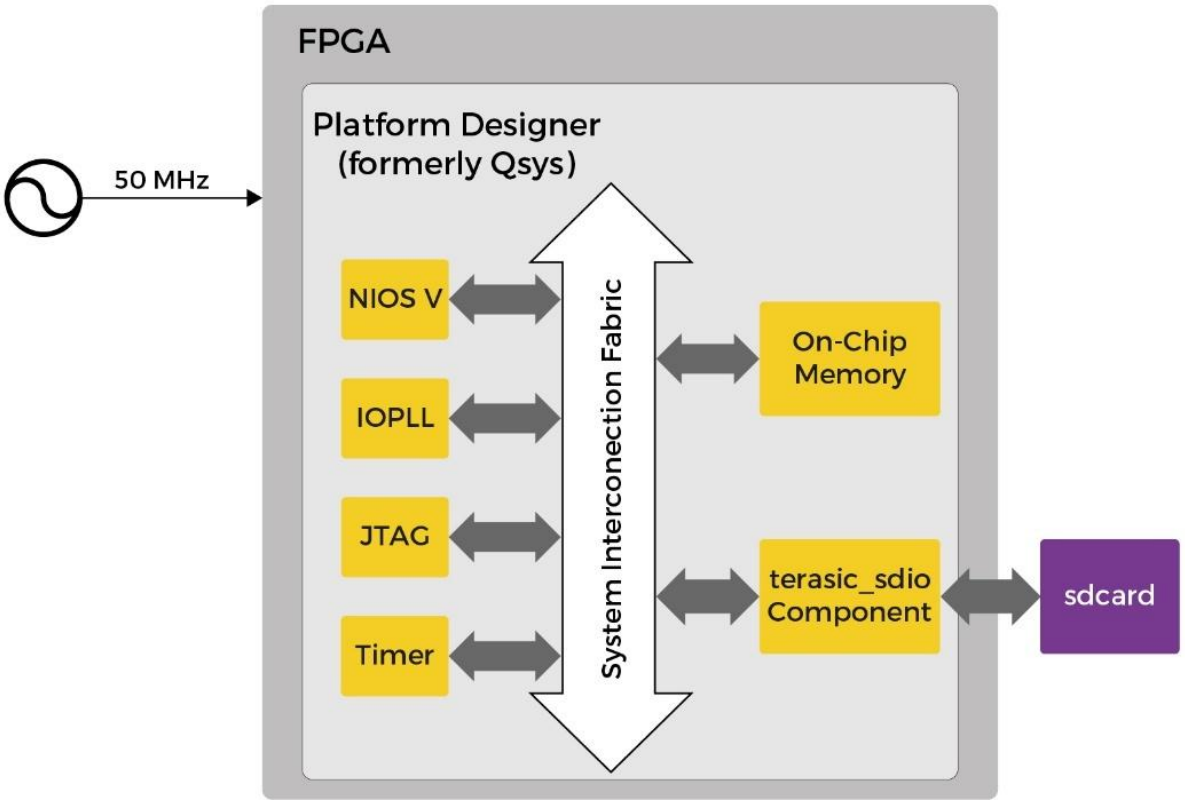

Figure 4-4 Block diagram of SDCARD demonstration

Figure 4-5 shows the Nios V software stack in this demonstration. The SD Card Controller is design to communicate with the terasic_sdio component via AVMM interface. The FAT32 File System block implements the FAT32 file system by using the open code FatFs https://elm-chan.org/fsw/ff/00index_e.html. The glue.c is design to attach the SD Card Controller to the FatFs.

The main block implements main control of this demonstration. When the program is executed, it will try to find a file called readme.txt in the sdcard. If the file is found, it will dump the file content. Second, it will create a new file create.txt with content “Write Test!”

Figure 4-5 Nios V Software Stack of the SD Card Demonstration

■ Demonstration Source Code

- Quartus Project directory: sdcard_filesyste
- Bitstream used: golden_top.sof
- Nios V Project workspace: sdcard_filesyste/software
- Nios V bin file: nios_app.elf

■ Demonstration File Locations

Demo batch directory: \sdcard_filesystem\demo_batch
The folder includes the following files:

- Batch file: test.bat
- FPGA configuration file : golden_top.sof
- Nios V bin file : nios_app.elf

■ Demonstration Setup

Please follow below procedures to set up the demonstrations.

1. Make sure UB3 blaster driver is installed.
2. Connect the UB3 type-C port of the FPGA board to your Host PC.
3. Make sure a SD Card is installed to the SD socket. The SD Card should be formatted as FAT32 and contain a text file called as readme.txt.
4. Power on the FPGA board.
5. Execute demo_test.bat to start the demo. The demo includes
   - Find a file readme.txt and display its content.
   - Create a new file called as create.txt with content “Write Test!”
6. The demo will launch Nios terminal and show the test result as shown in **Figure 4-6**.

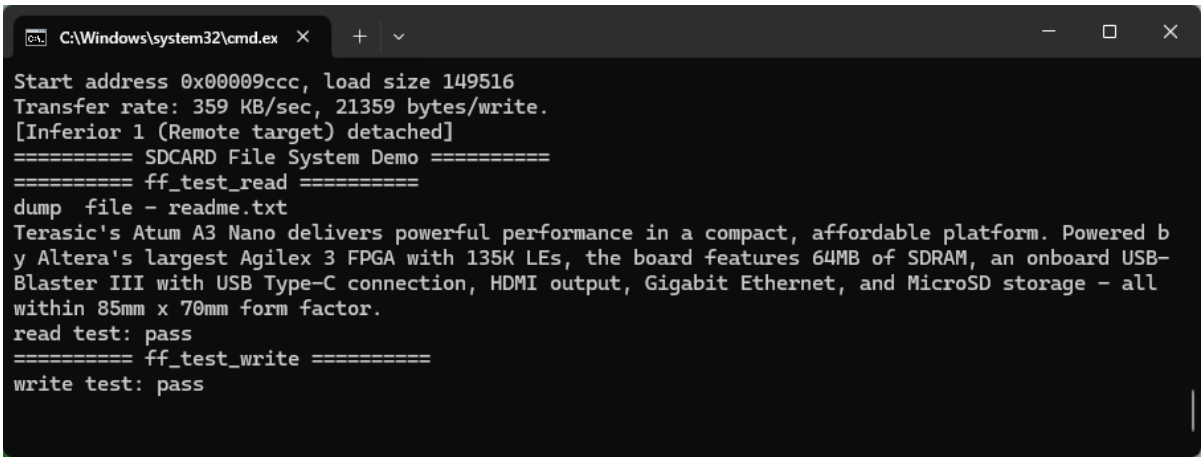

Figure 4-6 SD Card demonstration

Chapter 5

Programming the QSPI Flash

Atum A3 Nano board features an QSPI Flash memory connected to FPGA, allowing users to program their FPGA configuration files into the QSPI flash. This setup enables the FPGA's Secure Device Manager (SDM) to automatically load the configuration from QSPI Flash upon power-up. This chapter will describe the necessary tools, environment, and detailed steps involved in the programming proces.

5.1 Programming Flow

Figure 5-1 below illustrates the components/software and programming flow for the entire programming process. The flow is as follows:

1. Users need to compile their projects in Quartus Prime to generate a stream file (.sof).
2. Next, use the **Quartus Prime Programming File Generator** to convert the .sof file into a .jic file.
3. Users can use the **Quartus Prime Programmer** on the host computer and the USB Blaster circuit on the Atum A3 to transfer the .jic file from the host to the FPGA on the Atum A3.
4. The FPGA's internal Secure Device Manager (SDM) then programs the received .jic file into the QSPI Flash through the Active Serial x4 interface.

Note : If users program their own project into the QSPI flash, the factory code in the QSPI flash will be erased. If default factory functionality is required, please reprogram the factory flash file into your board.

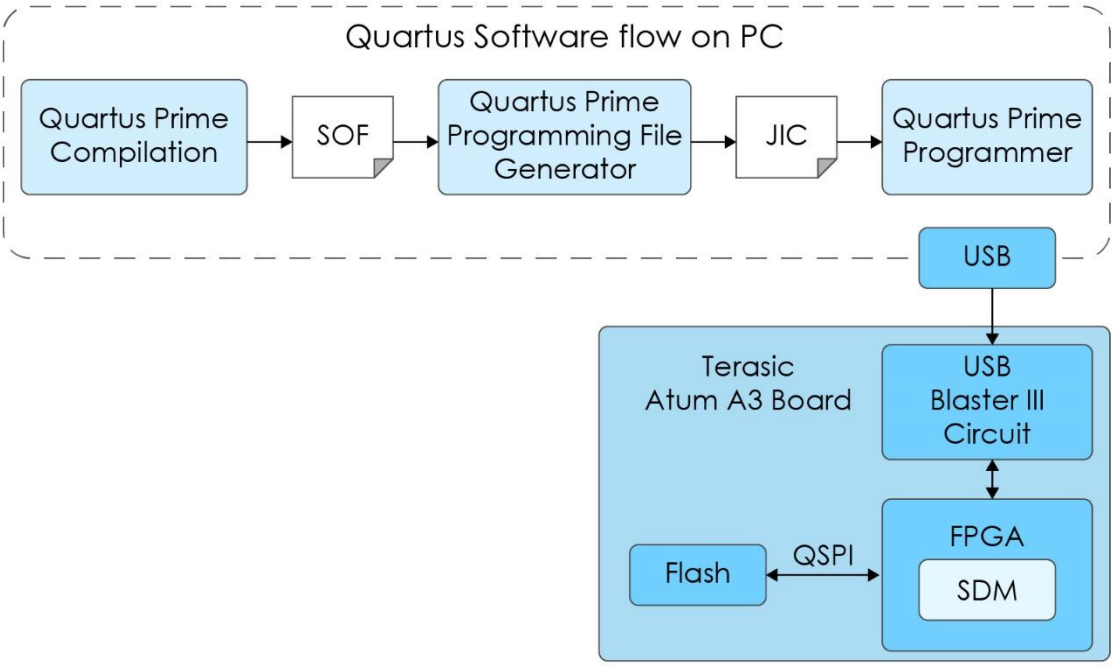

Figure 5-1 Programming Flow

5.2 Programming Steps with Batch Files

This section provides a guide on automating the conversion of your .sof (SRAM Object File) to .jic (JTAG Indirect Configuration File) format, and subsequently programming the .jic file into the QSPI flash memory, all through the use of batch files.

Terasic has pre-configured the necessary Quartus Prime commands within these batch files. This eliminates the need for users to manually launch the Quartus Prime software or input commands, significantly simplifying the programming process.

These convenient batch files are located in the following directory: System CD\Demonstrations\Default_Code\demo_batch\

For a detailed overview of the files and their corresponding descriptions within this directory, please consult Table 5-1.

Table 5-1 File Descriptions in demo_batch folder

File Name	Description
convert.bat	A batch script that automates the conversion of a .sof file to a .jic file.
flash_program.bat	A batch script designed to streamline the programming of a .jic file into the QSPI flash memory.
golden_top.jic	The .jic file generated from the conversion of golden_top.sof.
golden_top.sof	The source .sof file intended for conversion.
test.bat	A batch script used to download the .sof file directly to the FPGA.

Here are the steps for programming:

1. Copy the files in the following path to your host PC:
   System CD\Demonstrations\Default_Code\demo_batch
2. Rename the .sof file of your project to **golden_top.sof** and overwrite the existing file in demo_batch.
3. Execute **convert.bat** to convert **golden_top.sof** to **golden_top.jic**.
4. Verify that the MSEL Settings Switch on Atum A3 is set to AS mode (MSEL[2:1] = 2'b00) as shown in **Figure 5-2**.
5. Ensure that the USB blaster connector (J4) of your Atum A3 is connected to your host PC with a USB cable (see **Figure 5-3**) and the board's power is powered on.
6. Execute **flash_program.bat**, wait for the programming to complete (see **Figure 5-4**), then you can restart the board's power to check if your project runs correctly.

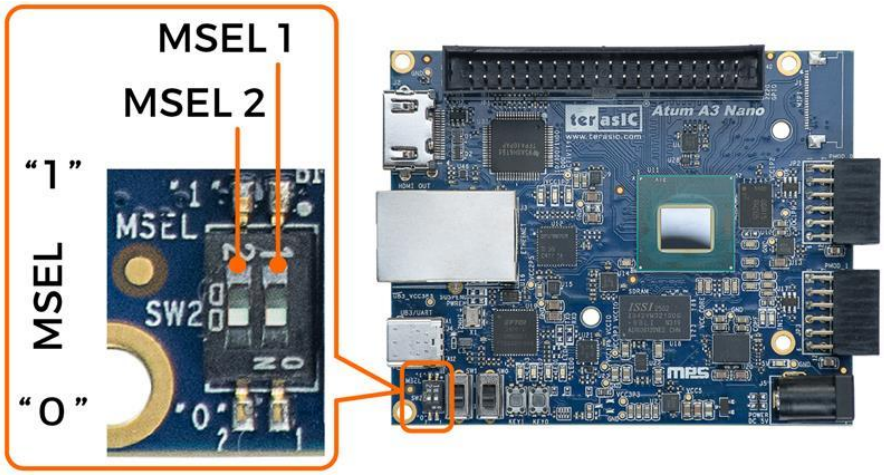

Figure 5-2 SW2 for FPGA Configuration Mode

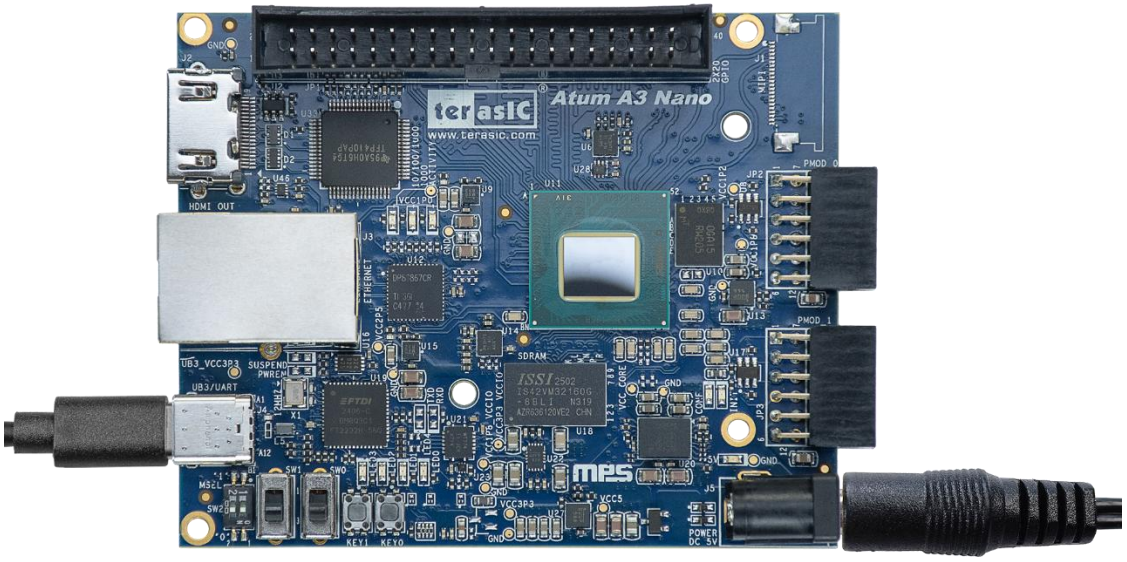

Figure 5-3 Power and USB cable connection for the board

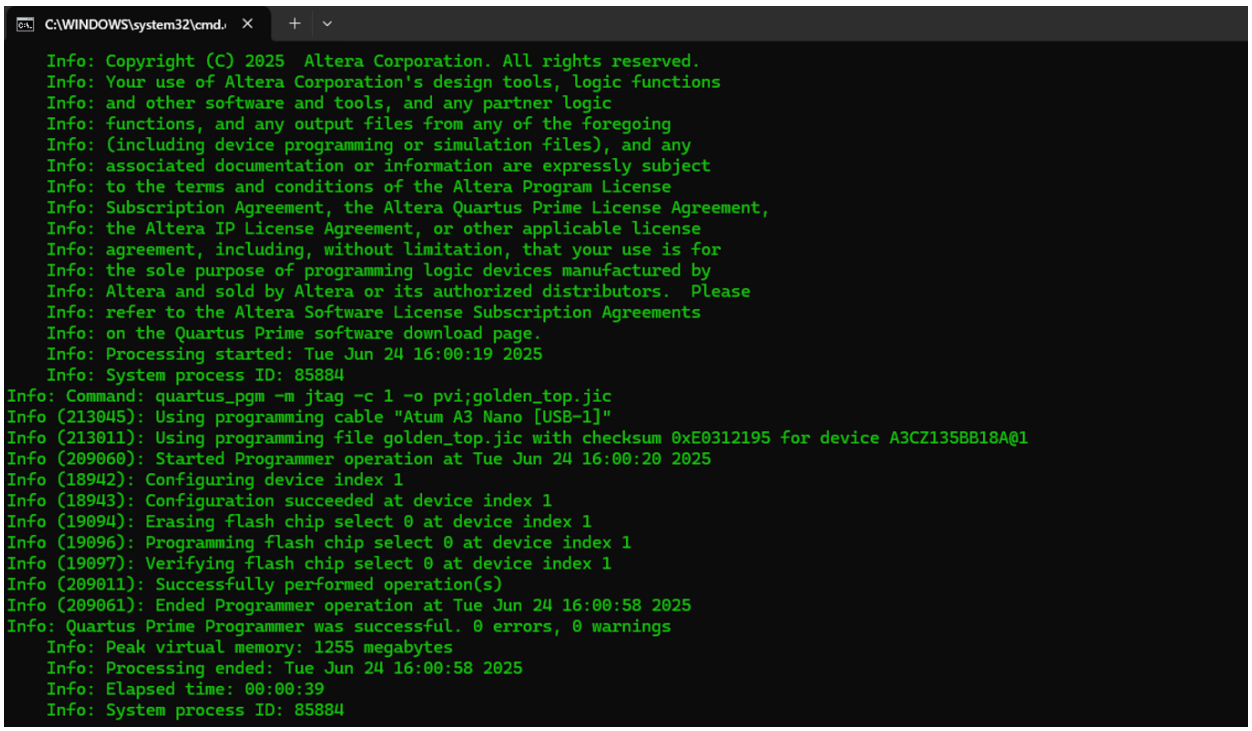

Figure 5-4 Programmed successfully

5.3 Programming Steps with Quartus GUI

This section explains how to use the Quartus Prime Programmer (GUI) to convert the jic file and program it into the QSPI Flash.

1. Open the Quartus Prime Programmer and select **File ➔ Programming File Generator…** (see Figure 5-5).

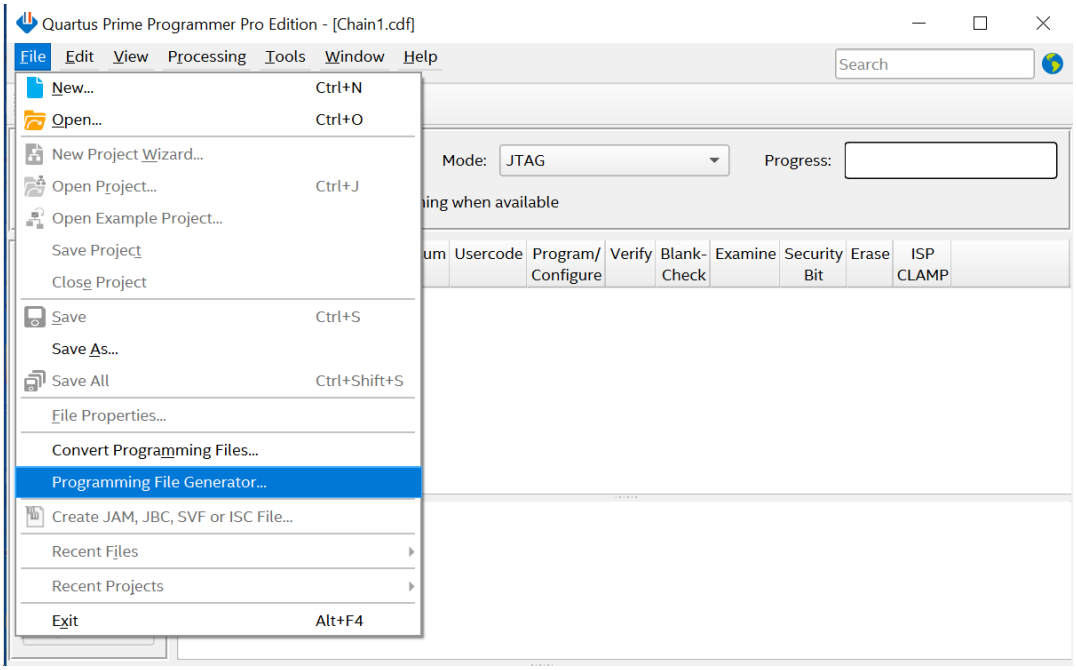

Figure 5-5 Open the Programming File Generator

When the Programming File Generator window appears (as shown in Figure 5-6):

- Set Device Family to Agilex 3.
- Set Configuration Mode to Active Serial x4.
- In the **Output Files** tab / **Output Directory**, choose the desired path to save the .jic file. Do not use the default Quartus path as it may cause errors during file generation.
- In the **Output Files** tab, check the JTAG Indirect Configuration File (.jic) option.

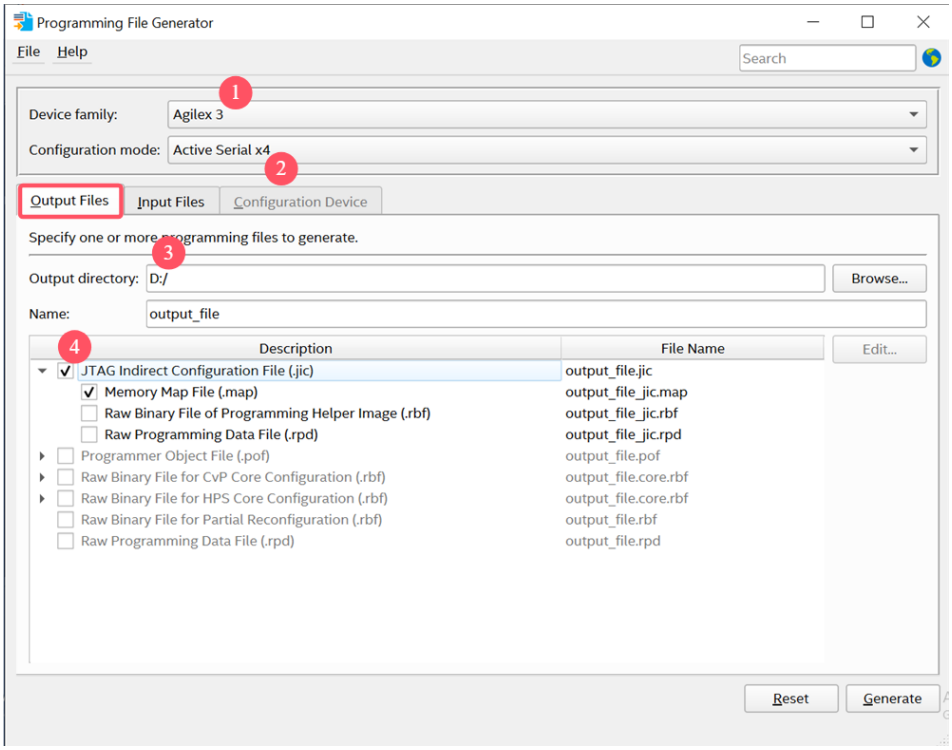

Figure 5-6 Setting output file

In the **Input Files** tab, click **Add Bitstream…** and select the .sof file you wish to convert. Once selected, it will appear in the window(see Figure 5-7).

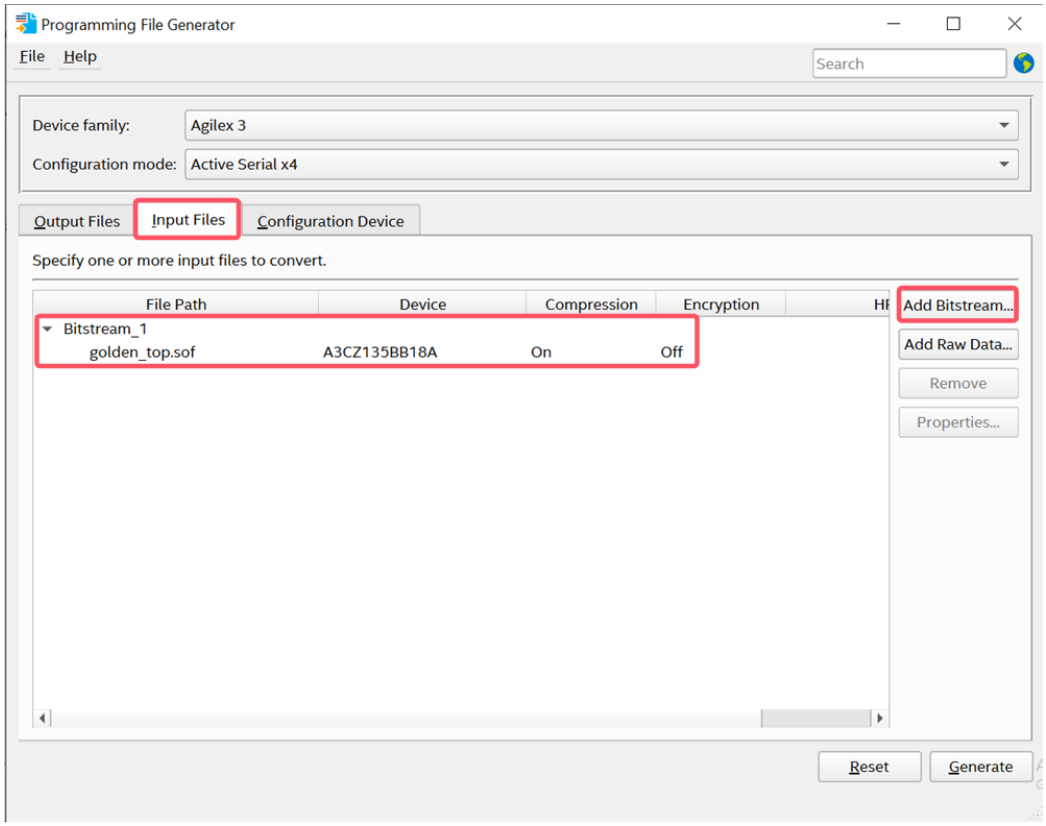

Figure 5-7 Setting input files

In the **Configuration Device** tab, click **Select…**. In the appeared **Select Devices** window, check **Agilex 3** and **A3CZ135BB18A**, then click **OK** to complete the setup(see Figure 5-8).

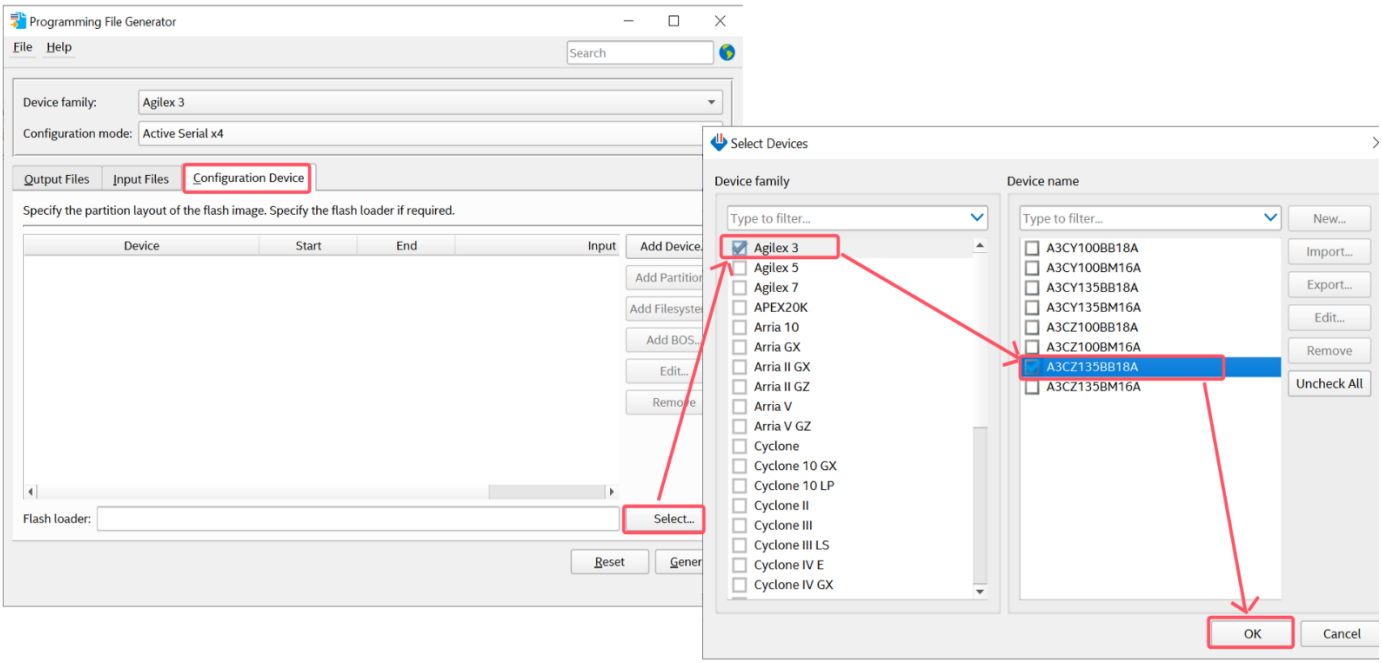

Figure 5-8 Setting Flash loader

In the **Configuration Device** tab, click **Add Device…**. In the **Configuration Device** window, select **QSPI128** and then click **OK** to finish(see Figure 5-9).

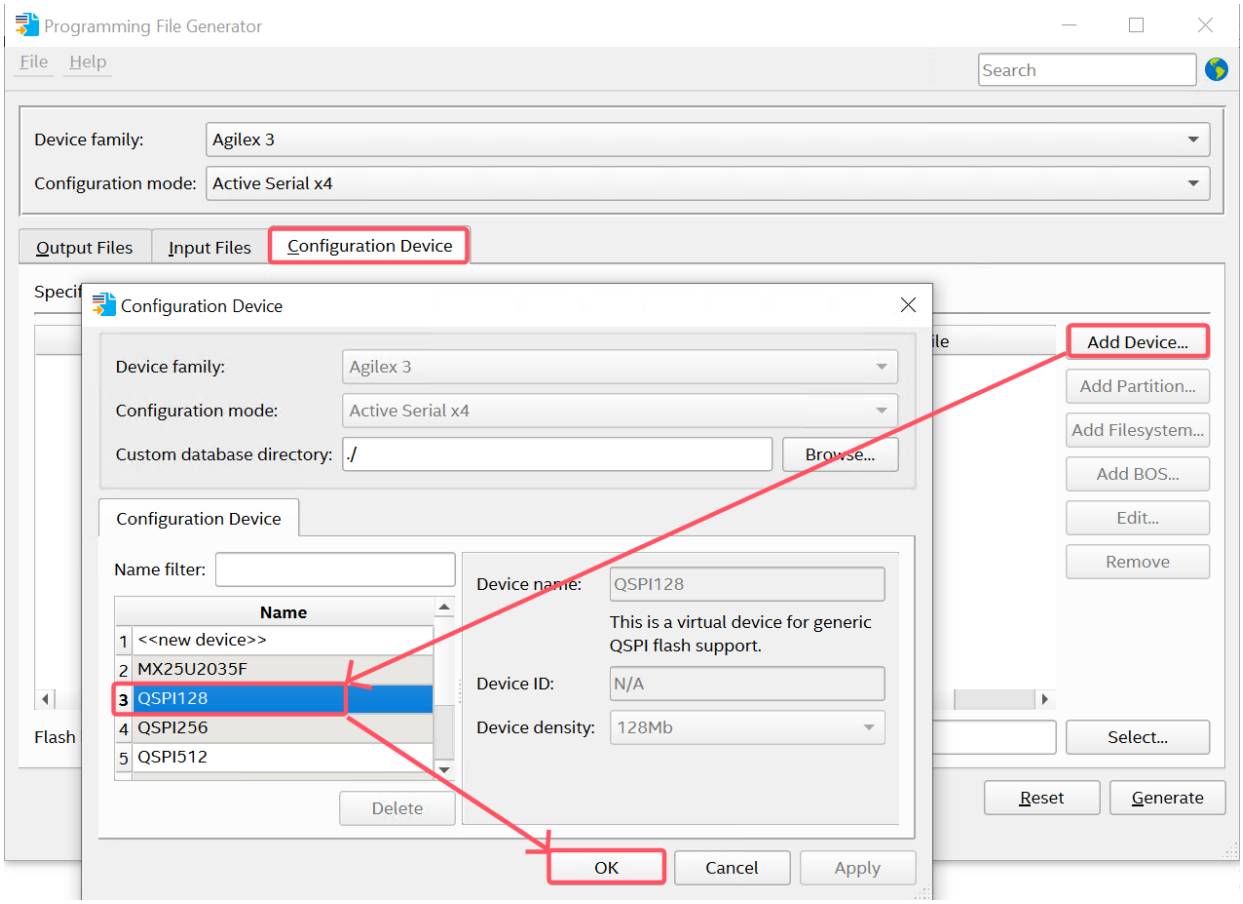

Figure 5-9 Setting Configuration Device

Next, in the **Configuration Device** tab, select the newly added QSPI128 and click **Add Partition**…. In the Add Partition window, choose the previously added .sof file under **Input File** and click **OK** to complete the setup(see Figure 5-10).

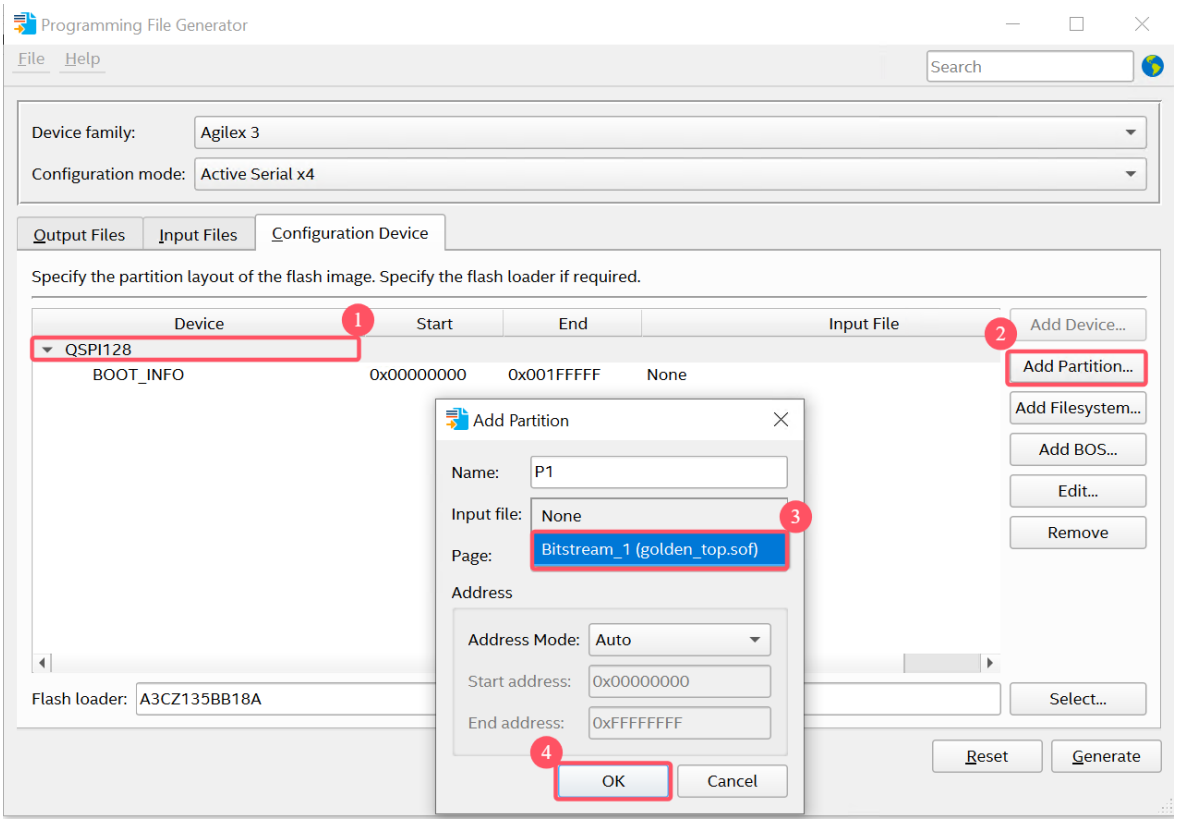

Figure 5-10 Setting Partition

Finally, click **Generate** to produce the .jic file(see Figure 5-11). When the window displays "Successfully generated output file(s)" (see Figure 5-12), it means the file conversion was successful.

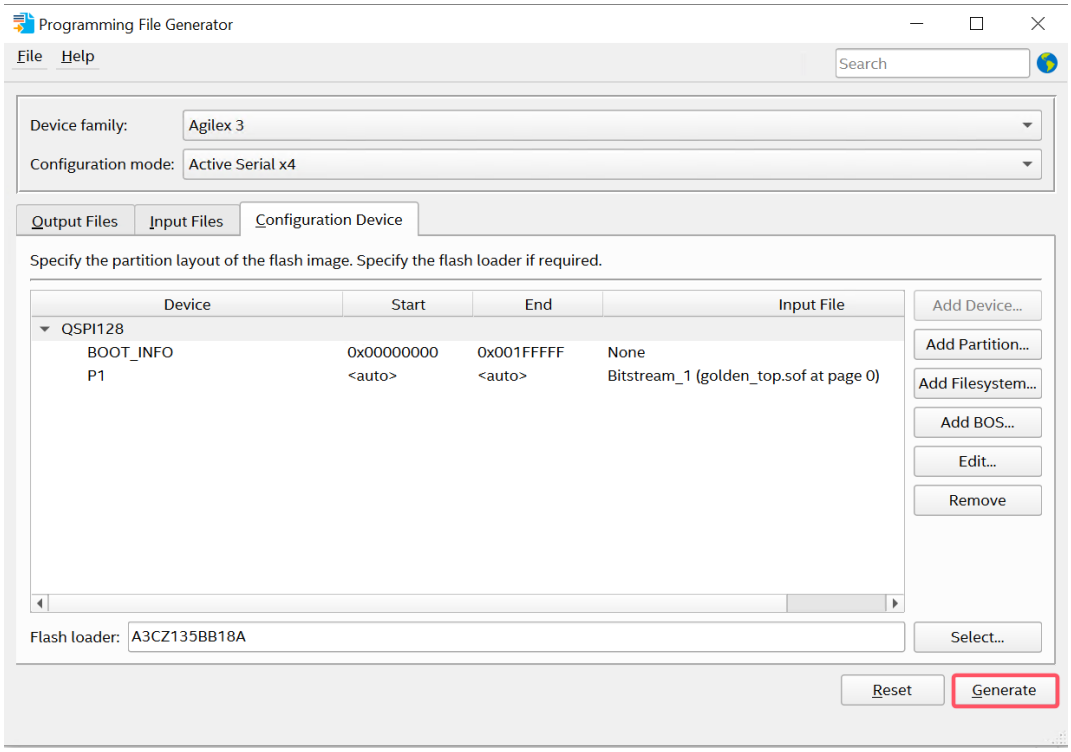

Figure 5-11 Generate .jic file

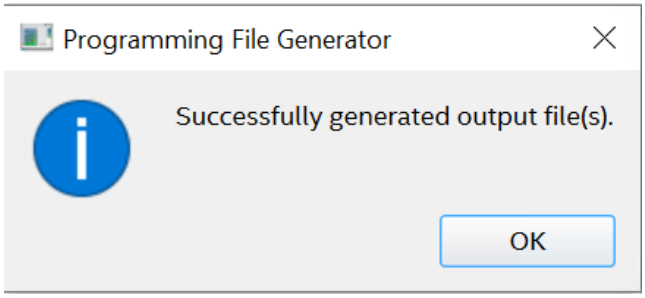

Figure 5-12 Generated file successfully

2. Verify that the MSEL Settings Switch on Atum A3 is set to AS mode (MSEL[2:1] = 2'b00) as shown in Figure 5-2.

   Ensure that the USB blaster connector (J4) of your Atum A3 is connected to your host PC with a USB cable (see Figure 5-3) and that the board's power is turned on.

3. Return to the **Quartus Prime Programmer** window. First, ensure that the USB Blaster is detected. If it is not, click **Hardware Setup** to select it (see Figure 5-13).

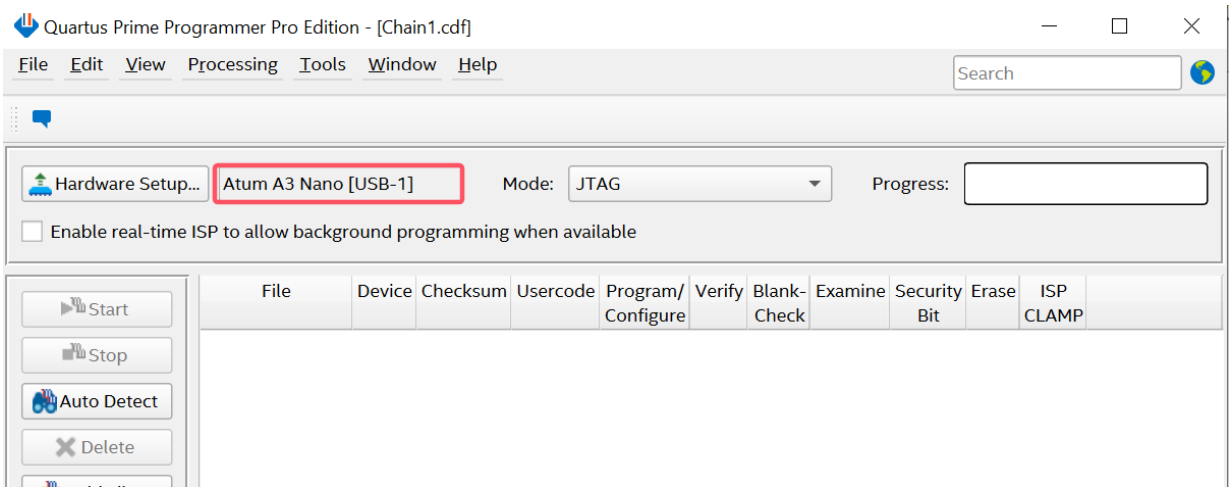

Figure 5-13 Setting USB blaster

Click **Auto Detect**. This command lists the devices attached to the board's JTAG chain(see Figure 5-14).

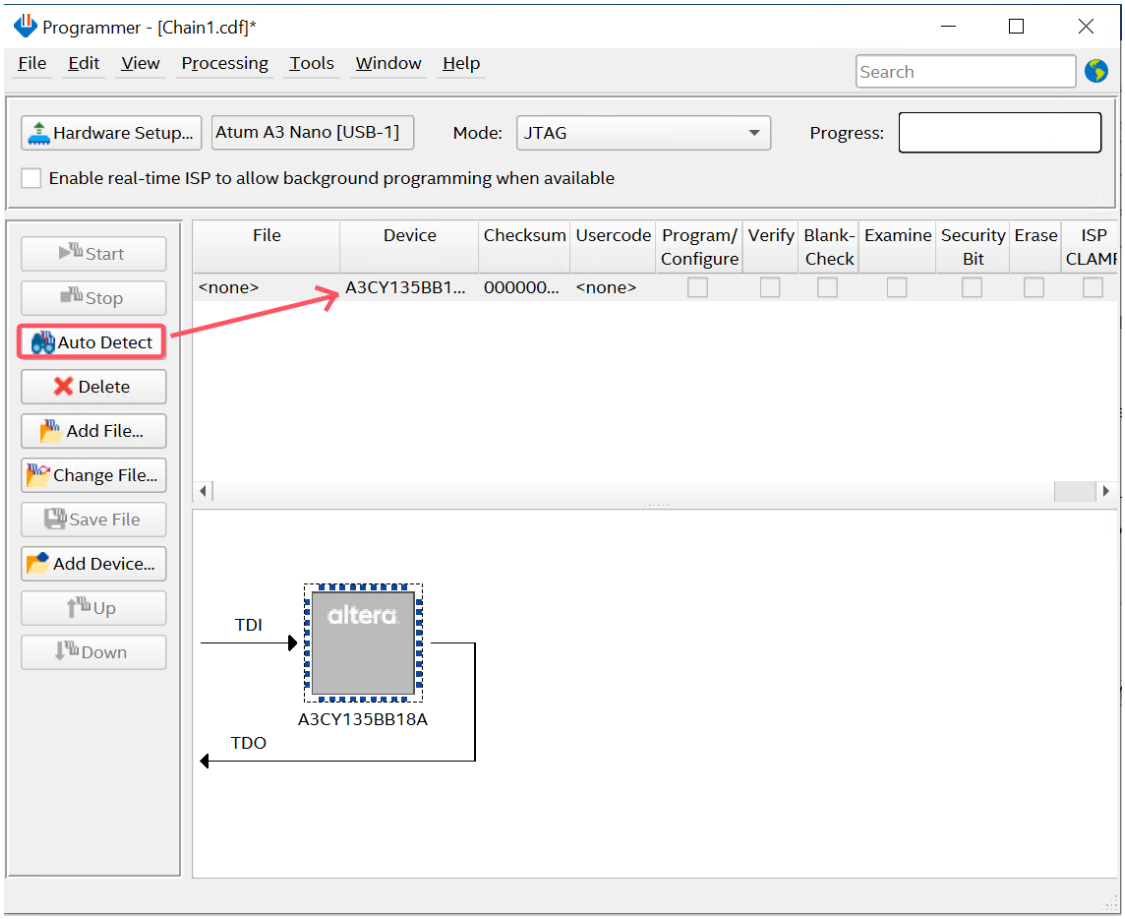

Figure 5-14 Detect FPGA device

Select the FPGA device that appears, then click Add File… to choose the .jic file to be programmed(see Figure 5-15).

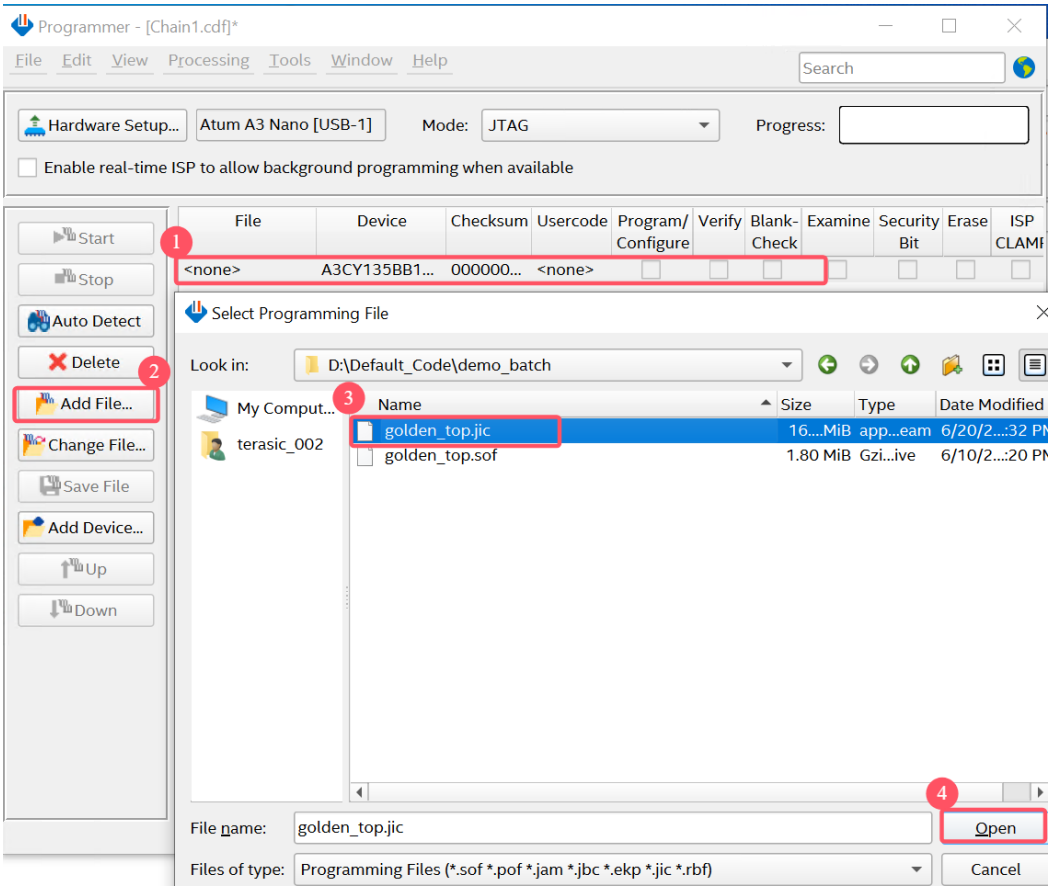

Figure 5-15 Add .jic file

Enable the **Program/Configure** option for the MT25QU128 device corresponding to the added file. Click **Start** to program the selected file to the flash(see Figure 5-16).

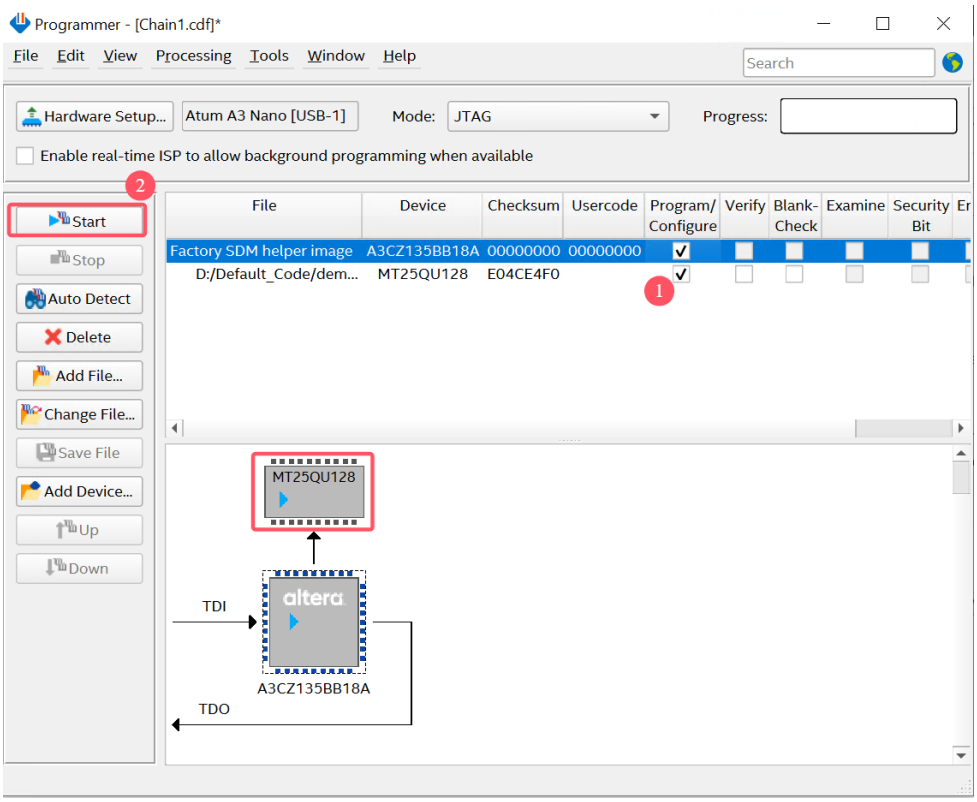

Figure 5-16 Enable programming file

Programming is complete when progress reaches 100%. Power cycle the board to verify your design(see Figure 5-17).

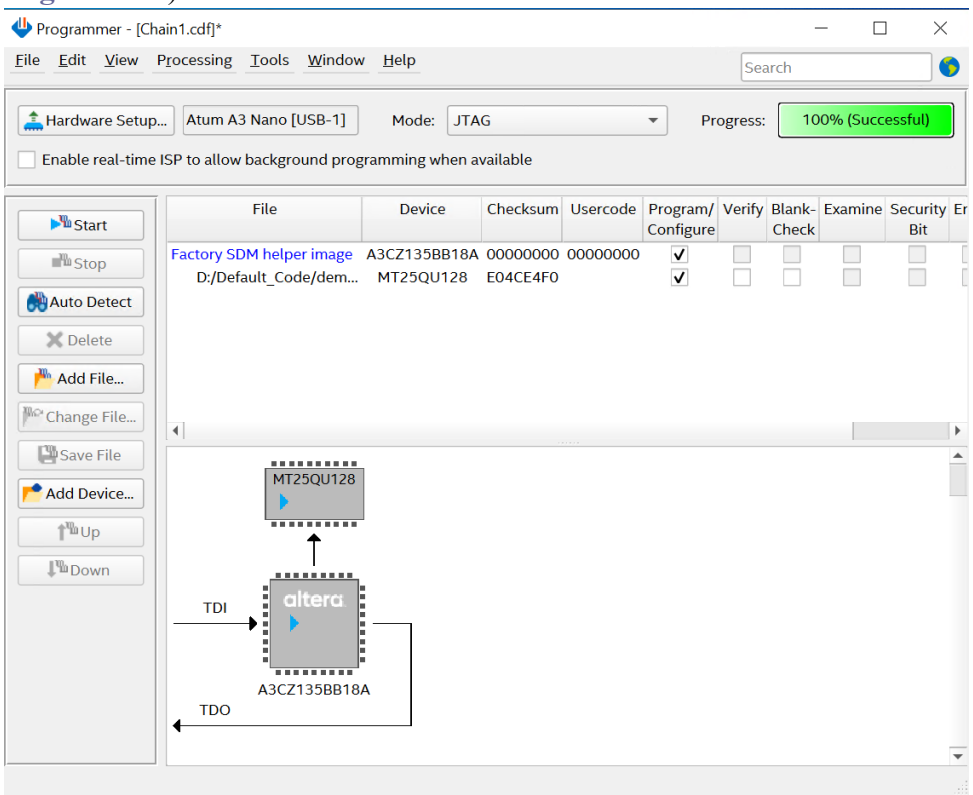

Figure 5-17 .jic file programmed successfully

Chapter 6

Appendix

6.1 Revision History

Version	*Change Log*
V1.0	Initail version
V1.1	Add section 3.12 UART
V1.2	Modify Table 3-11 Pin Assignment of TMD headers

6.2 Copyright Statement