HB0183
Handbook
CoreGPIO v3.2

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

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

1.1 Revision 6.0

Updated changes related to CoreGPIO v3.2.

1.2 Revision 5.0

Updated changes related to CoreGPIO v3.1.

1.3 Revision 4.0

Updated changes related to CoreGPIO v3.0.

1.4 Revision 3.0

Updated changes related to CoreGPIO v2.0.

1.5 Revision 2.0

Updated changes related to CoreGPIO v1.2.

1.6 Revision 1.0

Revision 1.0 was the first publication of this document. Created for CoreGPIO v1.0.

2 Introduction

Core GPIO provides an Advanced Peripheral Bus (APB) register-based interface to up to 32 general purpose inputs and 32 general purpose outputs. The input logic contains a simple three-stage synchronization circuit, and the output is also set synchronously. Each bit can be set to either fixed configuration or register-based configuration through top-level parameters, including input type, interrupt type / enable, and output enable.

2.1 Features

CoreGPIO has the following features:

- Advanced microcontroller bus architecture (AMBA) 2 APB support, forward compatibility with AMBA 3 APB
- 8-, 16-, or 32-bit APB data width
- 1 to 32 bits of I/O for all APB-width configurations
- Fixed or configurable interrupt generation:
  - Negative edge
  - Positive edge
  - Both edges
  - Level high
  - Level low
- Parameter-configurable for single-interrupt signal or up to 32-bit wide interrupt bus
- Fixed or configurable I/O type (input, output, or both)
- Configurable output enable (internal or external implementation)

2.2 Core Version

This handbook supports CoreGPIO version 3.2.

2.3 Supported Families

CoreGPIO v3.2 is a generic core and supports all the device families and utilization and performance results are shown for some of the families in Device Utilization and Performance, page 3.

2.4 Supported Interfaces

CoreGPIO is available with the APB slave interface and must be connected to an APB master interface. Microsemi recommends that you use SmartDesign in the Libero® Integrated Design Environment (IDE) or Libero System-on-Chip (SoC) Project Manager to instantiate, configure, connect and generate CoreGPIO in a processor-based system, using ARM® Cortex®-M1, Core8051s, or CoreABC.

2.5 Device Utilization and Performance

The estimated frequency reported is when all the ports of the CoreGPIO are driven.

A summary of utilization and performance data is shown in Table 1and Table 2. .

Table 1 • **CoreGPIO Utilization and Performance Data (minimum configuration)**

Family	Tiles			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total	
IGLOO/e/PLUS	32	34	66	AGL600V5	0.48	510
ProASIC3/E/L	32	33	65	A3P600	0.47	612
Fusion	32	33	65	AFS600	0.47	612.8
ProASIC^{PLUS}	32	32	64	APA150	1	450
Axcelerator	32	13	45	AX250	2	660.9
RTAX-S	32	13	45	RTAX250S	2	754.9
SmartFusion	32	33	65	A2F500M3G	0.56	612.8
SmartFusion2	32	13	46	M2S050	0.08	972.8
RTG4	32	15	47	RT4G150	0.03	737.3
PolarFire	32	13	45	MPF300T	0.01	2214.8

Note: Data in this table were achieved using typical synthesis and layout settings.

Note: Minimum configuration consists of the following parameter values: IO_NUM = 8, APB_WIDTH = 8, OE_TYPE = 0, INT_BUS = 0, FIXED_CONFIG_(0...7) = 1, IO_TYPE_(0...7) = 0, IO_INT_TYPE_(0...7) = 7

Table 2 • **CoreGPIO Utilization and Performance Data (maximum configuration)**

Family	Tiles			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total	
IGLOO/e/PLUS	512	1364	1876	AGL600V5	13.57	98.6
ProASIC3/E/L	512	1355	1867	A3P600	13.51	115.5
Fusion	512	1355	1867	AFS600	13.51	115.5
ProASIC^{PLUS}	512	1695	2207	APA150	36	108.1
Axcelerator	768	923	1691	AX250	41	188.2
RTAX-S	768	923	1691	RTAX250S	41	217.1
SmartFusion	768	1412	2180	A2F500M3G	18.92	115
SmartFusion2	548	919	1467	M2S050	2.6	218.8
RTG4	548	921	1469	RT4G150	0.97	185.8
PolarFire	512	987	1499	MPF300T	0.50	326.3

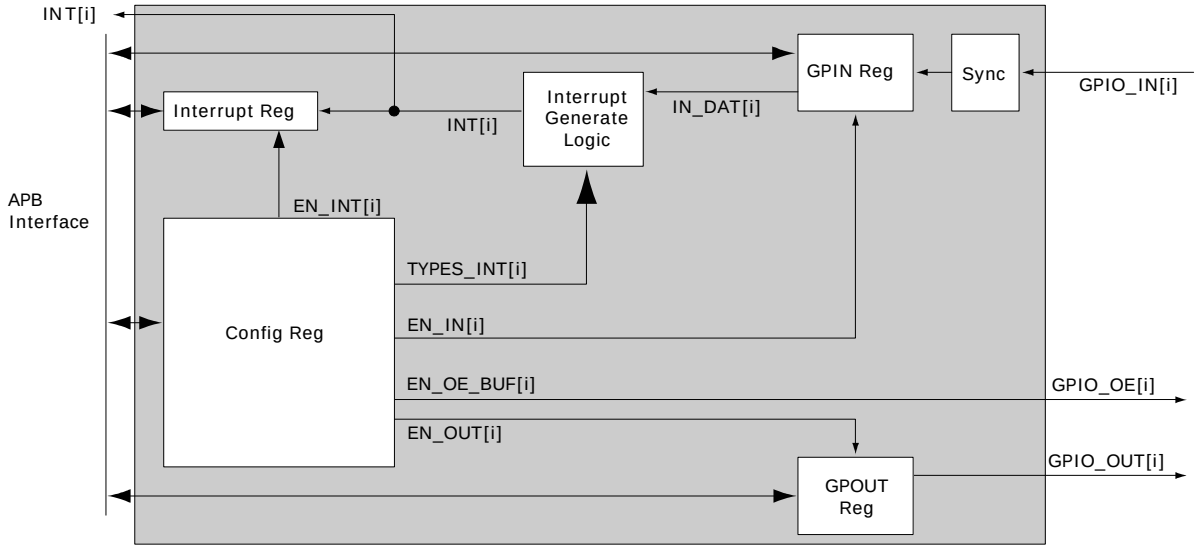
Note: Data in this table were achieved using typical synthesis and layout settings.

Note: Maximum configuration consists of the following parameter values: IO_NUM = 32, APB_WIDTH = 8, OE_TYPE = 1, INT_BUS = 1, FIXED_CONFIG (0...31) = 0.

2.6 Functional Block Diagram

Figure 1 illustrates a single-bit block diagram (this is replicated up to 32 times, depending on the number of I/Os).

Figure 1 • **Single I/O Bit Block Diagram for CoreGPIO**



3 Interface

3.1 Generics

Table 3 gives descriptions for the CoreGPIO generics.

Table 3 • **CoreGPIO Generics**

Parameter	Values	Default Value	Description
APB_WIDTH	8, 16, 32	32	APB data width.
IO_NUM	1–32	32	Number of GPIOs.
OE_TYPE	0 or 1	0	If 0, output buffering is implemented outside CoreGPIO. The user is responsible for instantiating tristate buffers outside of the core. If 1, output buffering (if enabled) is implemented inside the core. When GPIO_OE[i] is 0, GPIO_OUT is high impedance (Z).
FIXED_CONFIG_x	0 or 1	0	If 0, configuration for bit x (0-31) is set via APB-accessible register CONFIG_x (see the "Register Map" section). If 1, configuration for bit x (0-31) is set via "IO_INT_TYPE_x" (described below) and "IO_TYPE_x".
IO_INT_TYPE_x	0-5	0	Interrupt types selected according to the following scheme: 0 – Level High 1 – Level Low 2 – Edge Positive 3 – Edge Negative 4 – Edge Both 7 – Disabled Note that selecting one type will synthesize out logic for other types. For example, Level High will remove and/or gates for edge detect. **Note:** Interrupts are triggered through input depending upon the parameter IO_INT_TYPE.
IO_TYPE_x	0-2	0	If 0, bit x is of type input only. Output logic will be synthesized out. If 1, bit x is of type output only. Input logic will be synthesized out. If 2, bit x is of type input and output (both).
IO_VAL_x	0 or 1	0	Sets the output at reset for GPIO bit x.
INT_BUS	0 or 1	0	If 0, the INT_OR output is fixed at 0 (unused). If 1, the INT_OR output is set if any of the INT signals are set (OR operation).

3.2 Ports

Table 4 outlines the top-level signals for CoreGPIO.

Table 4 • **CoreGPIO Ports**

Name	Type	Description
APB Bus Signals		
PCLK	Input	APB System Clock – Reference clock for all internal logic
PRESETN	Input	APB active low asynchronous reset
PWDATA [APB_WIDTH-1:0]	Input	APB write data
PRDATA [APB_WIDTH-1:0]	Output	APB read data
PADDR[7:0]	Input	APB address bus
PENABLE	Input	APB strobe – Indicates the second cycle of an APB transfer
PSEL	Input	APB slave select
PWRITE	Input	APB write/read select signal
PREADY	Output	APB 3 ready signal for future APB 3 compliance; tied internally High
PSLVERR	Output	APB 3 transfer error signal for future APB 3 compliance; tied internally Low
GPIO Signals		
GPIO_IN [IO_NUM-1:0]	Input	GPIO input
GPIO_OUT[IO_NUM-1:0]	Output	GPIO output
GPIO_OE[IO_NUM-1:0]	Output	GPIO output enable
INT[IO_NUM-1:0]	Output	Interrupt mask; can be connected directly to processor (for example, Cortex-M1)
INT_OR	Output	Provides an OR'ed version (single wire) of the interrupt mask provided on INT[IO_NUM-1:0]
Note: Unless otherwise noted, all of the signals above are active High.		

4 Register Map

4.1 Overview

Table 5 through Table 7 describe the CoreGPIO Register map

Table 5 • **CoreGPIO Register Address Map (APB_WIDTH = 8)**

PADDR[7:0]	Type	Reset Value (Hex)	Brief Description
0x00-0x7C (0x00, 0x04, 0x08, …, 0x7C)	R/W	0x00	8-bit configuration registers for all 32 bits; 1 register per bit.
0x80	R/W	0x00	Interrupt clear register 1 (bits 7:0)
0x84	R/W	0x00	Interrupt clear register 2 (bits 15:8)
0x88	R/W	0x00	Interrupt clear register 3 (bits 23:16)
0x8C	R/W	0x00	Interrupt clear register 4 (bits 31:24)
0x90	R	0x00	Input register 1 (bits 7:0)
0x94	R	0x00	Input register 2 (bits 15:8)
0x98	R	0x00	Input register 3 (bits 23:16)
0x9C	R	0x00	Input register 4 (bits 31:24)
0xA0	R/W	0x00	Output register 1 (bits 7:0)
0xA4	R/W	0x00	Output register 2 (bits 15:8)
0xA8	R/W	0x00	Output register 3 (bits 23:16)
0xAC	R/W	0x00	Output register 4 (bits 31:24)
Note: Values shown in hexadecimal format; type designations: R = read only; R/W = read/write. **Note:** Lower 2 bits of PADDR are unconnected inside CoreGPIO.			

Table 6 • **CoreGPIO Register Address Map (APB_WIDTH = 16)**

PADDR[7:0]	Type	Reset Value (Hex)	Brief Description
0x00-0x7C (0x00, 0x04, 0x08, …, 0x7C)	R/W	0x00	8-bit configuration registers for all 32 bits; 1 register per bit.
0x80	R/W	0x00	Interrupt clear register 1 (bits 15:0)
0x84	R/W	0x00	Interrupt clear register 2 (1bits 31:16)
0x90	R	0x00	Input register 1 (bits 15:0)
0x94	R	0x00	Input register 2 (bits 31:16)
0xA0	R/W	0x00	Output register 1 (bits 15:0)
0xA4	R/W	0x00	Output register 2 (bits 31:16)
Note: Values shown in hexadecimal format; type designations: R = read only; R/W = read/write. **Note:** Lower 2 bits of PADDR are unconnected inside CoreGPIO.			

Table 7 • **CoreGPIO Register Address Map (APB_WIDTH = 32)**

PADDR[7:0]	Type	Reset Value (Hex)	Brief Description
0x00-0x7C			
(0x00, 0x04, 0x08, …, 0x7C)	R/W	0x00	8-bit configuration registers for all 32 bits; 1 register per bit.
0x80	R/W	0x00	Interrupt clear register 1 (bits 31:0)
0x90	R	0x00	Input register 1 (bits 31:0)
0xA0	R/W	0x00	Output register 1 (bits 31:0)

Note: Values shown in hexadecimal format; type designations: R = read only; R/W = read/write.

Note: Lower 2 bits of PADDR are unconnected inside CoreGPIO.

4.2 Configuration Registers

There are up to 32 8-bit configuration registers (depending on the IO_NUM parameter). Table 8 describes the CoreGPIO configuration register operation.

Table 8 • **Per-bit Configuration Register**

Bits	Name	Function
7:5	INTTYPE	Sets the interrupt type for this particular bit: 000 – Level High 001 – Level Low 010 – Edge Positive 011 – Edge Negative 100 – Edge Both 101 to 111 – Invalid
4	Reserved	Unused
3	INTENABLE	Interrupt enable for this particular bit 1 – Enable interrupt generation 0 – Disable interrupt generation
2	OUTBUFF	Sets the output enable for this particular bit, whether via the GPIO_OE signal or implemented internally (see parameter "OE_TYPE"). 1 – Enables output 0 – Disables output
1	INREG	Input register enable 1 – Enables input register for this particular bit 0 – Disables input register for this particular bit
0	OUTREG	Output register enable 1 – Enables output functionality for this particular bit 0 – Disables output functionality for this particular bit

4.3 Interrupt Registers

These are per-bit interrupt clear registers. Writing a 1 to any bit clears the interrupt bit register of the corresponding GPIO bit.

In 32-bit mode, all 32 interrupt bits are in a single 32-bit register located at address 0x80.

In 16-bit mode, 32 interrupt bits are split into two 16-bit registers located at addresses 0x80 and 0x84.

In 8-bit mode, 32 interrupt bits are split into four 8-bit registers located at addresses 0x80, 0x84, 0x88, and 0x8C.

4.4 Input Registers

Read-only for input configured ports. Disabling a bit in this register with the CONFIG_X[1] (INREG) bit will force the bit to 0 via a MUX, while keeping the incoming current value in the register.

In 32-bit mode, all 32 input bits are in a single 32-bit register located at address 0x90.

In 16-bit mode, 32 input bits are split into two 16-bit registers located at addresses 0x90 and 0x94.

In 8-bit mode, 32 input bits are split into four 8-bit registers located at addresses 0x90, 0x94, 0x98, and 0x9C.

4.5 Output Registers

The output registers are writeable/readable for output configured ports, and are logical "don't cares" for input configured ports. Disabling a bit in this register with the CONFIG_X[0] (OUTREG) bit will force the bit to 0 via a MUX, while keeping the previously written value in the output register.

In 32-bit mode, all 32 output bits are in a single 32-bit register located at address 0xA0.

In 16-bit mode, 32 output bits are split into two 16-bit registers located at addresses 0xA0 and 0xA4.

In 8-bit mode, 32 output bits are split into four 8-bit registers located at addresses 0xA0, 0xA4, 0xA8, and 0xAC.

5 Tool Flow

5.1 License

No license is required for this core.

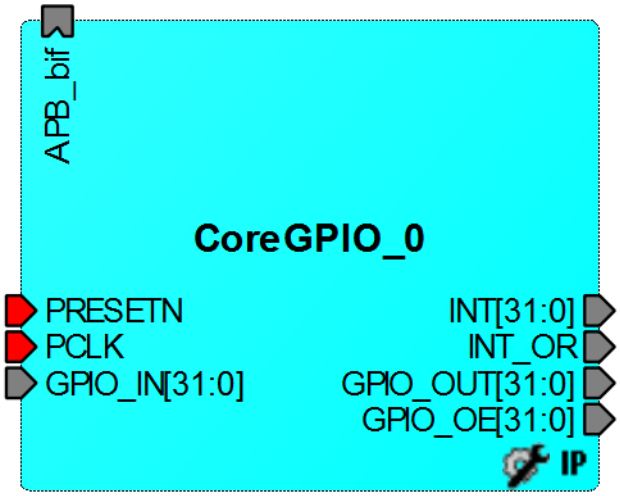
5.1.1 RTL

Complete RTL source code is provided for the core and testbench.

5.2 SmartDesign

CoreGPIO is available through the Libero SoC IP Catalog. Download it from a remote web-based repository and install into your local vault to make it ready to use. Once installed in the Libero software, you can instantiate, configure, connect, and generate the core using the SmartDesign tool.

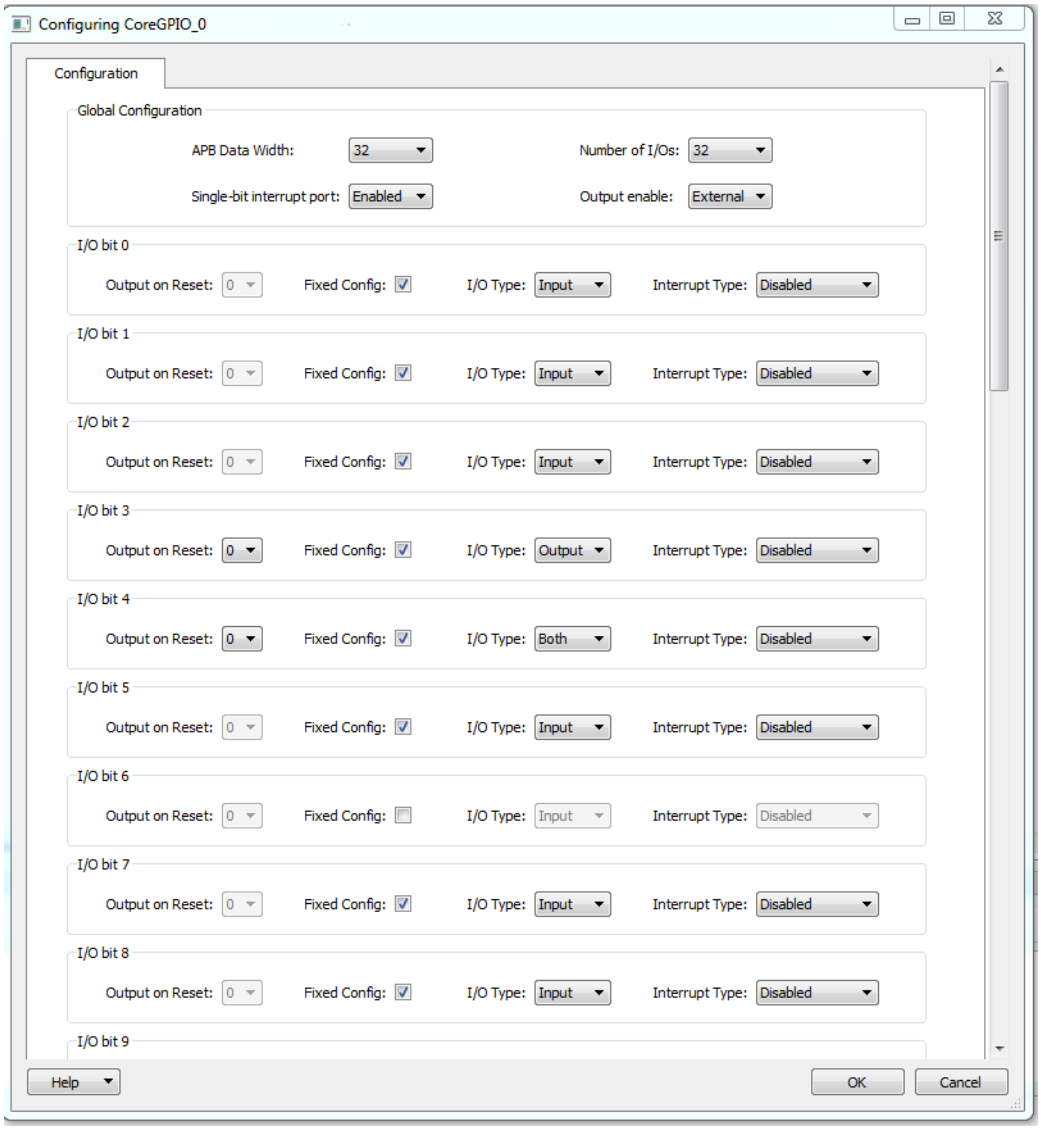
Figure 2 • **SmartDesign CoreGPIO Instance View**



5.2.1 Configuring CoreGPIO in SmartDesign

Figure 3 shows the CoreGPIO configuration window, as well as cross-references to the corresponding top-level parameters.

Figure 3 • **CoreGPIO Configuration Window**



5.3 Simulation Flows

To run simulations, select the user testbench in SmartDesign through the CoreGPIO IP configuration GUI. Generate the design in SmartDesign. The appropriate testbench files are now installed.

To run the testbenches, set the design root to the CoreGPIO instance in the Libero IDE or SoC Design Explorer and click the Simulation icon in the Project Flow tab. This invokes ModelSim® and automatically runs the simulation.

5.4 Synthesis in Libero

To run synthesis on the CoreGPIO, set the design root to the IP component instance and click on Synthesize in Libero Design Flow pane. This will invoke Synplify Pro and automatically runs the synthesis.

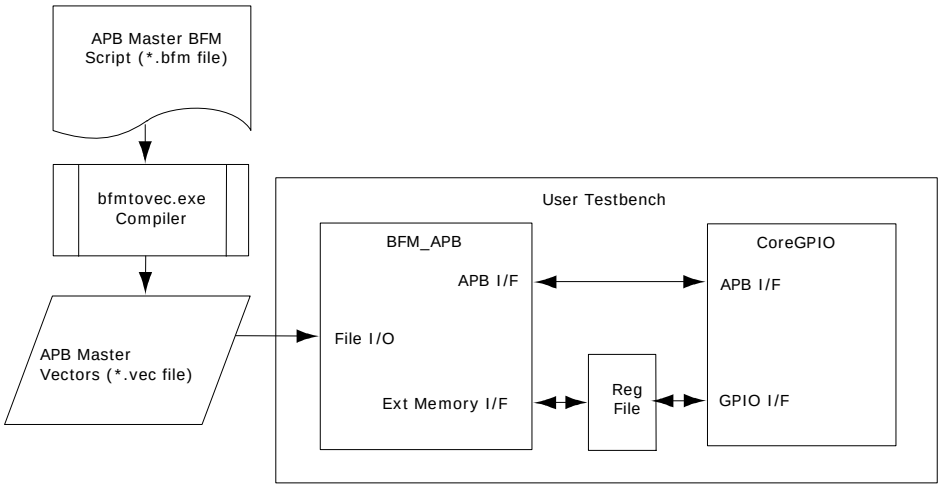
5.5 Place-and-Route in Libero

After setting the design root appropriately and running Synthesis, click the **Layout** icon in the Project Manager to invoke Designer. CoreGPIO requires no special place-and-route settings.

6 Testbench

An example user testbench is included with CoreGPIO for both VHDL and Verilog. The testbench is provided as an obfuscated bus functional model (BFM), connected as shown in Figure 3 to a CoreGPIO block. You can examine and change the testbench by modifying the *.bfm file and generating a *.vec APB master vector file, as shown in Figure 4.

Figure 4 • **CoreGPIO User Testbench Block Diagram**



The user testbench instantiates a Microsemi DirectCore AMBA BFM module to emulate an APB master that controls the operation of CoreGPIO via reads and writes to access internal registers. A BFM ASCII script source file with comments is included in the directory <proj>/simulation, where <proj> represents the path to your Libero IDE or SoC project.

The BFM source file, coregpio_usertb_apb_master.bfm, controls the APB master processor. This BFM source file is automatically recompiled each time the simulation is invoked from Libero IDE or SoC by the bfmtovec.exe executable, if running on a Windows® platform, or by the bfmtovec.lin executable, if running on a Linux platform. The coregpio_usertb_apb_master.vec vector file, created by the bfmtovec executable, is read in by the BFM module for simulation in ModelSim.

You can alter the BFM script, if desired. Refer to the Microsemi DirectCore AMBA BFM User Guide for more information.

7 System Integration

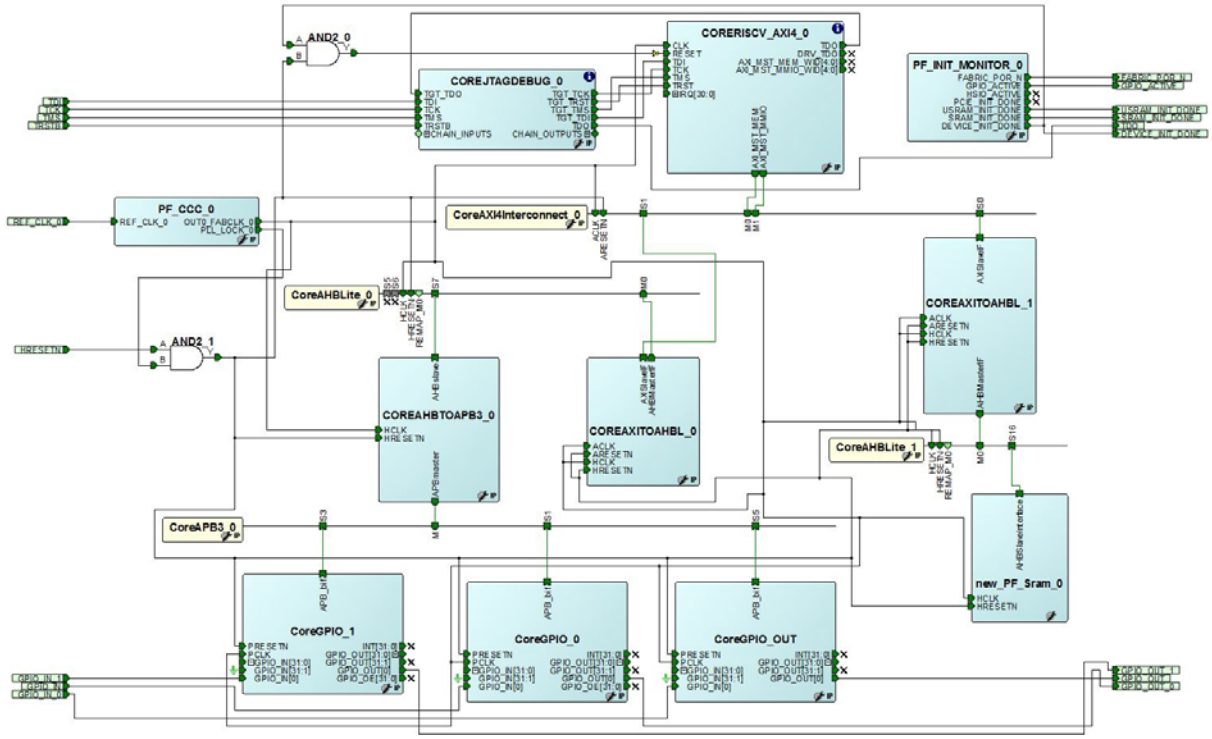
This section provides hints to ease the integration of CoreGPIO.

The example design tests a simple I/O application for CoreGPIO on PolarFire EVAL KIT.

In the design,

- CoreGPIO_0, CoreGPIO_1, CoreGPIO_OUT are connected to CoreRISCV_AXI4_0 through CoreAXI4Interconnect_0, CoreAXITOAHBL_0, CoreAHBLite_0, COREAHBTOAPB3_0 and COREAPB3_0
- CoreGPIO_0, CoreGPIO_1, CoreGPIO_OUT are configured by CoreRISCV_AXI4_0 firmware.

Figure 5 • **CoreGPIO Example Design**



- HRESETN which is and with PF_CCC_0/PLL_LOCK_0 is used for all the resets.
- CoreGPIO_0, CoreGPIO_1 & CoreGPIO_OUT have PCLK driven from PF_CCC_0/OUT0_FABCLK_0.
- CoreGPIO_0 is configured with 8bit of APB_WIDTH, CoreGPIO_1 with 16bit APB_WIDTH and CoreGPIO_OUT with 32 bit APBWIDTH.
- GPIO_OUT, GPIO_OUT_0, GPIO_OUT_1 output ports are connected to LEDs.
- GPIO_IN, GPIO_IN_0, GPIO_IN_1 input ports are connected to switches on the board.
- CoreRISCV_AXI4 firmware configures the instances of CoreGPIO and checks for CoreGPIOs' input status and accordingly writes onto CoreGPIOs' output ports.

7.1 Configuring CoreGPIO

This section provides information about CoreGPIO configuration.

7.1.1 Example 1:

The CoreGPIO is configured as mentioned below.

```
IO_NUM = 10
For I/O bit 0, IO_TYPE = INPUT
For I/O bit 1, IO_TYPE = INPUT
For I/O bit 2, IO_TYPE = INPUT
For I/O bit 3, IO_TYPE = OUTPUT
For I/O bit 4, IO_TYPE = OUTPUT
For I/O bit 5, IO_TYPE = OUTPUT
For I/O bit 6, IO_TYPE = INPUT
For I/O bit 7, IO_TYPE = INPUT
For I/O bit 8, IO_TYPE = BOTH
For I/O bit 9, IO_TYPE = BOTH
```

The width of top level ports GPIO_IN, GPIO_OUT, GPIO_INT, and GPIO_OE depends on the IO_NUM parameter as mentioned in Port description table. As the IO_NUM is 10, the ports will get generated as GPIO_OUT[9:0], GPIO_IN[9:0], GPIO_INT[9:0] and GPIO_OE [9:0] as shown in Figure 7.

Here GPIO_IN[0], GPIO_IN[1], GPIO_IN[2], GPIO_IN[6], GPIO_IN[7], GPIO_IN[8], GPIO_IN[9] are relevant inputs and GPIO_IN[3], GPIO_IN[4], GPIO_IN[5] need to tie low.

Similarly, GPIO_OUT[3], GPIO_OUT[4], GPIO_OUT[8], GPIO_OUT[9] are relevant outputs and GPIO_OUT[0], GPIO_OUT[1], GPIO_OUT[6], GPIO_OUT[7] can be left unused.

Hence, when using CoreGPIO, user should take care of which IO is configured as input, output, and inout and connect accordingly.

Figure 6 • **Configuring CoreGPIO Example 1**

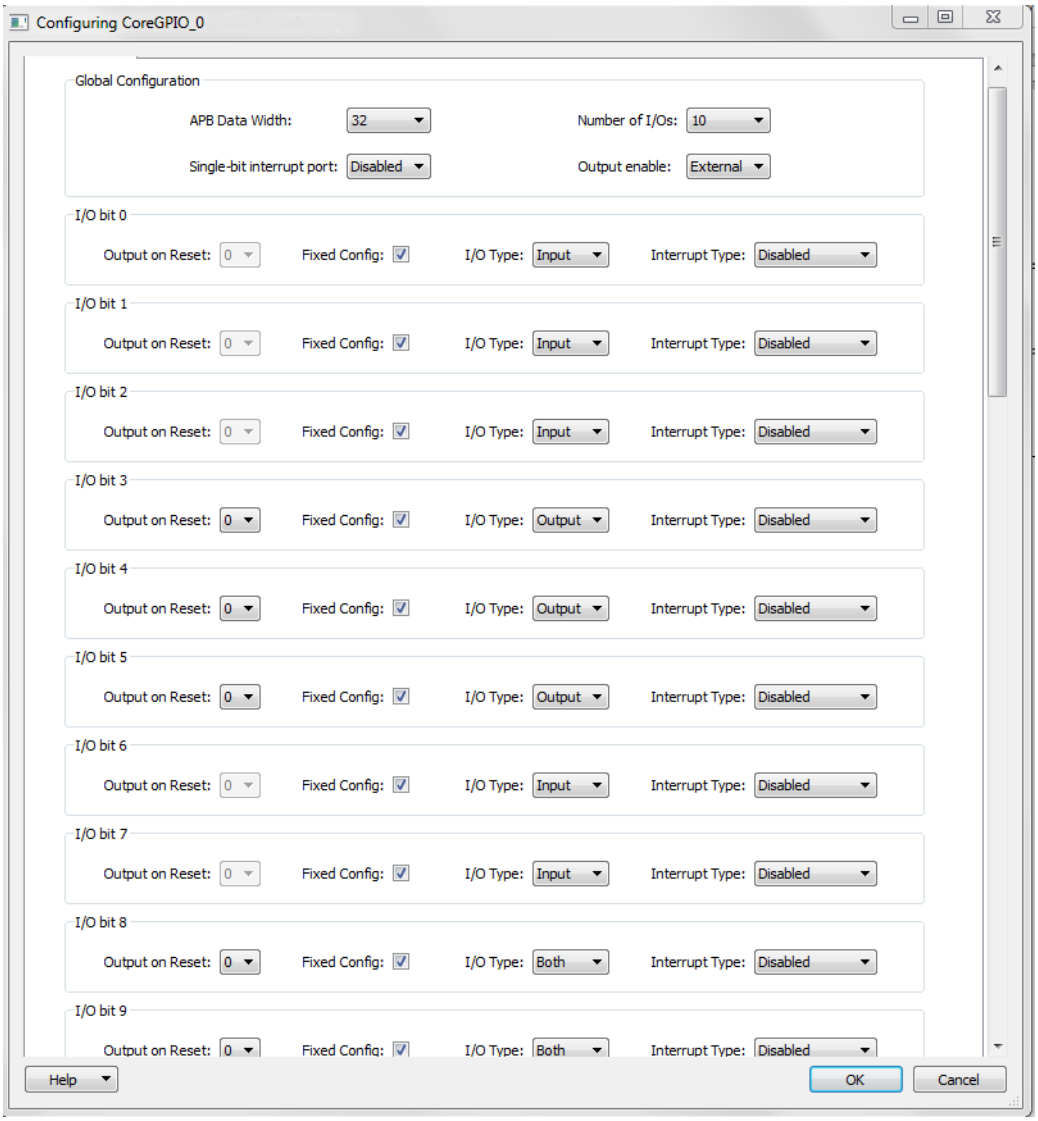
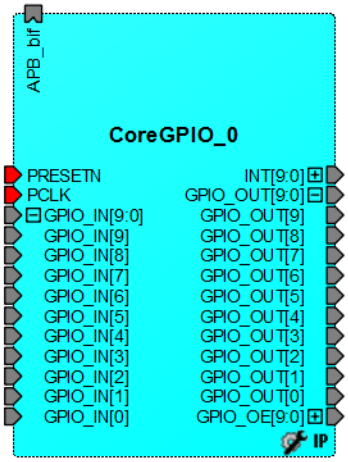


Figure 7 • **CoreGPIO SmartDesign Example 1**



7.1.2 Example 2:

The CoreGPIO is configured as mentioned below.

```
IO_NUM = 4,
For I/O bit 0, IO_TYPE = INPUT
For I/O bit 1, IO_TYPE = INPUT
For I/O bit 2, IO_TYPE = INPUT
For I/O bit 3, IO_TYPE = INPUT
```

The width of top level ports GPIO_IN, GPIO_OUT, GPIO_INT and GPIO_OE depends on the IO_NUM parameter as mentioned in Port description table. As the IO_NUM is 4, the ports will get generated as GPIO_OUT[3:0], GPIO_IN[3:0], GPIO_INT[3:0], and GPIO_OE [3:0] as shown in Figure 9.

In this case all IOs are configured as Input. So GPIO_IN [3:0] is relevant and GPIO_OUT[3:0] can be left unused.

Figure 8 • **Configuring CoreGPIO Example 2**

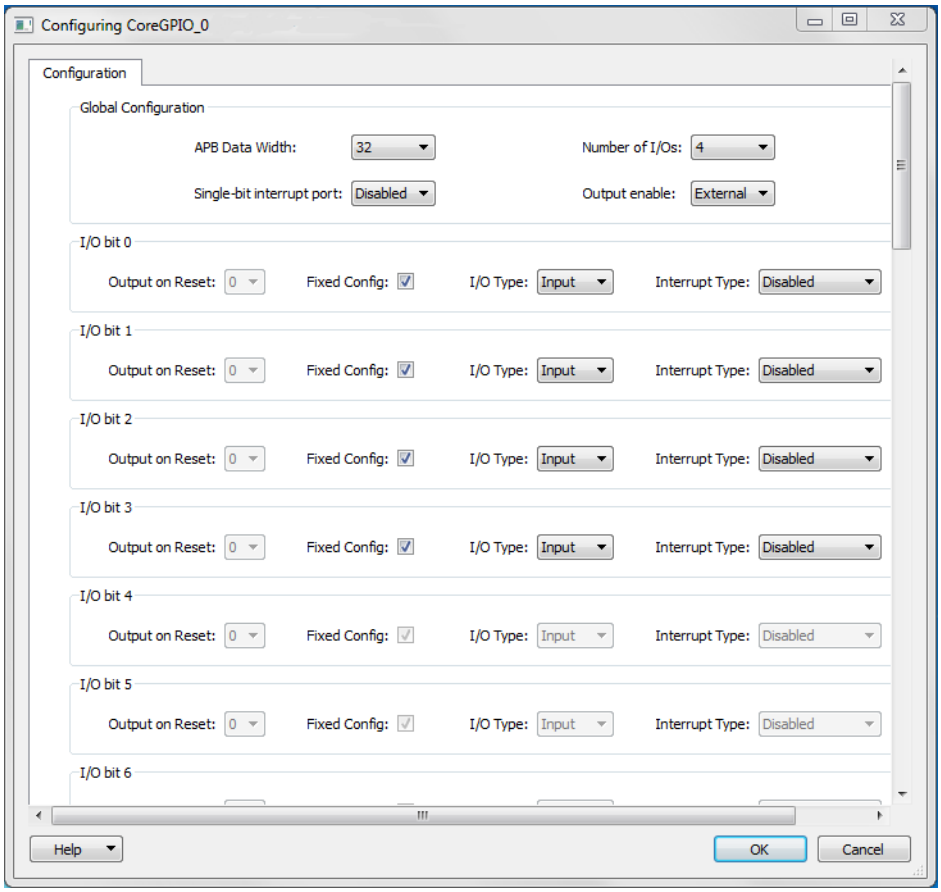


Figure 9 • **CoreGPIO SmartDesign Example 2**