




# Gowin Primitives

## User Guide

SUG283-2.7E, 10/28/2021

**Copyright © 2021 Guangdong Gowin Semiconductor Corporation. All Rights Reserved.**

**GOWIN**, , Gowin, LittleBee, GowinSynthesis, and GOWINSEMI are trademarks of Guangdong Gowin Semiconductor Corporation and are registered in China, the U.S. Patent and Trademark Office, and other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders. No part of this document may be reproduced or transmitted in any form or by any denotes, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of GOWINSEMI.

#### **Disclaimer**

GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. All information in this document should be treated as preliminary. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.

## Revision History

| Date       | Version | Description   |
|------------|---------|---|
| 04/20/2017 | 1.0E    | Initial version published.  |
| 09/19/2017 | 1.1E    | <ul style="list-style-type: none"> <li>● GW1NR-4, GW1N-6, GW1N-9, GW1NR-9 devices added.</li> <li>● ELVDS_IOBUF, TLVDS_IOBUF, BUFG, BUFS, OSC, IEM added.</li> <li>● DSP primitive updated.</li> <li>● Some ports of ODDR/ODDRC, IDDR_MEM, IDES4_MEM, IDES8_MEM, RAM16S1, RAM16S2, RAM16S4, RAM16SDP1, RAM16SDP2, RAM16SDP4, ROM16 updated.</li> <li>● Some Attribute of OSC, PLL and DLLDLY updated;</li> <li>● Some primitive instantiation updated.</li> <li>● MIPI_IBUF_HS, MIPI_IBUF_LP, MIPI_OBUF, IDES16 and OSER16 updated.</li> <li>● Some Attribute of CLKDIV updated.</li> </ul> |
| 04/12/2018 | 1.2E    | Vhdl primitives instantiation added.  |
| 08/08/2018 | 1.3E    | <ul style="list-style-type: none"> <li>● GW1N-2B, GW1N-4B, GW1NR-4B, GW1N-6ES, GW1N-9ES, GW1NR-9ES, GW1NS-2, GW1NS-2C devices added.</li> <li>● I3C_IOBUF, DHCEN added.</li> <li>● User Flash added.</li> <li>● EMPU added.</li> <li>● Primitive name updated.</li> </ul>   |
| 10/26/2018 | 1.4E    | <ul style="list-style-type: none"> <li>● GW1NZ-1, GW1NSR-2C devices added.</li> <li>● OSCZ, FLASH96KZ added.</li> </ul>   |
| 11/15/2018 | 1.5E    | <ul style="list-style-type: none"> <li>● GW1NSR-2 device added.</li> <li>● GW1N-6ES, GW1N-9ES, GW1NR-9ES devices removed.</li> </ul>  |
| 01/26/2019 | 1.6E    | <ul style="list-style-type: none"> <li>● GW1NS-2 supported by 8 frequency division of CLKDIV added.</li> <li>● Removed GW1N-1 from the devices supported by TLVDS_TBUF/OBUF.</li> </ul>   |
| 02/25/2019 | 1.7E    | Removed GW1N-1 from the devices supported by TLVDS_IOBUF.   |
| 05/20/2019 | 1.8E    | <ul style="list-style-type: none"> <li>● GW1N-1S device added.</li> <li>● MIPI_IBUF added.</li> <li>● OSCH added.</li> <li>● SPMI added.</li> <li>● I3C added.</li> <li>● Devices supported by OSC updated.</li> </ul>  |
| 10/20/2019 | 1.9E    | IOB, BSRAM, CLOCK modules updated.  |
| 11/28/2019 | 2.0E    | <ul style="list-style-type: none"> <li>● GSR and INV modules added in Miscellaneous.</li> <li>● Devices supported updated.</li> <li>● FLASH64KZ added and FLASH96KZ removed.</li> </ul>   |
| 01/16/2020 | 2.1E    | <ul style="list-style-type: none"> <li>● IODELAYA, rPLL, PLLVR, CLKDIV2 added.</li> <li>● DPB/DPX9B, SDPB/SDPX9B, rSDP/rSDPX9,</li> </ul>   |

| Date       | Version | Description  |
|------------|---------|--|
|            |         | rROM/rROMX9, pROM/pROMX9 added. <ul style="list-style-type: none"> <li>● EMCU, BANDGAP, FLASH64K added.</li> <li>● IODELAY, PLL, CLKDIV, OSC, DQCE updated.</li> <li>● Placement rule of FF、LATCH added.</li> <li>● GW2A-55C added.</li> <li>● GW1N-6/GW1N-9/GW1NR-9 disabled DP/DPX9, DPB/DPX9B.</li> <li>● Notes of register added in IOLOGIC.</li> <li>● GW1NZ-1 disabled 1, 2, 4, 8 bit width of DP/DPB and 9 bit width of DPX9/DPX9.</li> </ul>   |
| 03/09/2020 | 2.2E    | <ul style="list-style-type: none"> <li>● GW1NS-2, GW1NS-2C, GW1NSR-2, GW1NSR-2C, GW1NSE-2C disable DP/DPX9 and DPB/DPX9B.</li> <li>● OSCEN port description added in OSCF.</li> <li>● PLL/rPLL/PLLVR parameter description updated.</li> </ul>   |
| 06/08/2020 | 2.3E    | <ul style="list-style-type: none"> <li>● GW1N-2, GW1N-2B and GW1N-6 removed.</li> <li>● GW1N-9C and GW1NR-9C added.</li> <li>● IODELAYC, DHCENC and DCC added.</li> <li>● Functional description of MIPI_IBUF added.</li> <li>● MIPI_IBUF_HS, MIPI_IBUF_LP and DLL removed.</li> <li>● Port diagram of LUT5 and MUX8 added.</li> <li>● VCC and GND added.</li> <li>● PLLVR, FLASH64K, BUFS, EMPU and CLKDIV2 updated.</li> <li>● DP/DPX9, ROM/ROMX9, SDP/SDPX9, rSDP/rSDPX9, rROM/rROMX9 and PLL removed.</li> </ul> |
| 09/11/2020 | 2.4E    | The description of IP invoking of ADC, BANDGAP, SPMI and I3C modules added.  |
| 12/30/2020 | 2.5E    | The description of Chapter 2 CFU updated.  |
| 07/22/2021 | 2.6E    | <ul style="list-style-type: none"> <li>● activeFlash added.</li> <li>● Help information on IP GUI removed and figures updated.</li> <li>● GW1NZ-1C, GW1N-2, GW1N-2B, GW1N-1P5, GW1N-1P5B, GW1NR-2, GW1NR-2B added.</li> </ul>  |
| 10/28/2021 | 2.7E    | The description of activeFlash updated.  |

# Contents

|                              |            |
|------------------------------|------------|
| <b>Contents .....</b>        | <b>i</b>   |
| <b>List of Figures .....</b> | <b>ii</b>  |
| <b>List of Tables .....</b>  | <b>iii</b> |
| <b>1 IOB .....</b>           | <b>1</b>   |
| <b>2 CFU .....</b>           | <b>2</b>   |
| <b>3 Memory .....</b>        | <b>3</b>   |
| <b>4 DSP .....</b>           | <b>4</b>   |
| <b>5 Clock .....</b>         | <b>5</b>   |
| <b>6 User Flash .....</b>    | <b>6</b>   |
| <b>7 EMPU .....</b>          | <b>7</b>   |
| 7.1 MCU .....                | 7          |
| 7.2 EMCU .....               | 20         |
| 7.3 USB20_PHY .....          | 32         |
| 7.4 ADC .....                | 42         |
| <b>8 Miscellaneous .....</b> | <b>46</b>  |
| 8.1 GSR .....                | 46         |
| 8.2 INV .....                | 47         |
| 8.3 VCC .....                | 48         |
| 8.4 GND .....                | 49         |
| 8.5 BANDGAP .....            | 50         |
| 8.6 SPMI .....               | 52         |
| 8.7 I3C .....                | 57         |
| 8.8 activeFlash .....        | 65         |

# List of Figures

|  |    |
|--|----|
| Figure 7-1 MCU Port Diagram .....            | 8  |
| Figure 7-2 Port Diagram of EMCU .....        | 21 |
| Figure 7-3 USB20_PHY Port Diagram.....       | 33 |
| Figure 7-4 ADC Port Diagram .....            | 42 |
| Figure 7-5 IP Customization of ADC .....     | 44 |
| Figure 8-1 GSR Port Diagram.....             | 46 |
| Figure 8-2 INV Port Diagram .....            | 47 |
| Figure 8-3 VCC Port Diagram.....             | 48 |
| Figure 8-4 GND Port Diagram .....            | 49 |
| Figure 8-5 BANDGAP Port Diagram.....         | 50 |
| Figure 8-6 IP Customization of BandGap ..... | 51 |
| Figure 8-7 SPMI Port Diagram.....            | 52 |
| Figure 8-8 IP Customization of SPMI.....     | 55 |
| Figure 8-9 I3C Port Diagram.....             | 57 |
| Figure 8-10 IP Customization of I3C .....    | 64 |
| Figure 8-11 activeFlash Port Diagram .....   | 65 |

# List of Tables

|   |    |
|---|----|
| Table 7-1 MCU Devices Supported.....          | 7  |
| Table 7-2 Port Description.....               | 9  |
| Table 7-3 EMCU Devices Supported .....        | 20 |
| Table 7-4 Port Description.....               | 22 |
| Table 7-5 USB20_PHY Devices Supported.....    | 32 |
| Table 7-6 Port Description.....               | 33 |
| Table 7-7 Parameters.....                     | 35 |
| Table 7-8 ADC Devices Supported .....         | 42 |
| Table 7-9 Port Description.....               | 42 |
| Table 8-1 Port Description.....               | 46 |
| Table 8-2 Port Description.....               | 47 |
| Table 8-3 Port Description.....               | 48 |
| Table 8-4 Port Description.....               | 49 |
| Table 8-5 BANDGAP Devices Supported .....     | 50 |
| Table 8-6 Port Description.....               | 50 |
| Table 8-7 SPMI Devices Supported .....        | 52 |
| Table 8-8 Port Description.....               | 52 |
| Table 8-9 I3C Devices Supported .....         | 57 |
| Table 8-10 Port Description.....              | 57 |
| Table 8-11 activeFlash Devices Supported..... | 65 |
| Table 8-12 Port Description.....              | 65 |

# 1 IOB

IOB includes input/output buffer (IO Buffer) and input/output logic (IO Logic). For IO Buffer and IO Logic primitives, see [UG289](#), Gowin Programmable IO(GPIO) User Guide.



# 2<sub>CFU</sub>

The Configurable Function Unit (CFU) and the Configurable Logic Unit (CLU) are two basic units for FPGA core of GOWINSEMI. Each unit consists of four configurable logic sections (CLS) and its configurable routing unit (CRU). Configurable logical sections in CLU cannot be configured as SRAM, but as basic logic, ALU, and ROM. The configurable logic sections in the CFU can be configured as basic logic, ALU, SRAM, and ROM depending on the applications. For CFU primitives, you can see [UG288](#), Gowin Configurable Function Unit (CFU) User Guide.

# 3 Memory

Gowin FPGA products provide abundant memory resources, including Block Static Random Access Memory (BSRAM) and Shadow Static Random Access Memory (SSRAM). For BSRAM and SSRAM primitives, see [UG285](#), Gowin BSRAM & SSRAM User Guide.

# 4 DSP

Gowin FPGA products provide abundant DSP resources, which can meet the demand of high-performance digital signal processing. For DSP primitives, see [UG287](#), Gowin DSP User Guide.

# 5 Clock

Gowin FPGA products provide dedicated global clock (GCLK, including PCLK and SCLK) directly connected to all resources of the devices. In addition to GCLK, PLL, HCLK and bidirectional data strobe circuit (DQS) for DDR Memory are also available. For clock primitives, see [UG286](#), Gowin Clock User Guide.

# 6 User Flash

Gowin LittleBee® Family FPGA products provide User Flash. Different series of devices support different sizes of Flash. For the Flash primitives, see [UG295](#), Gowin Flash User Guide.

# 7 EMPU

## 7.1 MCU

### Primitive

ARM Cortex-M3 Microcontroller Unit (MCU) is a micro-processor based on the ARM Cortex-M3. The 32-bit AHB/APB bus is used. It supports the functions of two UARTs, two Timers and Watchdog. It also provides 16-bit GPIO, two UARTs, two JTAGs, two User Interrupt interfaces, AHB Flash read interface, AHB Sram read/write interface, two AHB bus extension interfaces, and one APB bus extension interface.

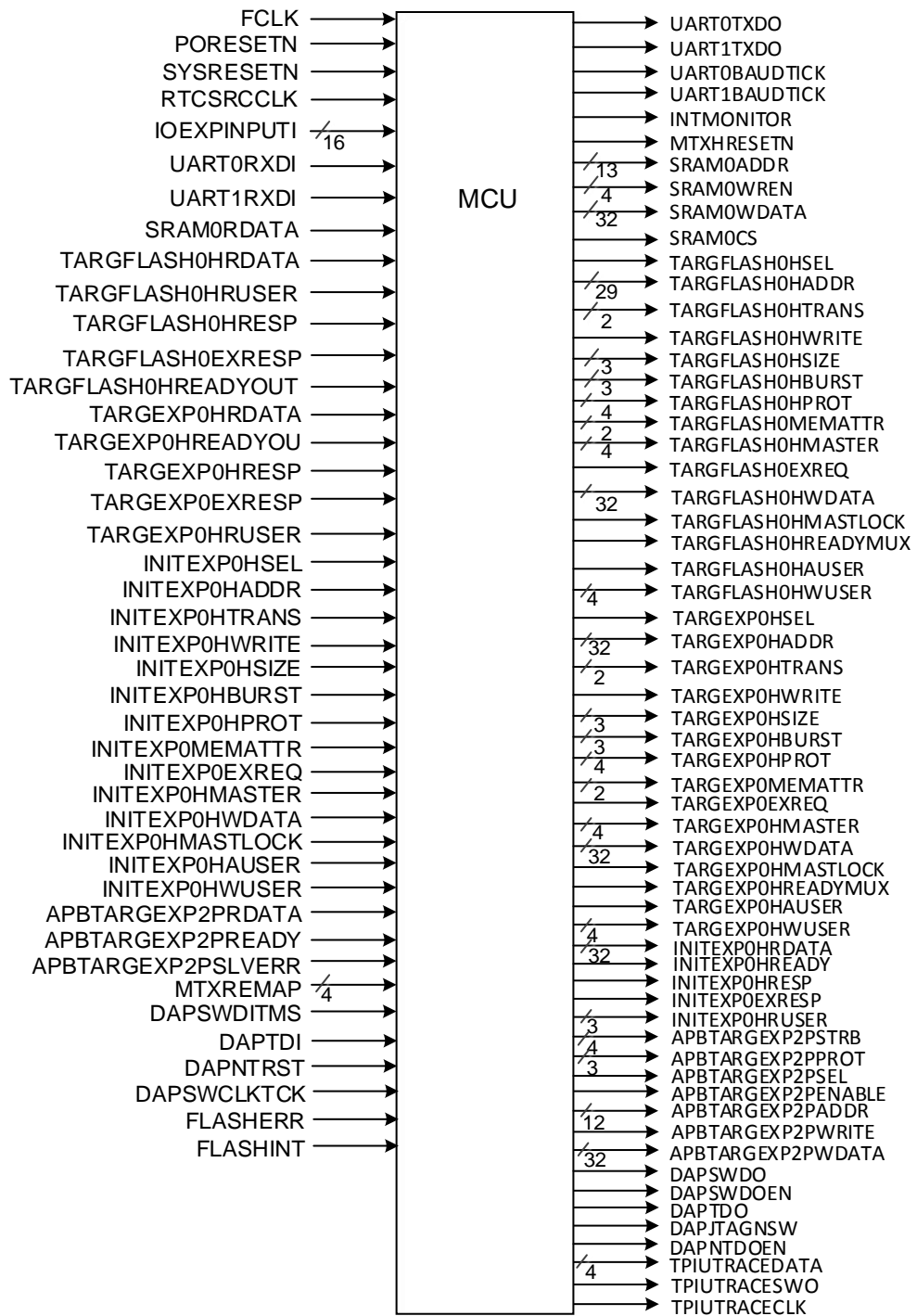
### Devices Supported

Table 7-1 MCU Devices Supported

| Family     | Series | Device    |
|------------|--------|-----------|
| LittleBee® | GW1NS  | GW1NS-2C  |
|            | GW1NSE | GW1NSE-2C |
|            | GW1NSR | GW1NSR-2C |

## Port Diagram

Figure 7-1 MCU Port Diagram



## Port Description

Table 7-2 Port Description

| Port                    | I/O   | Description  |
|-------------------------|-------|--|
| FCLK                    | input | Free running clock   |
| PORESETN                | input | Power on reset   |
| SYSRESETN               | input | System reset   |
| RTCSRCLK                | input | Used to generate RTC clock   |
| IOEXPINPUTI[15:0]       | input | IOEXPINPUTI  |
| UART0RXDI               | input | UART0RXDI  |
| UART1RXDI               | input | UART1RXDI  |
| SRAM0RDATA[31:0]        | input | SRAM Read data bus   |
| TARGFLASH0HRDATA[31:0]  | input | TARGFLASH0, HRDATA   |
| TARGFLASH0HRUSER[2:0]   | input | TARGFLASH0, HRUSER   |
| TARGFLASH0HRESP         | input | TARGFLASH0, HRESP  |
| TARGFLASH0EXRESP        | input | TARGFLASH0, EXRESP   |
| TARGFLASH0HREADYOUT     | input | TARGFLASH0, EXRESP   |
| TARGEXP0HRDATA[31:0]    | input | TARGEXP0, HRDATA   |
| TARGEXP0HREADYOUT       | input | TARGEXP0, HREADY   |
| TARGEXP0HRESP           | input | TARGEXP0, HRESP  |
| TARGEXP0EXRESP          | input | TARGEXP0, EXRESP   |
| TARGEXP0HRUSER[2:0]     | input | TARGEXP0, HRUSER   |
| INITEXP0HSEL            | input | INITEXP0, HSELx  |
| INITEXP0HADDR[31:0]     | input | INITEXP0, HADDR  |
| INITEXP0HTRANS[1:0]     | input | INITEXP0, HTRANS   |
| INITEXP0HWRITE          | input | INITEXP0, HWRITE   |
| INITEXP0HSIZE[2:0]      | input | INITEXP0, HSIZE  |
| INITEXP0HBURST[2:0]     | input | INITEXP0, HBURST   |
| INITEXP0HPROT[3:0]      | input | INITEXP0, HPROT  |
| INITEXP0MEMATTR[1:0]    | input | INITEXP0, MEMATTR  |
| INITEXP0EXREQ           | input | INITEXP0, EXREQ  |
| INITEXP0HMASTER[3:0]    | input | INITEXP0, HMASTER  |
| INITEXP0HWDATA[31:0]    | input | INITEXP0, HWDATA   |
| INITEXP0HMASTLOCK       | input | INITEXP0, HMASTLOCK  |
| INITEXP0HAUSER          | input | INITEXP0, HAUSER   |
| INITEXP0HWUSER[3:0]     | input | INITEXP0, HWUSER   |
| APBTARGEXP2PRDATA[31:0] | input | APBTARGEXP2, PRDATA  |
| APBTARGEXP2PREADY       | input | APBTARGEXP2, PREADY  |
| APBTARGEXP2PSLVERR      | input | APBTARGEXP2, PSLVERR   |
| MTXREMAP[3:0]           | input | The MTXREMAP signals control the remapping of the boot memory range. |
| DAPSWDITMS              | input | Debug TMS  |



| Port                   | I/O    | Description  |
|------------------------|--------|--|
| DAPTDI                 | input  | Debug TDI  |
| DAPNTRST               | input  | Test reset   |
| DAPSWCLKTCK            | input  | Test clock / SWCLK                                     |
| FLASHERR               | input  | Output clock, used by the TPA to sample the other pins |
| FLASHINT               | input  | Output clock, used by the TPA to sample the other pins |
| IOEXPOUTPUTO[15:0]     | output | IOEXPOUTPUTO   |
| IOEXPOUTPUTENO[15:0]   | output | IOEXPOUTPUTENO   |
| UART0TXDO              | output | UART0TXDO  |
| UART1TXDO              | output | UART1TXDO  |
| UART0BAUDTICK          | output | UART0BAUDTICK  |
| UART1BAUDTICK          | output | UART1BAUDTICK  |
| INTMONITOR             | output | INTMONITOR   |
| MTXHRESETN             | output | SRAM/Flash Chip reset                                  |
| SRAM0ADDR[12:0]        | output | SRAM address   |
| SRAM0WREN[3:0]         | output | SRAM Byte write enable                                 |
| SRAM0WDATA[31:0]       | output | SRAM Write data  |
| SRAM0CS                | output | SRAM Chip select                                       |
| TARGFLASH0HSEL         | output | TARGFLASH0, HSELx                                      |
| TARGFLASH0HADDR[28:0]  | output | TARGFLASH0, HADDR                                      |
| TARGFLASH0HTRANS[1:0]  | output | TARGFLASH0, HTRANS                                     |
| TARGFLASH0HWRITE       | output | TARGFLASH0, HWRITE                                     |
| TARGFLASH0HSIZE[2:0]   | output | TARGFLASH0, HSIZE                                      |
| TARGFLASH0HBURST[2:0]  | output | TARGFLASH0, HBURST                                     |
| TARGFLASH0HPROT[3:0]   | output | TARGFLASH0, HPROT                                      |
| TARGFLASH0MEMATTR[1:0] | output | TARGFLASH0, MEMATTR                                    |
| TARGFLASH0EXREQ        | output | TARGFLASH0, EXREQ                                      |
| TARGFLASH0HMASTER[3:0] | output | TARGFLASH0, HMASTER                                    |
| TARGFLASH0HWDATA[31:0] | output | TARGFLASH0, HWDATA                                     |
| TARGFLASH0HMASTLOCK    | output | TARGFLASH0, HMASTLOCK                                  |
| TARGFLASH0HREADYMUX    | output | TARGFLASH0, HREADYOUT                                  |
| TARGFLASH0HAUSER       | output | TARGFLASH0, HAUSER                                     |
| TARGFLASH0HWUSER[3:0]  | output | TARGFLASH0, HWUSER                                     |
| TARGEXP0HSEL           | output | TARGEXP0, HSELx  |
| TARGEXP0HADDR[31:0]    | output | TARGEXP0, HADDR  |
| TARGEXP0HTRANS[1:0]    | output | TARGEXP0, HTRANS                                       |
| TARGEXP0HWRITE         | output | TARGEXP0, HWRITE                                       |
| TARGEXP0HSIZE[2:0]     | output | TARGEXP0, HSIZE  |
| TARGEXP0HBURST[2:0]    | output | TARGEXP0, HBURST                                       |
| TARGEXP0HPROT[3:0]     | output | TARGEXP0, HPROT  |
| TARGEXP0MEMATTR[1:0]   | output | TARGEXP0, MEMATTR                                      |

| Port                    | I/O    | Description  |
|-------------------------|--------|--|
| TARGEXP0EXREQ           | output | TARGEXP0, EXREQ  |
| TARGEXP0HMASTER[3:0]    | output | TARGEXP0, HMASTER  |
| TARGEXP0HWDATA[31:0]    | output | TARGEXP0, HWDATA   |
| TARGEXP0HMASTLOCK       | output | TARGEXP0, HMASTLOCK                                      |
| TARGEXP0HREADYMUX       | output | TARGEXP0, HREADYOUT                                      |
| TARGEXP0HAUSER          | output | TARGEXP0, HAUSER   |
| TARGEXP0HWUSER[3:0]     | output | TARGEXP0, HWUSER   |
| INITEXP0HRDATA[31:0]    | output | INITEXP0, HRDATA   |
| INITEXP0HREADY          | output | INITEXP0, HREADY   |
| INITEXP0HRESP           | output | INITEXP0, HRESP  |
| INITEXP0EXRESP          | output | INITEXP0, EXRESP   |
| INITEXP0HRUSER[2:0]     | output | INITEXP0, HRUSER   |
| APBTARGEXP2PSTRB[3:0]   | output | APBTARGEXP2, PSTRB                                       |
| APBTARGEXP2PPROT[2:0]   | output | APBTARGEXP2, PPROT                                       |
| APBTARGEXP2PSEL         | output | APBTARGEXP2, PSELx                                       |
| APBTARGEXP2PENABLE      | output | APBTARGEXP2, PENABLE                                     |
| APBTARGEXP2PADDR[11:0]  | output | APBTARGEXP2, PADDR                                       |
| APBTARGEXP2PWRITE       | output | APBTARGEXP2, PWRITE                                      |
| APBTARGEXP2PWDATA[31:0] | output | APBTARGEXP2, PWDATA                                      |
| DAPSWDO                 | output | Serial Wire Data Out                                     |
| DAPSWDOEN               | output | Serial Wire Output Enable                                |
| DAPTDO                  | output | Debug TDO  |
| DAPJTAGNSW              | output | JTAG or Serial-Wire selection JTAG mode(1) or SW mode(0) |
| DAPNTDOEN               | output | TDO output pad control signal                            |
| TPIUTRACEDATA[3:0]      | output | Output data  |
| TPIUTRACESWO            | output | Serial Wire Viewer data                                  |
| TPIUTRACECLK            | output | Output clock, used by the TPA to sample the other pins   |

### Primitive Instantiation

#### Verilog Instantiation:

```
MCU u_sse050_top_syn (
    .FCLK(fclk),
    .PORESETN(poresetn),
    .SYSRESETN(sysresetn),
    .RTCSRCLK(rtsrcclk),
    .IOEXPINPUTI(ioexpinputi[15:0]),
    .IOEXPOUTPUTO(ioexpoutputo[15:0]),
```

```

.IOEXPOUTPUTPUTENO(ioexpoutputputeno[15:0]),
.UART0RXDI(uart0rxdi),
.UART0TXDO(uart0txdo),
.UART1RXDI(uart1rxdi),
.UART1TXDO(uart1txdo),
.SRAM0RDATA(sram0rdata[31:0]),
.SRAM0ADDR(sram0addr[12:0]),
.SRAM0WREN(sram0wren[3:0]),
.SRAM0WDATA(sram0wdata[31:0]),
.SRAM0CS(sram0cs),
.MTXHRESETN(mtxhreset),
.TARGFLASH0HSEL(targflash0hsel),
.TARGFLASH0HADDR(targflash0haddr[28:0]),
.TARGFLASH0HTRANS(targflash0htrans[1:0]),
.TARGFLASH0HWRITE(targflash0hwrite),
.TARGFLASH0HSIZE(targflash0hsize[2:0]),
.TARGFLASH0HBURST(targflash0hburst[2:0]),
.TARGFLASH0HPROT(targflash0hprot[3:0]),
.TARGFLASH0MEMATTR(targflash0memattr[1:0]),
.TARGFLASH0EXREQ(targflash0exreq),
.TARGFLASH0HMASTER(targflash0hmaster[3:0]),
.TARGFLASH0HWDATA(targflash0hwddata[31:0]),
.TARGFLASH0HMASTLOCK(targflash0hmastlock),
.TARGFLASH0HREADYMUX(targflash0hreadymux),
.TARGFLASH0HAUSER(targflash0hauser),
.TARGFLASH0HWUSER(targflash0hwuser[3:0]),
.TARGFLASH0HRDATA(targflash0hrdata[31:0]),
.TARGFLASH0HRUSER(targflash0hruser[2:0]),
.TARGFLASH0HRESP(targflash0hresp),
.TARGFLASH0EXRESP(targflash0exresp),
.TARGFLASH0HREADYOUT(targflash0hreadyout),
.TARGEXP0HSEL(targexp0hsel),
.TARGEXP0HADDR(targexp0haddr[31:0]),
.TARGEXP0HTRANS(targexp0htrans[1:0]),
.TARGEXP0HWRITE(targexp0hwrite),
.TARGEXP0HSIZE(targexp0hsize[2:0]),
.TARGEXP0HBURST(targexp0hburst[2:0]),
.TARGEXP0HPROT(targexp0hprot[3:0]),

```

```

.TARGEXP0MEMATTR(targexp0memattr[1:0]),
.TARGEXP0EXREQ(targexp0exreq),
.TARGEXP0HMASTER(targexp0hmaster[3:0]),
.TARGEXP0HWDATA(targexp0hwdata[31:0]),
.TARGEXP0HMASTLOCK(targexp0hmastlock),
.TARGEXP0HREADYMUX(targexp0hreadymux),
.TARGEXP0HAUSER(targexp0hauser),
.TARGEXP0HWUSER(targexp0hwuser[3:0]),
.TARGEXP0HRDATA(targexp0hrdata[31:0]),
.TARGEXP0HREADYOUT(targexp0hreadyout),
.TARGEXP0HRESP(targexp0hresp),
.TARGEXP0EXRESP(targexp0exresp),
.TARGEXP0HRUSER(targexp0hruser[2:0]),
.INITEXP0HSEL(initexp0hsel),
.INITEXP0HADDR(initexp0haddr[31:0]),
.INITEXP0HTRANS(initexp0htrans[1:0]),
.INITEXP0HWRITE(initexp0hwrite),
.INITEXP0HSIZE(initexp0hsize[2:0]),
.INITEXP0HBURST(initexp0hburst[2:0]),
.INITEXP0HPROT(initexp0hprot[3:0]),
.INITEXP0MEMATTR(initexp0memattr[1:0]),
.INITEXP0EXREQ(initexp0exreq),
.INITEXP0HMASTER(initexp0hmaster[3:0]),
.INITEXP0HWDATA(initexp0hwdata[31:0]),
.INITEXP0HMASTLOCK(initexp0hmastlock),
.INITEXP0HAUSER(initexp0hauser),
.INITEXP0HWUSER(initexp0hwuser[3:0]),
.INITEXP0HRDATA(initexp0hrdata[31:0]),
.INITEXP0HREADY(initexp0hready),
.INITEXP0HRESP(initexp0hresp),
.INITEXP0EXRESP(initexp0exresp),
.INITEXP0HRUSER(initexp0hruser[2:0]),
.APBTARGEXP2PSEL(apbtargexp2psel),
.APBTARGEXP2PENABLE(apbtargexp2penable),
.APBTARGEXP2PADDR(apbtargexp2paddr[11:0]),
.APBTARGEXP2PWRITE(apbtargexp2pwrite),
.APBTARGEXP2PWDATA(apbtargexp2pwwdata[31:0]),
.APBTARGEXP2PRDATA(apbtargexp2prdata[31:0]),

```

```

.APBTARGEXP2PREADY(apbtargexp2pready),
.APBTARGEXP2PSLVERR(apbtargexp2pslverr),
.APBTARGEXP2PSTRB(apbtargexp2pstrb[3:0]),
.APBTARGEXP2PPROT(apbtargexp2pprot[2:0]),
.MTXREMAP(mtxremap[3:0]),
.DAPSWDITMS(dapswditms),
.DAPSWDO(dapswdo),
.DAPSWDOEN(dapswdoen),
.DAPTDI(daptdi),
.DAPTDO(daptdo),
.DAPNTRST(dapntrst),
.DAPSWCLKTCK(dapswclk_tck),
.DAPNTDOEN(dapntdoen),
.DAPJTAGNSW(dapjtagnew),
.TPIUTRACEDATA(tpiutracedata[3:0]),
.TPIUTRACESWO(tpiutraceswo),
.TPIUTRACECLK(tpiutracedclk),
.FLASHERR(flasherr),
.FLASHINT(flashint)
);

```

**Vhdl Instantiation:**

COMPONENT MCU

```

    PORT(
FCLK:IN std_logic;
PORESETN:IN std_logic;
SYSRESETN:IN std_logic;
RTCSRCLK:IN std_logic;
UART0RXDI:IN std_logic;
UART1RXDI:IN std_logic;
CLK:IN std_logic;
RESET:IN std_logic;
IOEXPINPUTI:IN std_logic_vector(15 downto 0);
SRAM0RDATA:IN std_logic_vector(31 downto 0);
TARGFLASH0HRDATA:IN std_logic_vector(31 downto 0);
TARGFLASH0HRUSER:IN std_logic_vector(2 downto 0);
TARGFLASH0HRESP:IN std_logic;
TARGFLASH0EXRESP:IN std_logic;
TARGFLASH0HREADYOUT:IN std_logic;

```

TARGEXP0HRDATA: IN std\_logic\_vector(31 downto 0);  
TARGEXP0HREADYOUT: IN std\_logic;  
TARGEXP0HRESP: IN std\_logic;  
TARGEXP0EXRESP: IN std\_logic;  
TARGEXP0HRUSER: IN std\_logic\_vector(2 downto 0);  
INITEXP0HSEL: IN std\_logic;  
INITEXP0HADDR: IN std\_logic\_vector(31 downto 0);  
INITEXP0HTRANS: IN std\_logic\_vector(1 downto 0);  
INITEXP0HWRITE: IN std\_logic;  
INITEXP0HSIZE: IN std\_logic\_vector(2 downto 0);  
INITEXP0HBURST: IN std\_logic\_vector(2 downto 0);  
INITEXP0HPROT: IN std\_logic\_vector(3 downto 0);  
INITEXP0MEMATTR: IN std\_logic\_vector(1 downto 0);  
INITEXP0EXREQ: IN std\_logic;  
INITEXP0HMASTER: IN std\_logic\_vector(3 downto 0);  
INITEXP0HWDATA: IN std\_logic\_vector(31 downto 0);  
INITEXP0HMASTLOCK: IN std\_logic;  
INITEXP0HAUSER: IN std\_logic;  
INITEXP0HWUSER: IN std\_logic\_vector(3 downto 0);  
APBTARGEXP2PRDATA: IN std\_logic\_vector(3 downto 0);  
APBTARGEXP2PREADY: IN std\_logic;  
APBTARGEXP2PSLVERR: IN std\_logic;  
MTXREMAP: IN std\_logic\_vector(3 downto 0);  
DAPSWDITMS: IN std\_logic;  
DAPTDI: IN std\_logic;  
DAPNTRST: IN std\_logic;  
DAPSWCLKTCK: IN std\_logic;  
FLASHERR: IN std\_logic;  
FLASHINT: IN std\_logic;  
IOEXPOUTPUTO: OUT std\_logic\_vector(15 downto 0);  
IOEXPOUTPUTENO: OUT std\_logic\_vector(15 downto 0);  
IOEXPINPUTI: OUT std\_logic\_vector(15 downto 0);  
UART0TXDO: OUT std\_logic;  
UART1TXDO: OUT std\_logic;  
UART0BAUDTICK: OUT std\_logic;  
UART1BAUDTICK: OUT std\_logic;  
INTMONITOR: OUT std\_logic;  
MTXHRESETN: OUT std\_logic;

```
SRAM0ADDR:OUT std_logic_vector(12 downto 0);
SRAM0WREN:OUT std_logic_vector(3 downto 0);
SRAM0WDATA:OUT std_logic_vector(31 downto 0);
SRAM0CS:  OUT std_logic;
TARGFLASH0HSEL:  OUT std_logic;
TARGFLASH0HWRITE:  OUT std_logic;
TARGFLASH0EXREQ:  OUT std_logic;
TARGFLASH0HMASTLOCK:  OUT std_logic;
TARGFLASH0HREADYMUX:  OUT std_logic;
TARGFLASH0HAUSER:  OUT std_logic;
SRAM0RDATA:OUT std_logic_vector(31 downto 0);
TARGFLASH0HADDR:OUT std_logic_vector(28 downto 0);
TARGFLASH0HTRANS:OUT std_logic_vector(1 downto 0);
TARGFLASH0HSIZE:OUT std_logic_vector(2 downto 0);
TARGFLASH0HBURST:OUT std_logic_vector(2 downto 0);
TARGFLASH0HPROT:OUT std_logic_vector(3 downto 0);
TARGFLASH0MEMATTR:OUT std_logic_vector(1 downto 0);
TARGFLASH0HMASTER:OUT std_logic_vector(3 downto 0);
TARGFLASH0HWDATA:OUT std_logic_vector(31 downto 0);
TARGFLASH0HWUSER:OUT std_logic_vector(3 downto 0);
TARGFLASH0HRDATA:OUT std_logic_vector(31 downto 0);
TARGEXP0HADDR:OUT std_logic_vector(31 downto 0);
TARGEXP0HSEL:  OUT std_logic;
TARGEXP0HWRITE:  OUT std_logic;
TARGEXP0EXREQ:  OUT std_logic;
TARGEXP0HMASTLOCK:  OUT std_logic;
TARGEXP0HREADYMUX:  OUT std_logic;
TARGEXP0HAUSER:  OUT std_logic;
INITEXP0HREADY:  OUT std_logic;
INITEXP0HRESP:  OUT std_logic;
INITEXP0EXRESP:  OUT std_logic;
TARGEXP0HTRANS:OUT std_logic_vector(1 downto 0);
TARGEXP0HSIZE:OUT std_logic_vector(2 downto 0);
TARGEXP0HBURST:OUT std_logic_vector(2 downto 0);
TARGEXP0HPROT:OUT std_logic_vector(3 downto 0);
TARGEXP0MEMATTR:OUT std_logic_vector(1 downto 0);
TARGEXP0HMASTER:OUT std_logic_vector(3 downto 0);
TARGEXP0HWDATA:OUT std_logic_vector(31 downto 0);
```

```

    TARGEXP0HWUSER:OUT std_logic_vector(3 downto 0);
    INITEXP0HRDATA:OUT std_logic_vector(31 downto 0);
    INITEXP0HRUSER:OUT std_logic_vector(2 downto 0);
    APBTARGEXP2PSTRB:OUT std_logic_vector(3 downto 0);
    APBTARGEXP2PPROT:OUT std_logic_vector(2 downto 0);
    APBTARGEXP2PADDR:OUT std_logic_vector(11 downto 0);
    APBTARGEXP2PWDATA:OUT std_logic_vector(31 downto 0);
    TPIUTRACEDATA:OUT std_logic_vector(3 downto 0);
    APBTARGEXP2PSEL:  OUT std_logic;
    APBTARGEXP2PENABLE:  OUT std_logic;
    APBTARGEXP2PWRITE:  OUT std_logic;
    DAPSWDO:  OUT std_logic;
    DAPSWDOEN:  OUT std_logic;
    DAPTD0:  OUT std_logic;
    DAPJTAGNSW:  OUT std_logic;
    DAPNTDOEN:  OUT std_logic;
    TPIUTRACESWO:  OUT std_logic;
    TPIUTRACECLK:  OUT std_logic;
);
END COMPONENT;

```

```

uut: MCU
  PORT MAP (
    FCLK=> fclk;
    PORESETN=> poresetn;
    SYSRESETN=> sysresetn;
    RTCSRCLK=> rtsrcclk;
    UART0RXDI=> uart0rxdi;
    UART1RXDI=> uart1rxdi;
    CLK=>clk,
    RESET=>reset,
    IOEXPINPUTI=>ioexpinputi,
    SRAM0RDATA=>sram0rdata,
    TARGFLASH0HRDATA=>targflash0hrdata,
    TARGFLASH0HRUSER=>targflash0hruser,
    TARGFLASH0HRESP=>targflash0hresp,
    TARGFLASH0EXRESP=>targflash0exresp,
    TARGFLASH0HREADYOUT=>targflash0hreadyout,

```



TARGEXP0HRDATA=>targexp0hrdata,  
TARGEXP0HREADYOUT=>targexp0hreadyout,  
TARGEXP0HRESP=>targexp0hresp,  
TARGEXP0EXRESP=>targexp0exresp,  
TARGEXP0HRUSER=>targexp0hruser,  
INITEXP0HSEL=>initexp0hsel,  
INITEXP0HADDR=>initexp0haddr,  
INITEXP0HTRANS=>initexp0htrans,  
INITEXP0HWRITE=>initexp0hwrite,  
INITEXP0HSIZE=>initexp0hsize,  
INITEXP0HBURST=>initexp0hburst,  
INITEXP0HPROT=>initexp0hprot,  
INITEXP0MEMATTR=>initexp0memattr,  
INITEXP0EXREQ=>initexp0exreq,  
INITEXP0HMASTER=>initexp0hmaster,  
INITEXP0HWDATA=>initexp0hwdata,  
INITEXP0HMASTLOCK=>initexp0hmastlock,  
INITEXP0HAUSER=>initexp0hauser,  
INITEXP0HWUSER=>initexp0hwuser,  
APBTARGEXP2PRDATA=>apbtargexp2prdata,  
APBTARGEXP2PREADY=>apbtargexp2pready,  
APBTARGEXP2PSLVERR=>apbtargexp2pslverr,  
MTXREMAP=>mtxremap,  
DAPSWDITMS=>dapswditms,  
DAPTDI=>dapti,  
DAPNTRST=>dapntrst,  
DAPSWCLKTCK=>dapswclktck,  
FLASHERR=>flasherr,  
FLASHINT=>flashint,  
IOEXPOUTPUTO=>ioexpoutputo,  
IOEXPOUTPUTENO=>ioexpoutputeno,  
IOEXPINPUTI=>ioexpinputi,  
UART0TXDO=>uart0txdo,  
UART1TXDO=>uart1txdo,  
UART0BAUDTICK=>uart0baudtick,  
UART1BAUDTICK=>uart1baudtick,  
INTMONITOR=>intmonitor,  
MTXHRESETN=>mtxhresetn,

SRAM0ADDR=>sram0addr,  
SRAM0WREN=>sram0wren,  
SRAM0WDATA=>sram0wdata,  
SRAM0CS=>sram0cs,  
TARGFLASH0HSEL=>targflash0hsel,  
TARGFLASH0HWRITE=>targflash0hwrite,  
TARGFLASH0EXREQ=>targflash0exreq,  
TARGFLASH0HMASTLOCK=>targflash0hmastlock,  
TARGFLASH0HREADYMUX=>targflash0hreadymux,  
TARGFLASH0HAUSER=>targflash0hauser,  
SRAM0ORDATA=>sram0ordata,  
TARGFLASH0HADDR=>targflash0haddr,  
TARGFLASH0HTRANS=>targflash0htrans,  
TARGFLASH0HSIZE=>targflash0hsize,  
TARGFLASH0HBURST=>targflash0hburst,  
TARGFLASH0HPROT=>targflash0hprot,  
TARGFLASH0MEMATTR=>targflash0memattr,  
TARGFLASH0HMASTER=>targflash0hmaster,  
TARGFLASH0HWDATA=>targflash0hwdata,  
TARGFLASH0HWUSER=>targflash0hwuser,  
TARGFLASH0HRDATA=>targflash0hrdata,  
TARGEXP0HADDR=>targexp0haddr,  
TARGEXP0HSEL=>targexp0hsel,  
TARGEXP0HWRITE=>targexp0hwrite,  
TARGEXP0EXREQ=>targexp0exreq,  
TARGEXP0HMASTLOCK=>targexp0hmastlock,  
TARGEXP0HREADYMUX=>targexp0hreadymux,  
TARGEXP0HAUSER=>targexp0hauser,  
INITEXP0HREADY=>initexp0hready,  
INITEXP0HRESP=>initexp0hresp,  
INITEXP0EXRESP=>initexp0exresp,  
TARGEXP0HTRANS=>targexp0htrans,  
TARGEXP0HSIZE=>targexp0hsize,  
TARGEXP0HBURST=>targexp0hburst,  
TARGEXP0HPROT=>targexp0hprot,  
TARGEXP0MEMATTR=>targexp0memattr,  
TARGEXP0HMASTER=>targexp0hmaster,  
TARGEXP0HWDATA=>targexp0hwdata,

```

TARGEXP0HWUSER=>targexp0hwuser,
INITEXP0HRDATA=>initexp0hrdata,
INITEXP0HRUSER=>initexp0hruser,
APBTARGEXP2PSTRB=>apbtargexp2pstrb,
APBTARGEXP2PPROT=>apbtargexp2pprot,
APBTARGEXP2PADDR=>apbtargexp2paddr,
APBTARGEXP2PWDATA=>apbtargexp2pwdata,
TPIUTRACEDATA=>tpiutracedata,
APBTARGEXP2PSEL=>apbtargexp2psel,
APBTARGEXP2PENABLE=>apbtargexp2penable,
APBTARGEXP2PWRITE=>apbtargexp2pwrite,
DAPSWDO=>dapswdo,
DAPSWDOEN=>dapswdoen,
DAPTDO=>daptdo,
DAPJTAGNSW=>dapjtagnew,
DAPNTDOEN=>dapntdoen,
TPIUTRACESWO=>tpiutraceswo,
TPIUTRACECLK=>tpiutraceclk );

```

## 7.2 EMCU

### Primitive

ARM Cortex-M3 Microcontroller Unit (EMCU) is a micro-processor based on the ARM Cortex-m3. The 32-bit AHB/APB bus is used. It supports the functions of two UARTs, two Timers and Watchdog. It also provides 16-bit GPIO, two UARTs and JTAGs, six User Interrupt interfaces, AHB Flash read interface, AHB Sram read/write interface, two AHB bus extension interfaces, and one APB bus extension interface.

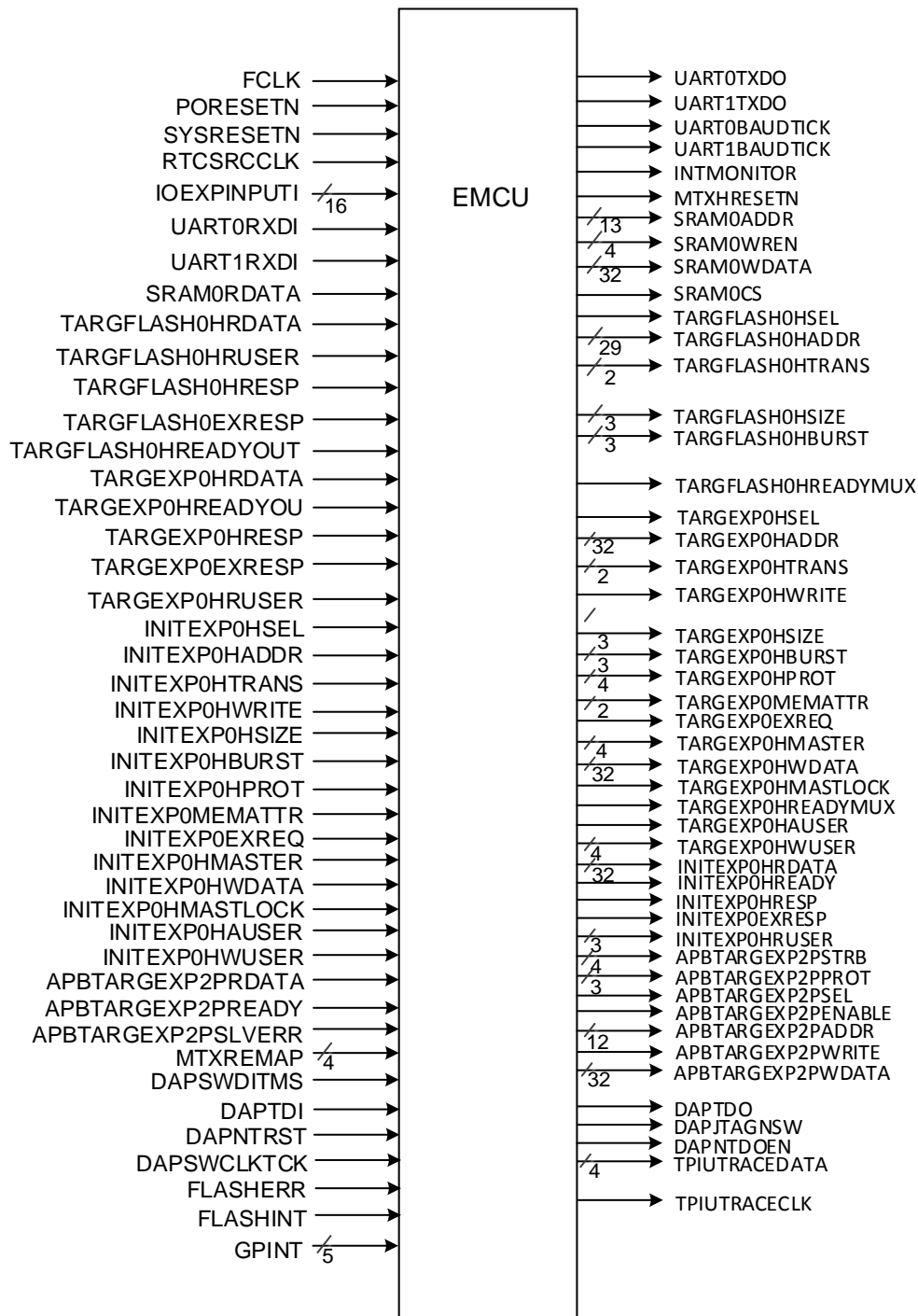
### Devices Supported

Table 7-3 EMCU Devices Supported

| Family     | Series  | Device     |
|------------|---------|------------|
| LittleBee® | GW1NS   | GW1NS-4C   |
|            | GW1NSR  | GW1NSR-4C  |
|            | GW1NSER | GW1NSER-4C |

## Port Diagram

Figure 7-2 Port Diagram of EMCU



## Port Description

Table 7-4 Port Description

| Port                    | I/O   | Description  |
|-------------------------|-------|--|
| FCLK                    | input | Free running clock   |
| PORESETN                | input | Power on reset   |
| SYSRESETN               | input | System reset   |
| RTCSRCCLK               | input | Used to generate RTC clock   |
| IOEXPINPUTI[15:0]       | input | IOEXPINPUTI  |
| UART0RXDI               | input | UART0RXDI  |
| UART1RXDI               | input | UART1RXDI  |
| SRAM0RDATA[31:0]        | input | SRAM Read data bus   |
| TARGFLASH0HRDATA[31:0]  | input | TARGFLASH0, HRDATA   |
| TARGFLASH0HRUSER[2:0]   | input | TARGFLASH0, HRUSER   |
| TARGFLASH0HRESP         | input | TARGFLASH0, HRESP  |
| TARGFLASH0EXRESP        | input | TARGFLASH0, EXRESP   |
| TARGFLASH0HREADYOUT     | input | TARGFLASH0, EXRESP   |
| TARGEXP0HRDATA[31:0]    | input | TARGEXP0, HRDATA   |
| TARGEXP0HREADYOUT       | input | TARGEXP0, HREADY   |
| TARGEXP0HRESP           | input | TARGEXP0, HRESP  |
| TARGEXP0EXRESP          | input | TARGEXP0, EXRESP   |
| TARGEXP0HRUSER[2:0]     | input | TARGEXP0, HRUSER   |
| INITEXP0HSEL            | input | INITEXP0, HSELx  |
| INITEXP0HADDR[31:0]     | input | INITEXP0, HADDR  |
| INITEXP0HTRANS[1:0]     | input | INITEXP0, HTRANS   |
| INITEXP0HWRITE          | input | INITEXP0, HWRITE   |
| INITEXP0HSIZE[2:0]      | input | INITEXP0, HSIZE  |
| INITEXP0HBURST[2:0]     | input | INITEXP0, HBURST   |
| INITEXP0HPROT[3:0]      | input | INITEXP0, HPROT  |
| INITEXP0MEMATTR[1:0]    | input | INITEXP0, MEMATTR  |
| INITEXP0EXREQ           | input | INITEXP0, EXREQ  |
| INITEXP0HMASTER[3:0]    | input | INITEXP0, HMASTER  |
| INITEXP0HWDATA[31:0]    | input | INITEXP0, HWDATA   |
| INITEXP0HMASTLOCK       | input | INITEXP0, HMASTLOCK  |
| INITEXP0HAUSER          | input | INITEXP0, HAUSER   |
| INITEXP0HWUSER[3:0]     | input | INITEXP0, HWUSER   |
| APBTARGEXP2PRDATA[31:0] | input | APBTARGEXP2, PRDATA  |
| APBTARGEXP2PREADY       | input | APBTARGEXP2, PREADY  |
| APBTARGEXP2PSLVERR      | input | APBTARGEXP2, PSLVERR   |
| MTXREMAP[3:0]           | input | The MTXREMAP signals control the remapping of the boot memory range. |
| DAPSWDITMS              | input | Debug TMS  |

| Port                  | I/O    | Description  |
|-----------------------|--------|--|
| DAPTDI                | input  | Debug TDI  |
| DAPNTRST              | input  | Test reset   |
| DAPSWCLKTCK           | input  | Test clock / SWCLK                                     |
| FLASHERR              | input  | Output clock, used by the TPA to sample the other pins |
| FLASHINT              | input  | Output clock, used by the TPA to sample the other pins |
| GPINT                 | input  | GPINT  |
| IOEXPOUTPUTO[15:0]    | output | IOEXPOUTPUTO   |
| IOEXPOUTPUTENO[15:0]  | output | IOEXPOUTPUTENO   |
| UART0TXDO             | output | UART0TXDO  |
| UART1TXDO             | output | UART1TXDO  |
| UART0BAUDTICK         | output | UART0BAUDTICK  |
| UART1BAUDTICK         | output | UART1BAUDTICK  |
| INTMONITOR            | output | INTMONITOR   |
| MTXHRESETN            | output | SRAM/Flash Chip reset                                  |
| SRAM0ADDR[12:0]       | output | SRAM address   |
| SRAM0WREN[3:0]        | output | SRAM Byte write enable                                 |
| SRAM0WDATA[31:0]      | output | SRAM Write data  |
| SRAM0CS               | output | SRAM Chip select                                       |
| TARGFLASH0HSEL        | output | TARGFLASH0, HSELx                                      |
| TARGFLASH0HADDR[28:0] | output | TARGFLASH0, HADDR                                      |
| TARGFLASH0HTRANS[1:0] | output | TARGFLASH0, HTRANS                                     |
| TARGFLASH0HSIZE[2:0]  | output | TARGFLASH0, HSIZE                                      |
| TARGFLASH0HBURST[2:0] | output | TARGFLASH0, HBURST                                     |
| TARGFLASH0HREADYMUX   | output | TARGFLASH0, HREADYOUT                                  |
| TARGEXP0HSEL          | output | TARGEXP0, HSELx  |
| TARGEXP0HADDR[31:0]   | output | TARGEXP0, HADDR  |
| TARGEXP0HTRANS[1:0]   | output | TARGEXP0, HTRANS                                       |
| TARGEXP0HWRITE        | output | TARGEXP0, HWRITE                                       |
| TARGEXP0HSIZE[2:0]    | output | TARGEXP0, HSIZE  |
| TARGEXP0HBURST[2:0]   | output | TARGEXP0, HBURST                                       |
| TARGEXP0HPROT[3:0]    | output | TARGEXP0, HPROT  |
| TARGEXP0MEMATTR[1:0]  | output | TARGEXP0, MEMATTR                                      |
| TARGEXP0EXREQ         | output | TARGEXP0, EXREQ  |
| TARGEXP0HMASTER[3:0]  | output | TARGEXP0, HMASTER                                      |
| TARGEXP0HWDATA[31:0]  | output | TARGEXP0, HWDATA                                       |
| TARGEXP0HMASTLOCK     | output | TARGEXP0, HMASTLOCK                                    |
| TARGEXP0HREADYMUX     | output | TARGEXP0, HREADYOUT                                    |
| TARGEXP0HAUSER        | output | TARGEXP0, HAUSER                                       |
| TARGEXP0HWUSER[3:0]   | output | TARGEXP0, HWUSER                                       |
| INITEXP0HRDATA[31:0]  | output | INITEXP0, HRDATA                                       |

| Port                    | I/O    | Description  |
|-------------------------|--------|--|
| INITEXP0HREADY          | output | INITEXP0, HREADY   |
| INITEXP0HRESP           | output | INITEXP0, HRESP  |
| INITEXP0EXRESP          | output | INITEXP0, EXRESP   |
| INITEXP0HRUSER[2:0]     | output | INITEXP0, HRUSER   |
| APBTARGEXP2PSTRB[3:0]   | output | APBTARGEXP2, PSTRB                                       |
| APBTARGEXP2PPROT[2:0]   | output | APBTARGEXP2, PPROT                                       |
| APBTARGEXP2PSEL         | output | APBTARGEXP2, PSELx                                       |
| APBTARGEXP2PENABLE      | output | APBTARGEXP2, PENABLE                                     |
| APBTARGEXP2PADDR[11:0]  | output | APBTARGEXP2, PADDR                                       |
| APBTARGEXP2PWRITE       | output | APBTARGEXP2, PWRITE                                      |
| APBTARGEXP2PWDATA[31:0] | output | APBTARGEXP2, PWDATA                                      |
| DAPTDO                  | output | Debug TDO  |
| DAPJTAGNSW              | output | JTAG or Serial-Wire selection JTAG mode(1) or SW mode(0) |
| DAPNTDOEN               | output | TDO output pad control signal                            |
| TPIUTRACEDATA[3:0]      | output | Output data  |
| TPIUTRACECLK            | output | Output clock, used by the TPA to sample the other pins   |

### Primitive Instantiaton

#### Verilog Instantiation:

MCU u\_sse050\_top\_syn (

```
.FCLK(fclk),
.PORESETN(poresetn),
.SYSRESETN(sysresetn),
.RTCSRCLK(rtsrcclk),
.IOEXPINPUTI(ioexpinputi[15:0]),
.IOEXPOUTPUTO(ioexpoutputo[15:0]),
.IOEXPOUTPUTENO(ioexpoutputeno[15:0]),
.UART0RXDI(uart0rxdi),
.UART0TXDO(uart0txdo),
.UART1RXDI(uart1rxdi),
.UART1TXDO(uart1txdo),
.SRAM0RDATA(sram0rdata[31:0]),
.SRAM0ADDR(sram0addr[12:0]),
.SRAM0WREN(sram0wren[3:0]),
.SRAM0WDATA(sram0wdata[31:0]),
.SRAM0CS(sram0cs),
.MTXHRESETN(mtxhreset),
```

.TARGFLASH0HSEL(targflash0hsel),  
.TARGFLASH0HADDR(targflash0haddr[28:0]),  
.TARGFLASH0HTRANS(targflash0htrans[1:0]),  
.TARGFLASH0HSIZE(targflash0hsize[2:0]),  
.TARGFLASH0HBURST(targflash0hburst[2:0]),  
.TARGFLASH0HREADYMUX(targflash0hreadymux),  
.TARGFLASH0HRDATA(targflash0hrdata[31:0]),  
.TARGFLASH0HRUSER(targflash0hruser[2:0]),  
.TARGFLASH0HRESP(targflash0hresp),  
.TARGFLASH0EXRESP(targflash0exresp),  
.TARGFLASH0HREADYOUT(targflash0hreadyout),  
.TARGEXP0HSEL(targexp0hsel),  
.TARGEXP0HADDR(targexp0haddr[31:0]),  
.TARGEXP0HTRANS(targexp0htrans[1:0]),  
.TARGEXP0HWRITE(targexp0hwrite),  
.TARGEXP0HSIZE(targexp0hsize[2:0]),  
.TARGEXP0HBURST(targexp0hburst[2:0]),  
.TARGEXP0HPROT(targexp0hprot[3:0]),  
.TARGEXP0MEMATTR(targexp0memattr[1:0]),  
.TARGEXP0EXREQ(targexp0exreq),  
.TARGEXP0HMASTER(targexp0hmaster[3:0]),  
.TARGEXP0HWDATA(targexp0hwdata[31:0]),  
.TARGEXP0HMASTLOCK(targexp0hmastlock),  
.TARGEXP0HREADYMUX(targexp0hreadymux),  
.TARGEXP0HAUSER(targexp0hauser),  
.TARGEXP0HWUSER(targexp0hwuser[3:0]),  
.TARGEXP0HRDATA(targexp0hrdata[31:0]),  
.TARGEXP0HREADYOUT(targexp0hreadyout),  
.TARGEXP0HRESP(targexp0hresp),  
.TARGEXP0EXRESP(targexp0exresp),  
.TARGEXP0HRUSER(targexp0hruser[2:0]),  
.INITEXP0HSEL(initexp0hsel),  
.INITEXP0HADDR(initexp0haddr[31:0]),  
.INITEXP0HTRANS(initexp0htrans[1:0]),  
.INITEXP0HWRITE(initexp0hwrite),  
.INITEXP0HSIZE(initexp0hsize[2:0]),  
.INITEXP0HBURST(initexp0hburst[2:0]),  
.INITEXP0HPROT(initexp0hprot[3:0]),



```

.INITEXP0MEMATTR(initexp0memattr[1:0]),
.INITEXP0EXREQ(initexp0exreq),
.INITEXP0HMASTER(initexp0hmaster[3:0]),
.INITEXP0HWDATA(initexp0hwdata[31:0]),
.INITEXP0HMASTLOCK(initexp0hmastlock),
.INITEXP0HAUSER(initexp0hauser),
.INITEXP0HWUSER(initexp0hwuser[3:0]),
.INITEXP0HRDATA(initexp0hrdata[31:0]),
.INITEXP0HREADY(initexp0hready),
.INITEXP0HRESP(initexp0hresp),
.INITEXP0EXRESP(initexp0exresp),
.INITEXP0HRUSER(initexp0hruser[2:0]),
.APBTARGEXP2PSEL(apbtargexp2psel),
.APBTARGEXP2PENABLE(apbtargexp2penable),
.APBTARGEXP2PADDR(apbtargexp2paddr[11:0]),
.APBTARGEXP2PWRITE(apbtargexp2pwrite),
.APBTARGEXP2PWDATA(apbtargexp2pwdata[31:0]),
.APBTARGEXP2PRDATA(apbtargexp2prdata[31:0]),
.APBTARGEXP2PREADY(apbtargexp2pready),
.APBTARGEXP2PSLVERR(apbtargexp2pslverr),
.APBTARGEXP2PSTRB(apbtargexp2pstrb[3:0]),
.APBTARGEXP2PPROT(apbtargexp2pprot[2:0]),
.MTXREMAP(mtxremap[3:0]),
.DAPSWDITMS(dapswditms),
.DAPTDI(daptdi),
.DAPTDO(daptdo),
.DAPNTRST(dapntrst),
.DAPSWCLKTCK(dapswclk_tck),
.DAPNTDOEN(dapntdoen),
.DAPJTAGNSW(dapjtagnew),
.TPIUTRACEDATA(tpiutracedata[3:0]),
.TPIUTRACECLK(tpiutracedclk),
.FLASHERR(flasherr),
.GPINT(gpint),
.FLASHINT(flashint)
);

```

#### **Vhdl Instantiation:**

COMPONENT MCU

```
    PORT(  
    FCLK:IN std_logic;  
    PORESETN:IN std_logic;  
    SYSRESETN:IN std_logic;  
    RTCSRCLK:IN std_logic;  
    UART0RXDI:IN std_logic;  
    UART1RXDI:IN std_logic;  
    CLK:IN std_logic;  
    RESET:IN std_logic;  
    IOEXPINPUTI:IN std_logic_vector(15 downto 0);  
    SRAM0RDATA:IN std_logic_vector(31 downto 0);  
    TARGFLASH0HRDATA:IN std_logic_vector(31 downto 0);  
    TARGFLASH0HRUSER:IN std_logic_vector(2 downto 0);  
    TARGFLASH0HRESP:IN std_logic;  
    TARGFLASH0EXRESP:IN std_logic;  
    TARGFLASH0HREADYOUT:IN std_logic;  
    TARGEXP0HRDATA: IN std_logic_vector(31 downto 0);  
    TARGEXP0HREADYOUT:IN std_logic;  
    TARGEXP0HRESP:IN std_logic;  
    TARGEXP0EXRESP:IN std_logic;  
    TARGEXP0HRUSER: IN std_logic_vector(2 downto 0);  
    INITEXP0HSEL:IN std_logic;  
    INITEXP0HADDR: IN std_logic_vector(31 downto 0);  
    INITEXP0HTRANS: IN std_logic_vector(1 downto 0);  
    INITEXP0HWRITE: IN std_logic;  
    INITEXP0HSIZE: IN std_logic_vector(2 downto 0);  
    INITEXP0HBURST: IN std_logic_vector(2 downto 0);  
    INITEXP0HPROT: IN std_logic_vector(3 downto 0);  
    INITEXP0MEMATTR: IN std_logic_vector(1 downto 0);  
    INITEXP0EXREQ: IN std_logic;  
    INITEXP0HMASTER: IN std_logic_vector(3 downto 0);  
    INITEXP0HWDATA: IN std_logic_vector(31 downto 0);  
    INITEXP0HMASTLOCK: IN std_logic;  
    INITEXP0HAUSER: IN std_logic;  
    INITEXP0HWUSER: IN std_logic_vector(3 downto 0);  
    APBTARGEXP2PRDATA: IN std_logic_vector(3 downto 0);  
    APBTARGEXP2PREADY: IN std_logic;  
    APBTARGEXP2PSLVERR: IN std_logic;
```

MTXREMAP: IN std\_logic\_vector(3 downto 0);  
DAPSWDITMS: IN std\_logic;  
DAPTDI: IN std\_logic;  
DAPNTRST: IN std\_logic;  
DAPSWCLKTCK: IN std\_logic;  
FLASHERR: IN std\_logic;  
FLASHINT: IN std\_logic;  
GPINT: IN std\_logic;  
IOEXPOUTPUTO:OUT std\_logic\_vector(15 downto 0);  
IOEXPOUTPUTENO:OUT std\_logic\_vector(15 downto 0);  
IOEXPINPUTI:OUT std\_logic\_vector(15 downto 0);  
UART0TXDO: OUT std\_logic;  
UART1TXDO: OUT std\_logic;  
UART0BAUDTICK: OUT std\_logic;  
UART1BAUDTICK: OUT std\_logic;  
INTMONITOR: OUT std\_logic;  
MTXHRESETN: OUT std\_logic;  
SRAM0ADDR:OUT std\_logic\_vector(12 downto 0);  
SRAM0WREN:OUT std\_logic\_vector(3 downto 0);  
SRAM0WDATA:OUT std\_logic\_vector(31 downto 0);  
SRAM0CS: OUT std\_logic;  
TARGFLASH0HSEL: OUT std\_logic;  
TARGFLASH0HREADYMUX: OUT std\_logic;  
SRAM0RDATA:OUT std\_logic\_vector(31 downto 0);  
TARGFLASH0HADDR:OUT std\_logic\_vector(28 downto 0);  
TARGFLASH0HTRANS:OUT std\_logic\_vector(1 downto 0);  
TARGFLASH0HSIZE:OUT std\_logic\_vector(2 downto 0);  
TARGFLASH0HBURST:OUT std\_logic\_vector(2 downto 0);  
TARGFLASH0HRDATA:OUT std\_logic\_vector(31 downto 0);  
TARGEXP0HADDR:OUT std\_logic\_vector(31 downto 0);  
TARGEXP0HSEL: OUT std\_logic;  
TARGEXP0HWRITE: OUT std\_logic;  
TARGEXP0EXREQ: OUT std\_logic;  
TARGEXP0HMASTLOCK: OUT std\_logic;  
TARGEXP0HREADYMUX: OUT std\_logic;  
TARGEXP0HAUSER: OUT std\_logic;  
INITEXP0HREADY: OUT std\_logic;  
INITEXP0HRESP: OUT std\_logic;

```

INITEXP0EXRESP: OUT std_logic;
TARGEXP0HTRANS:OUT std_logic_vector(1 downto 0);
TARGEXP0HSIZE:OUT std_logic_vector(2 downto 0);
TARGEXP0HBURST:OUT std_logic_vector(2 downto 0);
TARGEXP0HPROT:OUT std_logic_vector(3 downto 0);
TARGEXP0MEMATTR:OUT std_logic_vector(1 downto 0);
TARGEXP0HMASTER:OUT std_logic_vector(3 downto 0);
TARGEXP0HWDATA:OUT std_logic_vector(31 downto 0);
TARGEXP0HWUSER:OUT std_logic_vector(3 downto 0);
INITEXP0HRDATA:OUT std_logic_vector(31 downto 0);
INITEXP0HRUSER:OUT std_logic_vector(2 downto 0);
APBTARGEXP2PSTRB:OUT std_logic_vector(3 downto 0);
APBTARGEXP2PPROT:OUT std_logic_vector(2 downto 0);
APBTARGEXP2PADDR:OUT std_logic_vector(11 downto 0);
APBTARGEXP2PWDATA:OUT std_logic_vector(31 downto 0);
TPIUTRACEDATA:OUT std_logic_vector(3 downto 0);
APBTARGEXP2PSEL: OUT std_logic;
APBTARGEXP2PENABLE: OUT std_logic;
APBTARGEXP2PWRITE: OUT std_logic;
DAPTD0: OUT std_logic;
DAPJTAGNSW: OUT std_logic;
DAPNTDOEN: OUT std_logic;
TPIUTRACECLK: OUT std_logic;
);

END COMPONENT;

```

```

uut: MCU
  PORT MAP (
    FCLK=> fclk;
    PORESETN=> poresetn;
    SYSRESETN=> sysresetn;
    RTCSRCLK=> rtsrcclk;
    UART0RXDI=> uart0rxdi;
    UART1RXDI=> uart1rxdi;
    CLK=>clk,
    RESET=>reset,
    IOEXPINPUTI=>ioexpinputi,
    SRAM0RDATA=>sram0rdata,

```

TARGFLASH0HRDATA=>targflash0hrdata,  
TARGFLASH0HRUSER=>targflash0hruser,  
TARGFLASH0HRESP=>targflash0hresp,  
TARGFLASH0EXRESP=>targflash0exresp,  
TARGFLASH0HREADYOUT=>targflash0hreadyout,  
TARGEXP0HRDATA=>targexp0hrdata,  
TARGEXP0HREADYOUT=>targexp0hreadyout,  
TARGEXP0HRESP=>targexp0hresp,  
TARGEXP0EXRESP=>targexp0exresp,  
TARGEXP0HRUSER=>targexp0hruser,  
INITEXP0HSEL=>initexp0hsel,  
INITEXP0HADDR=>initexp0haddr,  
INITEXP0HTRANS=>initexp0htrans,  
INITEXP0HWRITE=>initexp0hwrite,  
INITEXP0HSIZE=>initexp0hsize,  
INITEXP0HBURST=>initexp0hburst,  
INITEXP0HPROT=>initexp0hprot,  
INITEXP0MEMATTR=>initexp0memattr,  
INITEXP0EXREQ=>initexp0exreq,  
INITEXP0HMASTER=>initexp0hmaster,  
INITEXP0HWDATA=>initexp0hwdata,  
INITEXP0HMASTLOCK=>initexp0hmastlock,  
INITEXP0HAUSER=>initexp0hauser,  
INITEXP0HWUSER=>initexp0hwuser,  
APBTARGEXP2PRDATA=>apbtargexp2prdata,  
APBTARGEXP2PREADY=>apbtargexp2pready,  
APBTARGEXP2PSLVERR=>apbtargexp2pslverr,  
MTXREMAP=>mtxremap,  
DAPSWDITMS=>dapswditms,  
DAPTDI=>daptdi,  
DAPNTRST=>dapntrst,  
DAPSWCLKTCK=>dapswclktck,  
FLASHERR=>flasherr,  
FLASHINT=>flashint,  
GPINT=>gpint,  
IOEXPOUTPUTO=>ioexpoutputo,  
IOEXPOUTPUTENO=>ioexpoutputeno,  
IOEXPINPUTI=>ioexpinputi,

UART0TXDO=>uart0txdo,  
UART1TXDO=>uart1txdo,  
UART0BAUDTICK=>uart0baudtick,  
UART1BAUDTICK=>uart1baudtick,  
INTMONITOR=>intmonitor,  
MTXHRESETN=>mtxhresetn,  
SRAM0ADDR=>sram0addr,  
SRAM0WREN=>sram0wren,  
SRAM0WDATA=>sram0wdata,  
SRAM0CS=>sram0cs,  
TARGFLASH0HSEL=>targflash0hsel,  
TARGFLASH0HREADYMUX=>targflash0hreadymux,  
SRAM0RDATA=>sram0rdata,  
TARGFLASH0HADDR=>targflash0haddr,  
TARGFLASH0HTRANS=>targflash0htrans,  
TARGFLASH0HSIZE=>targflash0hsize,  
TARGFLASH0HBURST=>targflash0hburst,  
TARGFLASH0HRDATA=>targflash0hrdata,  
TARGEXP0HADDR=>targexp0haddr,  
TARGEXP0HSEL=>targexp0hsel,  
TARGEXP0HWRITE=>targexp0hwrite,  
TARGEXP0EXREQ=>targexp0exreq,  
TARGEXP0HMASTLOCK=>targexp0hmastlock,  
TARGEXP0HREADYMUX=>targexp0hreadymux,  
TARGEXP0HAUSER=>targexp0hauser,  
INITEXP0HREADY=>initexp0hready,  
INITEXP0HRESP=>initexp0hresp,  
INITEXP0EXRESP=>initexp0exresp,  
TARGEXP0HTRANS=>targexp0htrans,  
TARGEXP0HSIZE=>targexp0hsize,  
TARGEXP0HBURST=>targexp0hburst,  
TARGEXP0HPROT=>targexp0hprot,  
TARGEXP0MEMATTR=>targexp0memattr,  
TARGEXP0HMASTER=>targexp0hmaster,  
TARGEXP0HWDATA=>targexp0hwdata,  
TARGEXP0HWUSER=>targexp0hwuser,  
INITEXP0HRDATA=>initexp0hrdata,  
INITEXP0HRUSER=>initexp0hruser,

```

APBTARGEXP2PSTRB=>apbtargexp2pstrb,
APBTARGEXP2PPROT=>apbtargexp2pprot,
APBTARGEXP2PADDR=>apbtargexp2paddr,
APBTARGEXP2PWDATA=>apbtargexp2pwwdata,
TPIUTRACEDATA=>tpiutracedata,
APBTARGEXP2PSEL=>apbtargexp2psel,
APBTARGEXP2PENABLE=>apbtargexp2penable,
APBTARGEXP2PWRITE=>apbtargexp2pwrite,
DAPTDO=>daptdo,
DAPJTAGNSW=>dapjtagnew,
DAPNTDOEN=>dapntdoen,
TPIUTRACECLK=>tpiutracedclk );

```

## 7.3 USB20\_PHY

### Primitive

The USB20\_PHY is a complete mixed signal IP solution that can implement OTG connection from Soc to other special manufacture technology. USB20\_PHY supports USB 2.0 480-Mbps protocol and data rate and it is backward compatible with USB 1.1 1.5-Mbps and 12-Mbps protocols and data rate.

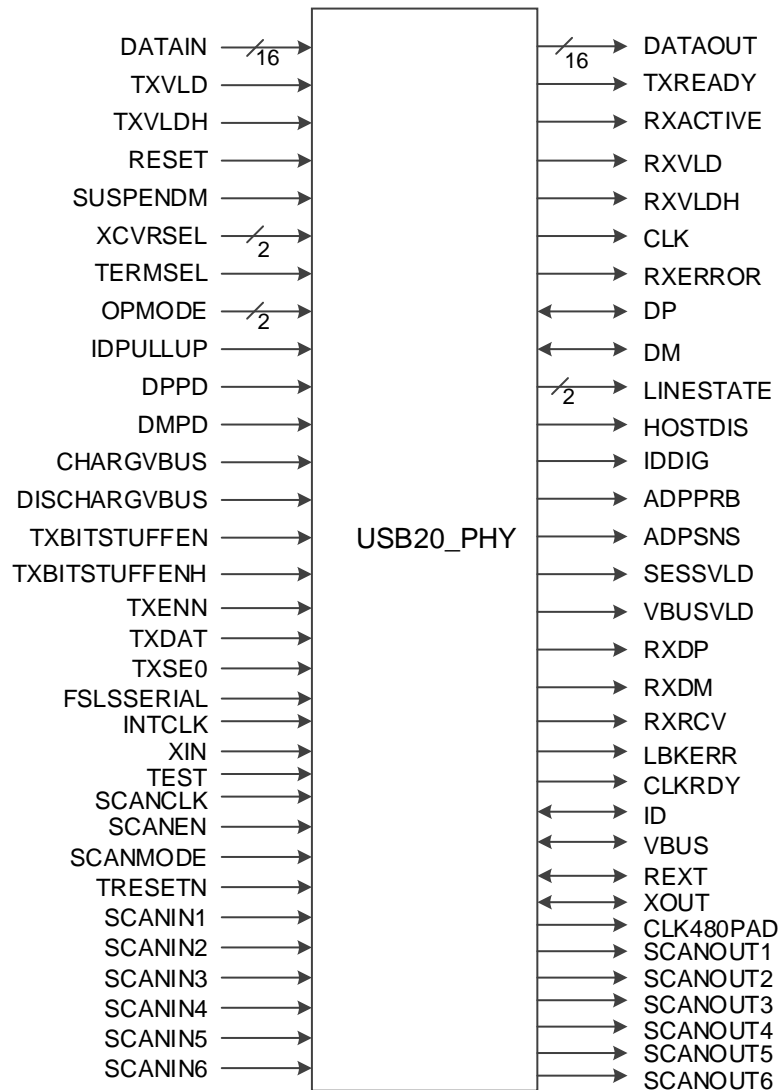
### Devices Supported

**Table 7-5 USB20\_PHY Devices Supported**

| Family     | Series | Device              |
|------------|--------|---------------------|
| LittleBee® | GW1NS  | GW1NS-2, GW1NS-2C   |
|            | GW1NSE | GW1NSE-2C           |
|            | GW1NSR | GW1NSR-2, GW1NSR-2C |

## Port Diagram

Figure 7-3 USB20\_PHY Port Diagram



## Port Description

Table 7-6 Port Description

| Port         | I/O   | Description  |
|--------------|-------|--|
| DATAIN[15:0] | input | 16-bit parallel USB data input bus   |
| TXVLD        | input | Transmit Valid. Indicates that the DataIn bus is valid.  |
| TXVLDH       | input | Transmit Valid High. When DataBus16_8 = 1, this signal indicates that the DataIn[15:8] bus contains valid transmit data. |
| RESET        | input | Reset. Reset all state machines in the UTM.  |
| SUSPENDM     | input | Suspend. 0:suspend, 1: normal  |
| XCVRSEL[1:0] | input | Transceiver Select. This signal selects between the LS, FS and HS transceivers   |
| TERMSEL      | input | Termination Select. This signal selects between the FS and HS terminations   |



| Port          | I/O    | Description   |
|---------------|--------|---|
| OPMODE[1:0]   | input  | Operational Mode. These signals select between various operational modes  |
| IDPULLUP      | input  | Signal that enables the sampling of the analog Id line.   |
| DPPD          | input  | This signal enables the 15k Ohm pull-down resistor on the DP line.  |
| DMPD          | input  | 0b : Pull-down resistor not connected to DM; 1b : Pull-down resistor connected to DM                                      |
| CHARGVBUS     | input  | This signal enables charging Vbus   |
| DISCHARGVBUS  | input  | The signal enables discharging Vbus.  |
| TXBITSTUFFEN  | input  | Indicates if the data on the DataOut[7:0] lines needs to be bitstuffed or not.  |
| TXBITSTUFFENH | input  | Indicates if the data on the DataOut[15:8] lines needs to be bitstuffed or not.   |
| TXENN         | input  | Active low enable signal. Only used when FsLsSerialMode is set to 1b  |
| TXDAT         | input  | Differential data at D+/D- output. Only used when FsLsSerialMode is set to 1b   |
| TXSE0         | input  | Force Single-Ended Zero. Only used when FsLsSerialMode is set to 1b   |
| FSLSSERIAL    | input  | 0b : FS and LS packets are sent using the parallel interface. 1b : FS and LS packets are sent using the serial interface. |
| INTCLK        | input  | Clock signals provided internally of the SoC  |
| TEST          | input  | For IP TESTING purpose. Please leave it unconnected since there are already soft pull-down in the IP                      |
| SCANCLK       | input  | Clock signals for scan mode   |
| SCANEN        | input  | Select to shift mode  |
| SCANMODE      | input  | High effective signal to enter scan mode  |
| TRESETN       | input  | Low effective RESET signal for scan mode  |
| SCANIN1       | input  | Scan chain input  |
| SCANIN2       | input  | Scan chain input  |
| SCANIN3       | input  | Scan chain input  |
| SCANIN4       | input  | Scan chain input  |
| SCANIN5       | input  | Scan chain input  |
| SCANIN6       | input  | Scan chain input  |
| DP            | inout  | USB data pin Data+  |
| DM            | inout  | USB data pin Data-  |
| ID            | inout  | ID signal from the cable  |
| VBUS          | inout  | Vbus signals connected with the cable   |
| REXT          | inout  | 12.7K High precision resistor   |
| XIN           | inout  | Crystal in signals, supported range is 12MHZ~24MHZ  |
| XOUT          | inout  | Crystal out signals   |
| DATAOUT[15:0] | output | DataOut. 16-bit parallel USB data output bus.   |
| TXREADY       | output | Transmit Data Ready.  |
| RXACTIVE      | output | Receive Active. Indicates that the receive state machine has detected SYNC and is active.                                 |

| Port           | I/O    | Description  |
|----------------|--------|--|
| RXVLD          | output | Receive Data Valid. Indicates that the DataOut bus has valid data.                                     |
| RXVLDH         | output | Receive Data Valid High.   |
| CLK            | output | Clock. This output is used for clocking receive and transmit parallel data.                            |
| RXERROR        | output | Receive Error.   |
| LINESTATE[1:0] | output | Line State. These signals reflect the current state of the single ended receivers.                     |
| HOSTDIS        | output | This signal is used for all types of peripherals connected to it.                                      |
| IDDIG          | output | Indicates whether the connected plug is a mini-A or mini-B.  |
| ADPPRB         | output | Indicates if the voltage on Vbus ( $0.6V < V_{th} < 0.75V$ ).  |
| ADPSNS         | output | Indicates if the voltage on Vbus ( $0.2V < V_{th} < 0.55V$ ).  |
| SESSVLD        | output | Indicates if the session for an A/B-peripheral is valid ( $0.8V < V_{th} < 2V$ ).                      |
| VBUSVLD        | output | Indicates if the voltage on Vbus is at a valid level for operation ( $4.4V < V_{th} < 4.75V$ ).        |
| RXDP           | output | Single-ended receive data, positive terminal. This signal is only valid if FsLsSerialMode is set to 1b |
| RXDM           | output | Single-ended receive data, negative terminal. This signal is only valid if FsLsSerialMode is set to 1b |
| RXRCV          | output | Receive data. This signal is only valid if FsLsSerialMode is set to 1b                                 |
| LBKERR         | output | used for observation   |
| CLKRDY         | output | Observation/debug signal to show that the internal PLL has locked and is ready.                        |
| CLK480PAD      | output | 480MHZ clock output for observation  |
| SCANOUT1       | output | Scan chain output  |
| SCANOUT2       | output | Scan chain output  |
| SCANOUT3       | output | Scan chain output  |
| SCANOUT4       | output | Scan chain output  |
| SCANOUT5       | output | Scan chain output  |
| SCANOUT6       | output | Scan chain output  |

## Parameters

Table 7-7 Parameters

| Name        | Default | Description   |
|-------------|---------|---|
| DATABUS16_8 | 1'b0    | Selects between 8 and 16 bit data transfers.  |
| ADP_PRBEN   | 1'b0    | Enables/disables the ADP Probe comparator   |
| TEST_MODE   | 5'b0    | used for testing and debugging purpose  |
| HSDRV1      | 1'b0    | High speed drive adjustment. Please connect to 0 for normal operation.                              |
| HSDRV0      | 1'b0    | High speed drive adjustment. Please connect to 0 for normal operation.                              |
| CLK_SEL     | 1'b0    | Clock source selection signal. 0 to select external clock provided by the crystal connected on XIN, |

| Name     | Default   | Description  |
|----------|-----------|--|
|          |           | XOUT. 1 to select internal clock provided on INTCLK port                             |
| M        | 4'b0      | M divider input data bit   |
| N        | 6'b101000 | N divider input data bit   |
| C        | 2'b01     | Control charge pump current input data bit, it supports from 30uA (00) to 60uA (11). |
| FOC_LOCK | 1'b0      | 0: LOCK is generated by PLL lock detector. 1: LOCK is always high(always lock)       |

### Primitive Instantiation

#### Verilog Instantiation:

```

        USB20_PHY usb20_phy_inst (
            .DATAOUT(dataout[15:0]),
            .TXREADY(txready),
            .RXACTIVE(rxactive),
            .RXVLD(rxvld),
            .RXVLDH(rxvldh),
            .CLK(clk),
            .RXERROR(rxerror),
            .DP(dp),
            .DM(dm),
            .LINESTATE(linestate[1:0]),
            .DATAIN(datain[15:0]),
            .TXVLD(txvld),
            .TXVLDH(txvldh),
            .RESET(reset),
            .SUSPENDM(suspendm),
            .XCVRSEL(xcvrsel[1:0]),
            .TERMSEL(termsel),
            .OPMODE(opmode[1:0]),
            .HOSTDIS(hostdis),
            .IDDIG(iddig),
            .ADPPRB(adpprb),
            .ADPSNS(adpsns),
            .SESSVLD(sessvld),
            .VBUSVLD(vbusvld),
            .RXDP(rxdp),
            .RXDM(rxdm),

```

.RXRCV(rxrcv),  
.IDPULLUP(idpullup),  
.DPPD(dppd),  
.DMPD(dmpd),  
.CHARGVBUS(chargvbus),  
.DISCHARGVBUS(dischargvbus),  
.TXBITSTUFFEN(txbitstuffen),  
.TXBITSTUFFENH(txbitstuffenh),  
.TXENN(txenn),  
.TXDAT(txdat),  
.TXSE0(txse0),  
.FSLSSERIAL(fslsserial),  
.LBKERR(lbkerr),  
.CLKRDY(clkrdy),  
.INTCLK(intclk),  
.ID(id),  
.VBUS(vbus),  
.REXT(rext),  
.XIN(xin),  
.XOUT(xout),  
.CLK480PAD(clk480pad),  
.TEST(test),  
.SCANOUT1(scanout1),  
.SCANOUT2(scanout2),  
.SCANOUT3(scanout3),  
.SCANOUT4(scanout4),  
.SCANOUT5(scanout5),  
.SCANOUT6(scanout6),  
.SCANCLK(scanclk),  
.SCANEN(scanen),  
.SCANMODE(scanmode),  
.TRESETN(tresetn),  
.SCANIN1(scanin1),  
.SCANIN2(scanin2),  
.SCANIN3(scanin3),  
.SCANIN4(scanin4),  
.SCANIN5(scanin5),  
.SCANIN6(scanin6)

```

);
defparam usb20_phy_inst.DATABUS16_8 = 1'b0;
defparam usb20_phy_inst.ADP_PRBEN = 1'b0;
defparam usb20_phy_inst.TEST_MODE = 5'b0;;
defparam usb20_phy_inst.HSDRV1 = 1'b0;
defparam usb20_phy_inst.HSDRV0 = 1'b0;
defparam usb20_phy_inst.CLK_SEL = 1'b0;
defparam usb20_phy_inst.M = 4'b0;
defparam usb20_phy_inst.N = 6'b101000;
defparam usb20_phy_inst.C = 2'b01;
defparam usb20_phy_inst.FOC_LOCK = 1'b0;

```

Vhdl Instantiation:

COMPONENT USB20\_PHY

GENERIC (

```

TEST_MODE:bit_vector:="00000";
        DATABUS16_8:bit:='0';
        ADP_PRBEN:bit:='0';
        HSDRV1:bit:='0';
        HSDRV0:bit:='0';
        CLK_SEL:bit:='0';
        M:bit_vector:="0000";
        N:bit_vector:=" 101000";
        C:bit_vector:="01";
        FOC_LOCK:bit:='0';

```

);

PORT(

```

        DATAIN:IN std_logic_vector(15 downto 0);
TXVLD:IN std_logic;
TXVLDH:IN std_logic;
RESET:IN std_logic;
SUSPENDM:IN std_logic;
XCVRSEL:IN std_logic_vector(1 downto 0);
TERMSEL:IN std_logic;
OPMODE:IN std_logic_vector(1 downto 0);
DATAOUT:OUT std_logic_vector(15 downto 0);
TXREADY:OUT std_logic;
RXACTIVE:OUT std_logic;

```

```
RXVLD:OUT std_logic;  
RXVLDH:OUT std_logic;  
CLK:OUT std_logic;  
RXERROR:OUT std_logic;  
DP:INOUT std_logic;  
DM:INOUT std_logic;  
LINESTATE:OUT std_logic_vector(1 downto 0);  
IDPULLUP:IN std_logic;  
DPPD:IN std_logic;  
DMPD:IN std_logic;  
CHARGVBUS:IN std_logic;  
DISCHARGVBUS:IN std_logic;  
TXBITSTUFFEN:IN std_logic;  
TXBITSTUFFENH:IN std_logic;  
TXENN:IN std_logic;  
TXDAT:IN std_logic;  
TXSE0:IN std_logic;  
FSLSSERIAL:IN std_logic;  
HOSTDIS:OUT std_logic;  
IDDIG:OUT std_logic;  
ADPPRB:OUT std_logic;  
ADPSNS:OUT std_logic;  
SESSVLD:OUT std_logic;  
VBUSVLD:OUT std_logic;  
RXDP:OUT std_logic;  
RXDM:OUT std_logic;  
RXRCV:OUT std_logic;  
LBKERR:OUT std_logic;  
CLKRDY:OUT std_logic;  
INTCLK:IN std_logic;  
ID:INOUT std_logic;  
VBUS:INOUT std_logic;  
REXT:INOUT std_logic;  
XIN:IN std_logic;  
XOUT:INOUT std_logic;  
TEST:IN std_logic;  
CLK480PAD:OUT std_logic;  
SCANCLK:IN std_logic;
```

```

SCANEN:IN std_logic;
SCANMODE:IN std_logic;
TRESETN:IN std_logic;
SCANIN1:IN std_logic;
SCANOUT1:OUT std_logic;
SCANIN2:IN std_logic;
SCANOUT2:OUT std_logic;
SCANIN3:IN std_logic;
SCANOUT3:OUT std_logic;
SCANIN4:IN std_logic;
SCANOUT4:OUT std_logic;
SCANIN5:IN std_logic;
SCANOUT5:OUT std_logic;
SCANIN6:IN std_logic;
SCANOUT6:OUT std_logic;
    );
END COMPONENT;
uut:  USB20_PHY
    PORT MAP (
DATAIN=>datain,
TXVLD=>txvld,
TXVLDH=>txvldh,
RESET=>reset,
SUSPENDM=>suspendm,
XCVRSEL=>xcvrssel,
TERMSEL=>termssel,
OPMODE=>opmode,
DATAOUT=>dataout,
TXREADY=>txready,
RXACTIVE=>rxactive,
RXVLD=>rxvld,
RXVLDH=>rxvldh,
CLK=>clk,
RXERROR=>rxerror,
DP=>dp,
DM=>dm,
LINESTATE=>linestate,
OIDPULLUP=>idpullup,

```

DPPD=>dppd,  
DMPD=>dmpd,  
CHARGVBUS=>chargvbus,  
DISCHARGVBUS=>dischargvbus,  
TXBITSTUFFEN=>txbitstufen,  
TXBITSTUFFENH=>txbitstuffenh,  
TXENN=>txenn,  
TXDAT=>txdat,  
TXSE0=>txse0,  
FSLSSERIAL=>fslsserial,  
HOSTDIS=>hostdis,  
IDDIG=>iddig,  
ADPPRB=>adpprb,  
ADPSNS=>adpsns,  
SESSVLD=>sessvld,  
VBUSVLD=>vbusvld,  
RXDP=>rxdp,  
RXDM=>rxdm,  
RXRCV=>rxrcv,  
LBKERR=>lbkerr,  
CLKRDY=>clkrdy,  
INTCLK=>intclk,  
ID=>id,  
VBUS=>vbus,  
REXT=>rext,  
XIN=>xin,  
XOUT=>xout,  
TEST=>test,  
CLK480PAD=>clk480pad,  
SCANCLK=>scanclk,  
SCANEN=>scanen,  
SCANMODE=>scanmode,  
TRESETN=>tresetn,  
SCANIN1=>scanin1,  
SCANOUT1=>scanout1,  
SCANIN2=>scanin2,  
SCANOUT2=>scanout2,  
SCANIN3=>scanin3,



```

SCANOUT3=>scanout3,
SCANIN4=>scanin4,
SCANOUT4=>scanout4,
SCANIN5=>scanin5,
SCANOUT5=>scanout5,
SCANIN6=>scanin6,
SCANOUT6=>scanout6
);

```

## 7.4 ADC

### Primitive

It is an 8-channel, 12-bit, single-ended Analog-to-digital Converter (ADC) with the features of low power, low leakage and high-dynamic.

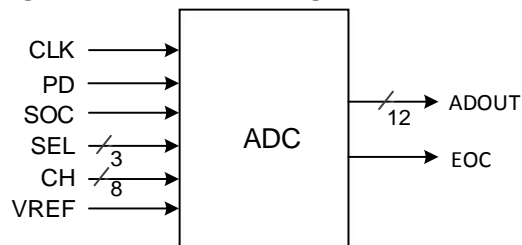
### Devices Supported

Table 7-8 ADC Devices Supported

| Family     | Series | Device              |
|------------|--------|---------------------|
| LittleBee® | GW1NS  | GW1NS-2, GW1NS-2C   |
|            | GW1NSE | GW1NSE-2C           |
|            | GW1NSR | GW1NSR-2, GW1NSR-2C |

### Port Diagram

Figure 7-4 ADC Port Diagram



### Port Description

Table 7-9 Port Description

| Port        | I/O    | Description                                |
|-------------|--------|--|
| ADOUT[11:0] | Output | ad conversion results.                     |
| EOC         | Output | end of conversion.                         |
| CLK         | Input  | main clock.                                |
| PD          | Input  | power down signal.                         |
| SOC         | Input  | start of conversion.                       |
| SEL[2:0]    | Input  | channel select signal.                     |
| CH[7:0]     | Input  | channel single-ended analog voltage input. |

| Port | I/O   | Description       |
|------|-------|-------------------|
| VREF | Input | voltage reference |

### Primitive Instantiation

#### Verilog Instantiation:

```

ADC adc_inst(
    .CLK(clk),
    .PD(pd),
    .SOC(soc),
    .SEL(sel[2:0]),
    .CH(ch[7:0]),
    .VREF(vref),
    .EOC(eoc),
    .ADOUT(adout[11:0])
);

```

#### Vhdl Instantiation:

```

COMPONENT ADC
PORT(
    CLK=>IN std_logic;
    PD=>IN std_logic;
    SOC=>IN std_logic;
    SEL=>IN std_logic_vector(2 downto 0);
    CH=>IN std_logic_vector(7 downto 0);
    VREF=>IN std_logic;
    EOC=>OUT std_logic;
    ADOUT=>OUT std_logic_vector(11 downto 0)
);
END COMPONENT;

uut=> ADC
PORT MAP (
    CLK=>clk,
    PD=>pd,
    SOC=>soc,
    SEL=>sel,
    CH=>ch,
    VREF=>vref,
    EOC=>eoc,
    ADOUT=>adout

```

```
);
```

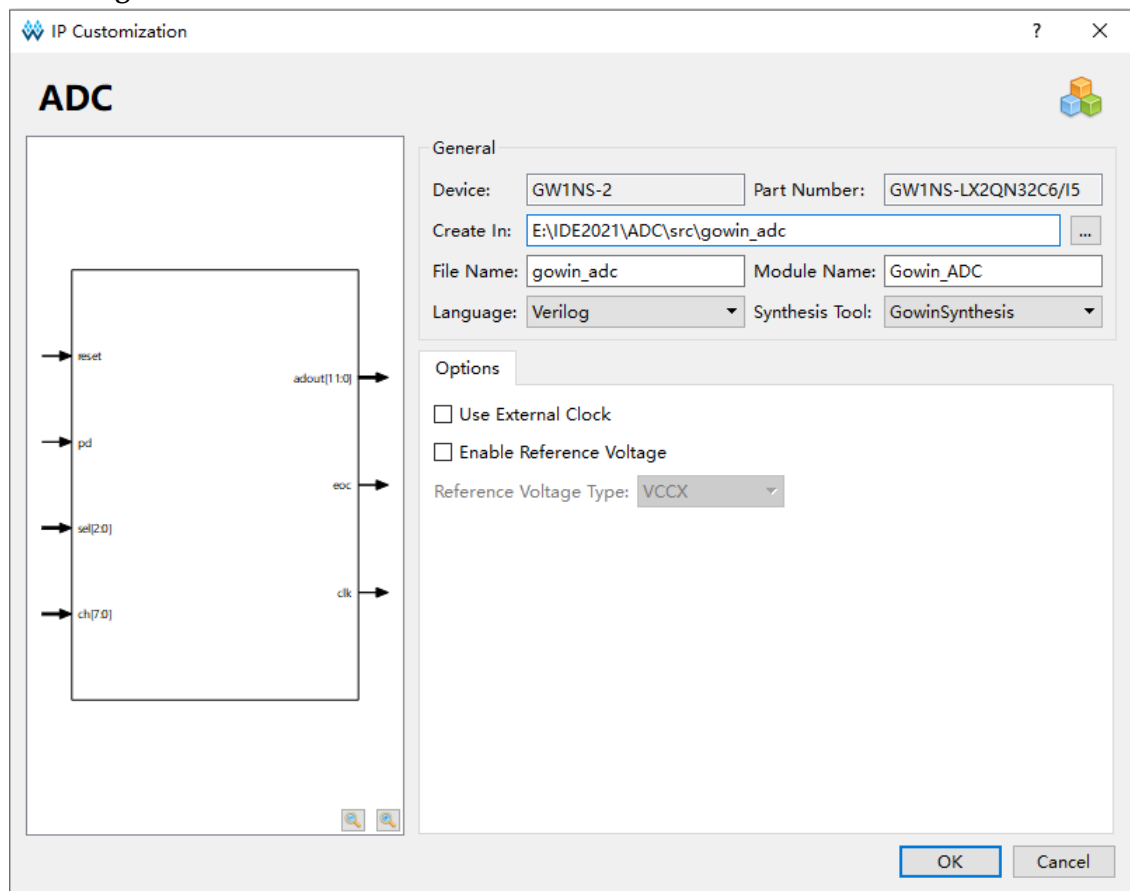
### Invoke IP

Click "ADC" on the IP Core Generator, and a brief introduction to the ADC will be displayed.

### IP Configuration

Double-click on the "ADC" to open the "IP Customization" window. This displays the "File", "Options", and ports diagram, as shown in Figure 7-5.

Figure 7-5 IP Customization of ADC



#### 1. File

- Device: Selected device;
- Part Number: Selected Part Number;
- Create In: The target path. You can reedit in the textbox or select a path by clicking the button.
- File Name: The file name. You can reedit in the textbox.
- Module Name: The module name. You can reedit in the textbox. The module name can not be the same as the primitive name. If it is the same, an error prompt will pop up.
- Language: Verilog and VHDL;

- Synthesis Tool: GowinSynthesis®.

## 2. Options

- Use External Clock: Configure external clock.
- Enable Reference Voltage: Configure enable reference voltage, and the default is VCCX.
- Reference Voltage Type: VCCX, 34/40(\*VCCX), 31/40(\*VCCX), 29/40(\*VCCX), 27/40(\*VCCX), 22/40(\*VCCX), 20/40(\*VCCX) and External.

## 3. Ports Diagram

The ports diagram is based on the current IP Core configuration, as shown in Figure 7-5;

### Generated Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin\_adc.v" file is a complete Verilog module to generate instance SPMI, and it is generated according to the IP configuration;
- "gowin\_adc\_tmp.v" is the instance template file;
- "gowin\_adc.ipc" file is IP configuration file. The user can load the file to configure the IP.

### Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

# 8 Miscellaneous

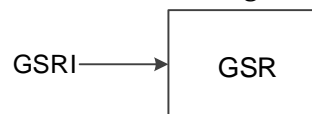
## 8.1 GSR

### Primitive

Global Reset/Set (GSR) is the global set/reset module, which can realize the global set/reset function, active-low. The default is connected to high level, if you want to control dynamically, you can connect an external signal to pull down to achieve set/reset of registers and other modules.

### Port Diagram

Figure 8-1 GSR Port Diagram



### Port Description

Table 8-1 Port Description

| Name | I/O   | Description           |
|------|-------|-----------------------|
| GSRI | Input | GSR input, active-low |

### Primitive Instantiation

#### Verilog instantiation:

```

GSR gsr_inst(
    .GSRI(GSRI)
);
  
```

#### Vhdl instantiation:

```

COMPONENT GSR
PORT (
    GSRI:IN std_logic
);
  
```

```

END COMPONENT;
gsr_inst:GSR
  PORT MAP(
    GSRI => GSRI
  );

```

## 8.2 INV

### Primitive

Inverter (INV)

### Port Diagram

Figure 8-2 INV Port Diagram



### Port Description

Table 8-2 Port Description

| Name | I/O    | Description     |
|------|--------|-----------------|
| I    | Input  | INV data input  |
| O    | Output | INV data output |

### Primitive Instantiation

#### Verilog instantiation:

```

INV uut (
  .O(O),
  .I(I)
);

```

#### Vhdl instantiation:

```

COMPONENT INV
  PORT (
    O:OUTPUT std_logic;
    I:IN std_logic

  );
END COMPONENT;
uut:INV
  PORT MAP(

```

```

        O => O,
        I => I
    );

```

## 8.3 VCC

### Primitive

VCC is the logic high level generator.

### Port Diagram

Figure 8-3 VCC Port Diagram



### Port Description

Table 8-3 Port Description

| Name | I/O    | Description |
|------|--------|-------------|
| V    | Output | VCC output  |

### Primitive Instantiation

#### Verilog instantiation:

```

VCC uut (
    .V(V)
);

```

#### Vhdl instantiation:

```

COMPONENT VCC
  PORT (
    V:OUT std_logic
  );
END COMPONENT;
uut:VCC
  PORT MAP(
    V => V
  );

```

## 8.4 GND

### Primitive

GND is the logic low level generator.

### Port Diagram

Figure 8-4 GND Port Diagram



### Port Description

Table 8-4 Port Description

| Name | I/O    | Description |
|------|--------|-------------|
| G    | Output | GND output  |

### Primitive Instantiation

#### Verilog instantiation:

```
GND uut (  
    .G(G)  
);
```

#### Vhdl instantiation:

```
COMPONENT GND  
    PORT (  
        G:OUT std_logic  
    );  
END COMPONENT;  
uut:GND  
    PORT MAP(  
        G => G
```

```
);
```



## 8.5 BANDGAP

### Primitive

BANDGAP is used to provide constant voltage and current for certain modules in the chip. If BANDGAP is turned off, the OSC, PLL, FLASH and other modules will no longer work, which can reduce the power consumption.

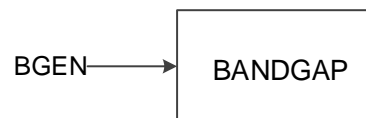
### Supported Devices

Table 8-5 BANDGAP Devices Supported

| Family     | Series | Device                               |
|------------|--------|--------------------------------------|
| LittleBee® | GW1NZ  | GW1NZ-1, GW1NZ-1C                    |
|            | GW1N   | GW1N-2, GW1N-2B, GW1N-1P5, GW1N-1P5B |
|            | GW1NR  | GW1NR-2, GW1NR-2B                    |

### Port Diagram

Figure 8-5 BANDGAP Port Diagram



### Port Description

Table 8-6 Port Description

| Name | I/O   | Description                         |
|------|-------|-------------------------------------|
| BGEN | Input | BANDGAP enable signal, active-high. |

### Primitive Instantiation

#### Verilog Instantiation:

```

BANDGAP uut (
    .BGEN(bgen)
);
  
```

#### Vhdl Instantiation:

```

COMPONENT BANDGAP
PORT (
    BGEN:IN std_logic
);
END COMPONENT;

uut:BANDGAP
PORT MAP(
    BGEN=> I
  
```

```
);
```

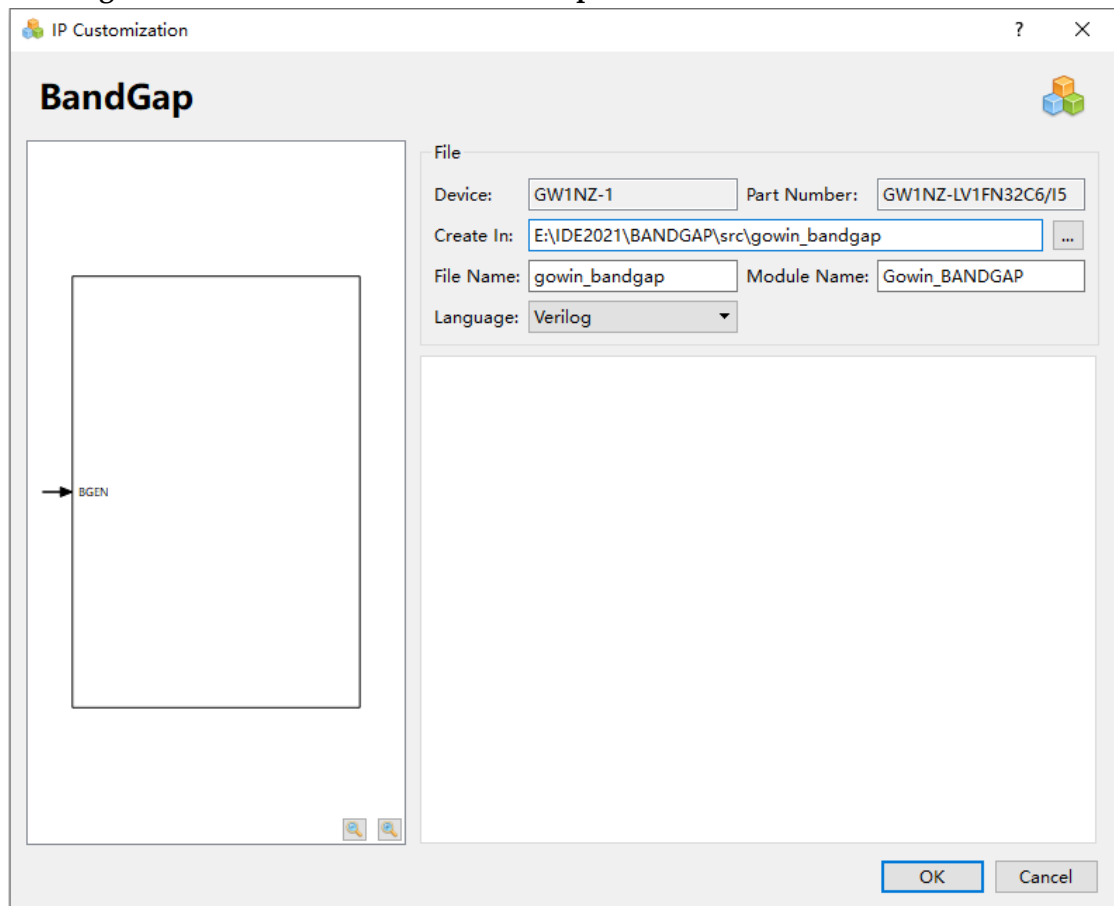
### Invoke IP

Click "BandGap" on the IP Core Generator, and a brief introduction to the BandGap will be displayed.

### IP Configuration

Double-click the "BandGap" to open the "IP Customization" window. This displays the "File", "Options", and ports diagram, as shown in Figure 8-6.

Figure 8-6 IP Customization of BandGap



#### 1. File

The File displays the basic information related to BandGap. The BandGap file configuration is similar to that of ADC. For the detailed configuration, please see 7.4 ADC > Invoke IP.

#### 2. Ports Diagram

The ports diagram is based on the current IP Core configuration, as shown in Figure 8-6;

### Generated Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin\_bandgap.v " file is a complete Verilog module to generate instance BandGap, and it is generated according to the IP configuration;
- "gowin\_bandgap\_tmp.v" is the instance template file;
- "gowin\_bandgap.ipc" file is IP configuration file. The user can load the file to configure the IP.

**Note!**

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

## 8.6 SPMI

### Primitive

System Power Management Interface (SPMI) is a two-wire serial interface, which can be used to dynamically control the internal power supply of the on-chip system.

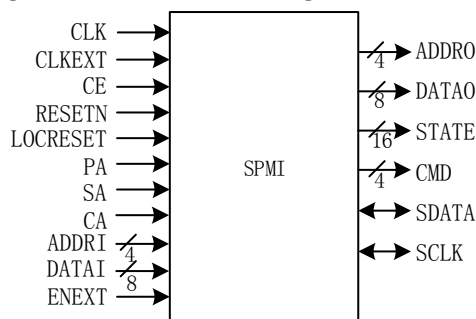
### Devices Supported

**Table 8-7 SPMI Devices Supported**

| Family     | Series | Device            |
|------------|--------|-------------------|
| LittleBee® | GW1NZ  | GW1NZ-1, GW1NZ-1C |

### Port Diagram

**Figure 8-7 SPMI Port Diagram**



### Port Description

**Table 8-8 Port Description**

| Port     | I/O   | Description                |
|----------|-------|----------------------------|
| CLK      | input | Clock input                |
| CLKEXT   | input | External clock input       |
| CE       | input | Clock Enable               |
| RESETN   | input | Reset input                |
| ENEXT    | input | Enext input                |
| LOCRESET | input | Local reset input          |
| PA       | input | Priority arbitration input |

| Port  | I/O    | Description                  |
|-------|--------|------------------------------|
| SA    | input  | Secondary arbitration input  |
| CA    | input  | Connection arbitration input |
| ADDRI | input  | Addr input                   |
| DATAI | input  | Data input                   |
| ADDRO | output | Addr output                  |
| DATAO | output | data output                  |
| STATE | output | state output                 |
| CMD   | output | command output               |
| SDATA | inout  | SPMI Serial data             |
| SCLK  | inout  | SPMI Serial Clock            |

### Primitive Instantiation

#### Verilog Instantiation:

```

SPMI uut (
    .ADDRO(addrro),
    .DATAO(datao),
    .STATE(state),
    .CMD(cmd),
    .SDATA(sdata),
    .SCLK(sclk),
    .CLK(clk),
    .CE(ce),
    .RESETN(resetn),
    .LOCRESET(locreset),
    .PA(pa),
    .SA(sa),
    .CA(ca),
    .ADDRI(addrri),
    .DATAI(datai),
    .CLKEXT(clkext),
    .ENEXT(enext)
);

```

#### Vhdl Instantiation:

```

COMPONENT SPMI
PORT(
    CLK:IN std_logic;
    CLKEXT:IN std_logic;

```

```

        CE:IN std_logic;
        RESETN:IN std_logic;
        ENEXT:IN std_logic;
        LOCRESET:IN std_logic;
        PA:IN std_logic;
        SA:IN std_logic;
        CA:IN std_logic;
        ADDR:IN std_logic_vector(3 downto 0);
        DATAI:IN std_logic_vector(7 downto 0);
        ADDRO:OUT std_logic_vector(3 downto 0);
        DATAO:OUT std_logic_vector(7 downto 0);
        STATE:OUT std_logic_vector(15 downto 0);
        CMD:OUT std_logic_vector(3 downto 0);
        SDATA:INOUT std_logic;
        SCLK:INOUT std_logic
    );
END COMPONENT;
uut: SPMI
PORT MAP (
    CLK=>clk,
        CLKEXT=>clkext,
        CE=>ce,
        RESETN=>resetn,
        ENEXT=>enext,
        LOCRESET=>locreset,
        PA=>pa,
        SA=>sa,
        CA=>ca,
        ADDR=>addr,
        DATAI=>datai,
        ADDRO=>addro,
        DATAO=>datao,
        STATE=>state,
        CMD=>cmd,
        SDATA=>sdata,
        SCLK=>sclk
    );

```

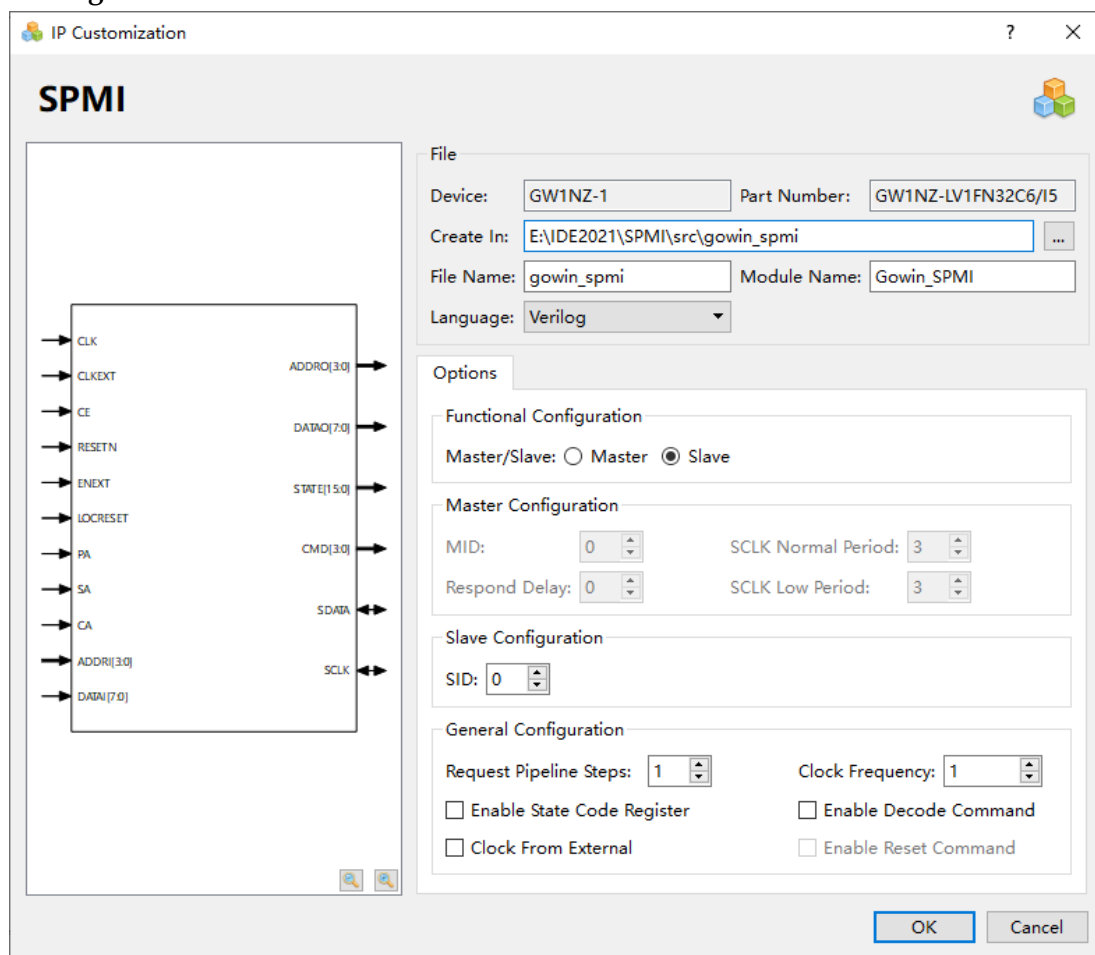
## Invoke IP

Click "SPMI" on the IP Core Generator, and a brief introduction to the SPMI will be displayed.

## IP Configuration

Double-click the "SPMI" to open the "IP Customization" window. This displays the "File", "Options", and ports diagram, as shown in Figure 8-8.

Figure 8-8 IP Customization of SPMI



### 1. File

The File displays the basic information related to SPMI. The SPMI file configuration box is similar to that of ADC. For the details, please see 7.4 ADC > Invoke IP.

### 2. Options

- The Options is used to configure SPMI by users, as shown in Figure 8-8.
- Functional Configuration:
  - Shutdown by VCCEN: Shutdown by external pin VCCEN. If this option is checked, the communication function of SPMI will be disabled.

- Master/Slave: Set SPMI as Master or Slave.
  - Master Configuration:
    - MID: Master ID. The range is 0-3, and default value is 0.
    - Respond Delay: Set the response delay.
    - SCLK Normal Period: Set SCLK period in normal mode.
    - SCLK Normal Period: Set SCLK period in low mode.
  - Slave Configuration:
    - SID: Slave ID.
  - General configuration:
    - Enable State Code Register: Enable or disable the state code register. If "Enable State Code Register" is checked, the output state code will pass a register.
    - Request Pipeline Steps: Set the sampling delay step of the request signal.
    - Enable Decode Command: Enable or disable decode. If "Enable Decode Command" is checked, SPMI will decode the reset, sleep, shutdown, and wakeup.
    - Enable Decode Command: Enable or disable reset.
    - Clock From External: Enable or disable the external clock.
    - Clock Frequency: System clock frequency.
3. Ports Diagram
- The ports diagram is based on the current IP Core configuration, as shown in Figure 8-8;

#### Generated Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin\_spmi.v " file is a complete Verilog module to generate instance SPMI, and it is generated according to the IP configuration;
- "gowin\_spmi\_tmp.v" is the instance template file;
- "gowin\_spmi.ipc " file is IP configuration file. The user can load the file to configure the IP.

#### Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

## 8.7 I3C

### Primitive

Improved Inter Integrated Circuit (I3C) is a two-wire bus with the key features of I<sup>2</sup>C and SPI, which can effectively reduce the physical ports of integrated circuit, support the advantages of low power, high data rate and other existing port protocols.

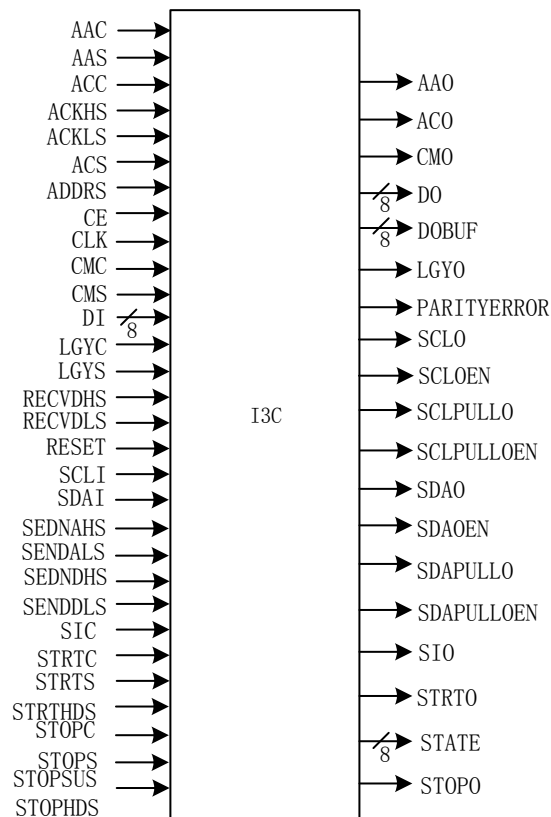
### Devices Supported

Table 8-9 I3C Devices Supported

| Family     | Series | Device            |
|------------|--------|-------------------|
| LittleBee® | GW1NZ  | GW1NZ-1, GW1NZ-1C |

### Port Diagram

Figure 8-9 I3C Port Diagram



### Port Description

Table 8-10 Port Description

| Port  | I/O   | Description  |
|-------|-------|--------------|
| CE    | input | Clock Enable |
| RESET | input | Reset input  |
| CLK   | input | Clock input  |



| Port    | I/O    | Description   |
|---------|--------|---|
| LGYS    | input  | The current communication object is the I2C setting signal                  |
| CMS     | input  | The device enters the Master's set signal                                   |
| ACS     | input  | Select the setting signal when determining whether to continue.             |
| AAS     | input  | Reply the ACK setting signal when a reply is required from the ACK/NACK     |
| STOPS   | input  | Input the STOP command  |
| STRTS   | input  | Input the START command.  |
| LGYC    | input  | The current communication object is the I2C                                 |
| CMC     | input  | The reset signal that the device is in master.                              |
| ACC     | input  | The reset signal that selects continue when selecting whether to continue   |
| AAC     | input  | Reply the ACK reset signal when a reply is required from the ACK/NACK       |
| SIC     | input  | Interrupt to identify the reset signal                                      |
| STOPC   | input  | The reset signal is in STOP state   |
| STRTC   | input  | The reset signal is in START state  |
| STRTHDS | input  | Adjust the setting signal when generating START                             |
| SENDAHS | input  | Adjust the setting signal of SCL at a high level when the address is sent.  |
| SENDALS | input  | Adjust the setting signal of SCL at a low level when the address is sent    |
| ACKHS   | input  | Adjust the setting signal of SCL at a high level in ACK.                    |
| SENDDL  | input  | Adjust the setting signal of SCL at a low level in ACK.                     |
| RECVHDS | input  | Adjust the setting signal of SCL at a high level when the data are received |
| RECVLDS | input  | Adjust the setting signal of SCL at a low level when the data are received  |
| ADDRS   | input  | The slave address setting interface   |
| DI      | input  | Data Input.   |
| SDAI    | input  | I3C serial data input   |
| SCLI    | input  | I3C serial clock input  |
| LGYO    | output | Output the current communication object as the I2C command.                 |
| CMO     | output | Output the command of the device is in the Master mode.                     |
| ACO     | output | Continue to output when selecting whether to continue                       |
| AAO     | output | Reply ACK when you need to reply ACK/NACK                                   |
| SIO     | output | Interrupt to output the identity bit  |

| Port        | I/O    | Description                                      |
|-------------|--------|--|
| STOPO       | output | Output the STOP command                          |
| STRTO       | output | Output the START command                         |
| PARITYERROR | output | Output check when receiving data                 |
| DOBUF       | output | Data output after caching                        |
| DO          | output | Data output directly                             |
| STATE       | output | Output the internal state                        |
| SDAO        | output | I3C serial data output                           |
| SCLO        | output | I3C serial clock output                          |
| SDAOEN      | output | I3C serial data oen output                       |
| SCLOEN      | output | I3C serial clock oen output                      |
| SDAPULLO    | output | Controllable pull-up of the I3C serial data      |
| SCLPULLO    | output | Controllable pull-up of the I3C serial clock     |
| SDAPULLOEN  | output | Controllable pull-up of the I3C serial data oen  |
| SCLPULLOEN  | output | Controllable pull-up of the I3C serial clock oen |

### Primitive Instantiation

Verilog Instantiation:

```
I3C i3c_inst (
    .LGYO(lgyo),
    .CMO(cmo),
    .ACO(aco),
    .AAO(aao),
    .SIO(sio),
    .STOPO(stopo),
    .STRTO(strto),
    .PARITYERROR(parityerror),
    .DOBUF(dobuf),
    .DO(dout),
    .STATE(state),
    .SDAO(sdao),
    .SCLO(sclo),
    .SDAOEN(sdaoen),
    .SCLOEN(scloen),
    .SDAPULLO(sdapullo),
    .SCLPULLO(sclpullo),
    .SDAPULLOEN(sdapulloen),
    .SCLPULLOEN(sclpulloen),
    .LGYS(lgys),
```

```

.CMS(cms),
.ACS(acs),
.AAS(aas),
.STOPS(stops),
.STRTS(strts),
.LGYC(lgyc),
.CMC(cmc),
.ACC(acc),
.AAC(aac),
.SIC(sic),
.STOPC(stopc),
.STRTC(strtc),
.STRTHDS(strthds),
.SENDAHS(sendahs),
.SENDALS(sendals),
.ACKHS(ackhs),
.ACKLS(ackls),
.STOPSUS(stopsus),
.STOPHDS(stophds),
.SENDDHS(senddhs),
.SENDDLs(senddls),
.RECVDHS(recvdhs),
.RECVDLS(recvdls),
.ADDRS(addr),
.DI(di),
.SDAI(sdai),
.SCLI(scli),
.CE(ce),
.RESET(reset),
.CLK(clk)
);

```

#### **Vhdl Instantiation:**

```
COMPONENT I3C
```

```
PORT (
```

```

    LGYO: OUT STD_LOGIC;
    CMO: OUT STD_LOGIC;
    ACO: OUT STD_LOGIC;
    AAO: OUT STD_LOGIC;

```

SIO: OUT STD\_LOGIC;  
STOPO: OUT STD\_LOGIC;  
STRTO: OUT STD\_LOGIC;  
PARITYERROR: OUT STD\_LOGIC;  
DOBUF: OUT STD\_LOGIC\_VECTOR(7 DOWNT0 0);  
DOUT: OUT STD\_LOGIC\_VECTOR(7 DOWNT0 0);  
STATE: OUT STD\_LOGIC\_VECTOR(7 DOWNT0 0);  
SDAO: OUT STD\_LOGIC;  
SCLO: OUT STD\_LOGIC;  
SDAOEN: OUT STD\_LOGIC;  
SCLOEN: OUT STD\_LOGIC;  
SDAPULLO: OUT STD\_LOGIC;  
SCLPULLO: OUT STD\_LOGIC;  
SDAPULLOEN: OUT STD\_LOGIC;  
SCLPULLOEN: OUT STD\_LOGIC;  
LGYS: IN STD\_LOGIC;  
CMS: IN STD\_LOGIC;  
ACS: IN STD\_LOGIC;  
AAS: IN STD\_LOGIC;  
STOPS: IN STD\_LOGIC;  
STRTS: IN STD\_LOGIC;  
LGYC: IN STD\_LOGIC;  
CMC: IN STD\_LOGIC;  
ACC: IN STD\_LOGIC;  
AAC: IN STD\_LOGIC;  
SIC: IN STD\_LOGIC;  
STOPC: IN STD\_LOGIC;  
STRTC: IN STD\_LOGIC;  
STRTHDS: IN STD\_LOGIC;  
SENDAHS: IN STD\_LOGIC;  
SENDALS: IN STD\_LOGIC;  
ACKHS: IN STD\_LOGIC;  
ACKLS: IN STD\_LOGIC;  
STOPSUS: IN STD\_LOGIC;  
STOPHDS: IN STD\_LOGIC;  
SENDDHS: IN STD\_LOGIC;  
SENDDL: IN STD\_LOGIC;  
RECVHDS: IN STD\_LOGIC;

```

    RECVCLS: IN STD_LOGIC;
    ADDR: IN STD_LOGIC;
    DI: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
    SDAI: IN STD_LOGIC;
    SCLI: IN STD_LOGIC;
    CE: IN STD_LOGIC;
    RESET: IN STD_LOGIC;
    CLK: IN STD_LOGIC
  );
END COMPONENT;

```

uut: I3C

```

PORT MAP (
  LGYO => lgyo,
  CMO => cmo,
  ACO => aco,
  AAO => aao,
  SIO => sio,
  STOPO => stopo,
  STRTO => strto,
  PARITYERROR => parityerror,
  DOBUF => dobuf,
  DOUT => dout,
  STATE => state,
  SDAO => sdao,
  SCLO => sclo,
  SDAOEN => sdaoen,
  SCLOEN => scloen,
  SDAPULLO => sdapullo,
  SCLPULLO => sclpullo,
  SDAPULLOEN => sdapulloen,
  SCLPULLOEN => sclpulloen,
  LGYS => lgys,
  CMS => cms,
  ACS => acs,
  AAS => aas,
  STOPS => stops,
  STRTS => strts,

```

```
LGYC => lgyc,  
CMC => cmc,  
ACC => acc,  
AAC => aac,  
SIC => sic,  
STOPC => stopc,  
STRTC => strtc,  
STRTHDS => strthds,  
SENDAHS => sendahs,  
SENDALS => sendals,  
ACKHS => ackhs,  
ACKLS => ackls,  
STOPSUS => stopsus,  
STOPHDS => stophds,  
SENDDHS => senddhs,  
SENDDLs => senddls,  
RECVHDS => recvdhs,  
RECVDLs => recvdls,  
ADDRS => addrs,  
DI => di,  
SDAI => sdai,  
SCLI => scli,  
CE => ce,  
RESET => reset,  
CLK => clk  
);
```

### Invoke IP

Click "I3C > I3C SDR" on the "IP Core Generator" page. A brief introduction to the I3C SDR will be displayed.

### Configure IP

Double-click "I3C SDR", and the "IP Customization" window pops up. This displays the "File", "Options", and port diagram, as shown in Figure 8-10.

Figure 8-10 IP Customization of I3C

**I3C SDR**

**File**

Device: GW1NZ-1 Part Number: GW1NZ-LV1FN32C6/15

Create In: E:\IDE2021\I3C\src\gowin\_i3c

File Name: gowin\_i3c Module Name: Gowin\_I3C

Language: Verilog

**Options**

SLAVE STATIC ADDRESS: 00 (7'h00~7'h7F)

OK Cancel

### 1. File

The File displays the basic information related to the I3C. The I3C file configuration is similar to that of ADC. For the detailed configuration instructions, please see 7.4 ADC > Invoke IP.

### 2. Options

- The Options is used to configure I3C by users, as shown in Figure 8-10.
- SLAVE STATIC ADDRESS: Specify the static address of the Slave.

### 3. Ports Diagram

The ports diagram is based on the IP Core configuration, as shown in Figure 8-10;

### Generated Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin\_i3c.v" file is a complete Verilog module to generate instance I3C, and it is generated according to the IP configuration;
- "gowin\_i3c\_tmp.v" is the instance template file;
- "gowin\_i3c.ipc" file is IP configuration file. You can load the file to configure the IP.

**Note!**

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

## 8.8 activeFlash

### Primitive

Activate embedded SPI Nor Flash module. When activeFlash instantiated, I\_active\_flash\_sclk must be a clock signal and I\_active\_flash\_holdn needs to be pulled up.

**Note!**

When activeFlash instantiated, the user design will add 14 LUTs and 10 REGs.

### Devices Supported

**Table 8-11 activeFlash Devices Supported**

| Family | Series | Device              |
|--------|--------|---------------------|
| Arora  | GW2AN  | GW2AN-18X, GW2AN-9X |

### Port Diagram

**Figure 8-11 activeFlash Port Diagram**



### Port Description

**Table 8-12 Port Description**

| Port                 | I/O    | Description  |
|----------------------|--------|--|
| I_active_flash_holdn | input  | Clock hold signal; high input activates Flash.         |
| I_active_flash_sclk  | input  | Clock input clock                                      |
| O_active_flash_ready | output | Ready signal; high indicates Flash has been activated. |

### Primitive Instantiation

#### Verilog Instantiation:

```

activeFlash activeFlash_inst (
    . I_active_flash_holdn I_active_flash_holdn),
    . I_active_flash_sclk (I_active_flash_sclk),
    . O_active_flash_ready (O_active_flash_ready)
);
  
```

#### Vhdl Instantiation:

```

COMPONENT activeFlash
  
```



```
    PORT(  
        I_active_flash_holdn:IN std_logic;  
        I_active_flash_sclk:IN std_logic;  
        O_active_flash_ready:OUT std_logic  
    );  
END COMPONENT;  
uut: activeFlash  
    PORT MAP (  
        I_active_flash_holdn => I_active_flash_holdn,  
        I_active_flash_sclk => I_active_flash_sclk,  
        O_active_flash_ready => O_active_flash_ready  
    );
```

### Conditions

When one of the following conditions is met, the activeFlash module needs to be instantiated in the user design.

1. When Configuration - Bitstream background programming in Gowin Software is set to I2C/JTAG/SSPI/QSSPI, activeFlash needs to be instantiated.
2. When Configuration - Bitstream background programming is set to OFF and the SPI Nor Flash Interface IP is used, activeFlash needs to be instantiated.

