Copyright Notice

Contact Information

Should you have any problems with the information contained herein or any suggestions, please contact Nuclei System Technology by email support@nucleisys.com, or visit “Nuclei User Center” website http://user.nucleisys.com for supports or online discussion.

Revision History

Rev.	Revision Date	Revised Section	Revised Content
1.0.0	2020/1/20	N/A	1. First version as the full English

Table of Contents

List of Tables

List of Figures

1. SoC Overview

To easy user to evaluate Nuclei Processor Core, the prototype SoC (called Hummingbird SoC) is provided for evaluation purpose. This prototype SoC includes:

- Processor Core, it can be Nuclei N class, NX class or UX class Processor Core.
- On-Chip SRAMs for instruction and data.
- The SoC buses.
- The basic peripherals, such as UART, GPIO, SPI, I2C, etc.

With this prototype SoC, user can run simulations, map it into the FPGA board, and run with real embedded application examples.

2. SoC Diagram

The SoC diagram is as depicted in Figure 2 -1. There are Processor Core, Buses (System Bus and Peripheral Bus) and Peripherals in the SoC.

Note:

- If there is no ILM and DLM interface configured for core, then ILM and DLM will be only connected to System Bus.
- If there is no FIO (Fast-IO) configured for core, then the GPIO will be connected to Peripheral Bus.
- If there is no PPI configured for core, then the Peripheral Bus will be connected to System Bus.

For more information of the peripherals, please refer to Chapter 9..

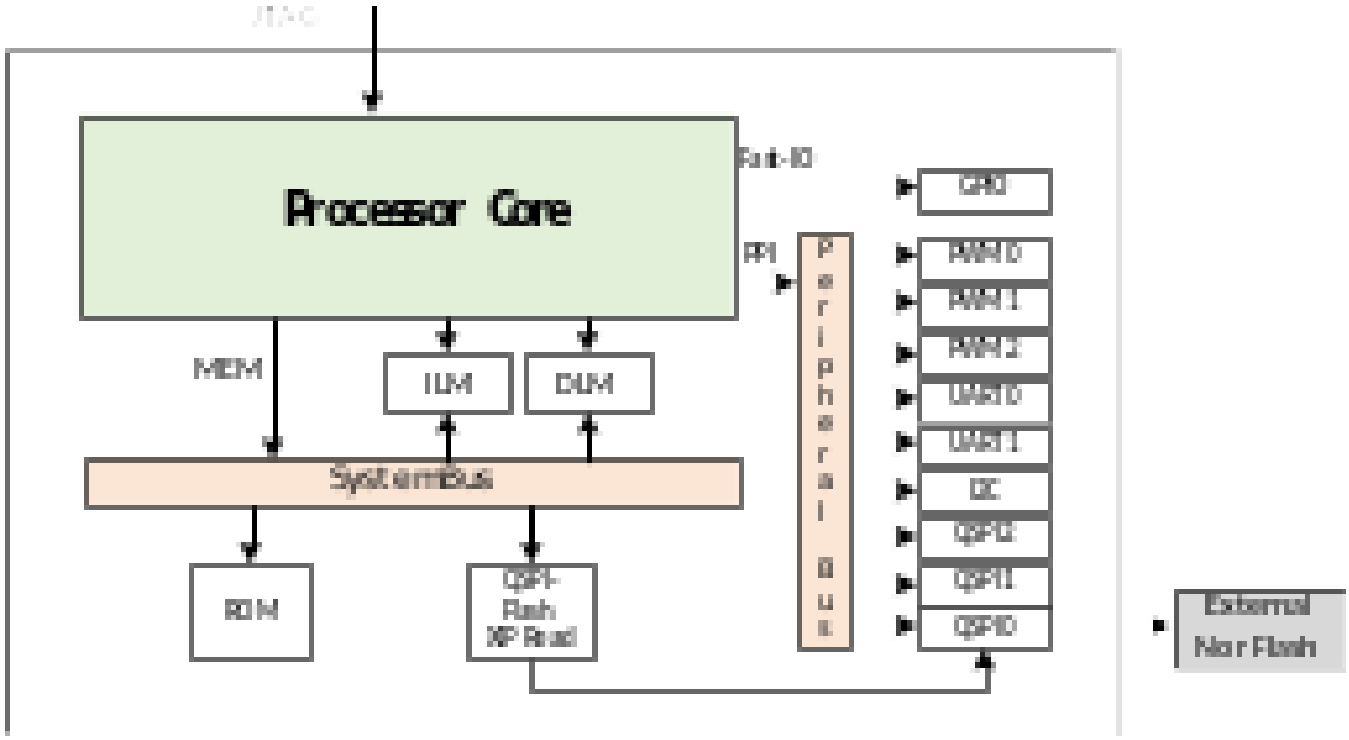

Figure 2-1 SoC Diagram

3. SoC Bus

The SoC have Nuclei defined IoT on-chip bus, Internal Chip Bus (ICB).

3.1. ICB Bus Signals

The ICB is composed with two independent in-order channels as depicted in Figure 3 -2. Each channel is handshake based protocol and similar to AXI bus style.

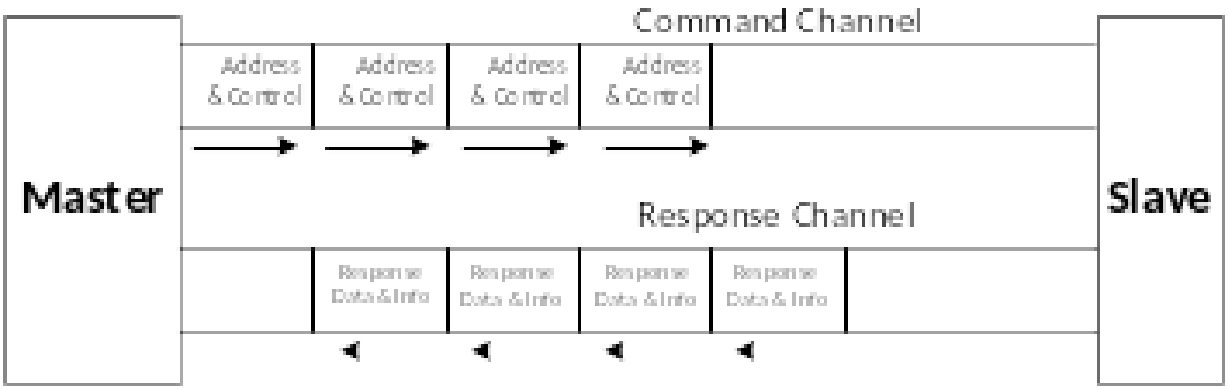

Figure 3-2 ICB Bus Channels

The signals of each ICB channel are as shown in Table 3 -1.

Table 3-1 ICB Bus Signals

Channel	Direction	Width	Signal Name	Description
Command Channel	Output	1	icb_cmd_valid	Command channel valid handshake signal
	Input	1	icb_cmd_ready	Command channel ready handshake signal
	Output	32	icb_cmd_addr	Command channel address
	Output	1	icb_cmd_read	Read or write indication.
	Output	32	icb_cmd_wdata	Write data signal. The format is same as in AXI protocol.
	Output	4	icb_cmd_wmask	Write mask signal. The format is same as in AXI protocol.
Response Channel	Input	1	icb_rsp_valid	Response channel valid handshake signal
	Output	1	icb_rsp_ready	Response channel ready handshake signal
	Input	32	icb_rsp_rdata	Response channel data signal. The format is same as in AXI protocol.
	Input	1	icb_rsp_err	Response channel error indication.

3.2. ICB Bus Protocol

The detailed bus protocol of ICB is introduced in Chapter 12 of Chinese Book《手把手教你设计 CPU——RISC-V 处理器篇》. And for the English user, if want to know more info, please contact Nuclei.

3.3. SoC Bus Structure

The SoC Bus Structure is as depicted in Figure 2 -1, there are two buses:

- The System Bus is to connect the memory resources, such as, ROM, ILM, DLM, and Flash XIP mode XIP read path. For more information of the memory resources, please refer to Chapter 4..
- The Peripheral Bus is to connect the peripherals. For more information of the peripherals, please refer to Chapter 9..

4. Memory Resources of SoC

The SoC contained memory resources including on-chip SRAM resource and External Flash support.

4.1. On-Chip SRAM Resource

The SoC contained on-chip SRAM and ROM, detailed as below.

- ILM with features:
  - Size configurable.
  - Is mainly used to save the instructions, but also can be used to save the data.
  - Base address is allocated as shown in Table 5 -2.
  - The ILM is connected to system bus, so the Core can access it with system bus interface.
- DLM with features:
  - Size configurable.
  - Is mainly used to save the data.
  - Base address is allocated as shown in Table 5 -2.
  - The DLM is connected to system bus, so the Core can access it with system bus interface.
- ROM with features:
  - Size is 4KB.
  - The DLM is connected to system bus, so the Core can access it with system bus interface.
  - There is just 1~2 instructions in ROM, after executed the instructions, Core will to jump to ILM (address at 0x8000_0000). That is to say, in SIMULATION environment, the Core will just jump to ILM after reset and start execution from ILM.

4.2. External Flash Support

The SoC support external SPI Nor Flash with QSPI0 interface, detailed as below.

- The QSPI0 support the eXecute-In-Place (XIP) mode, and it is the default mode after reset.
- With XIP mode, external Flash can be treated as a read-only memory directly accessible (address space is 0x2000_0000 ~ 0x3FFF_FFFF in this SoC).

- Hence, the instructions program can be saved at external Flash (will not be lost even after power down), and then after power-on reset, the Processor Core can fetch instruction directly from the external Flash with XIP mode through this QSPI master interface, and start to execute the programs.
- Please refer to Section 9.2. for more info of QSPI XIP mode.

5. Address Allocation of SoC

The address allocation of the SoC is as shown in Table 5 -2.

Table 5-2 Address Allocation of SoC

	Component	Address Spaces	Description
Core Private Peripherals	TIMER	0x0200_0000 ~ 0x0200_0FFF	TIMER Unit address space.
	ECLIC	0x0C00_0000 ~ 0x0C00_FFFF	ECLIC Unit address space.
	DEBUG	0x0000_0000 ~ 0x0000_0FFF	DEBUG Unit address space.
Memory Resource	ILM	0x8000_0000 ~	ILM address space.
	DLM	0x9000_0000 ~	DLM address space.
	ROM	0x0000_1000 ~ 0x0000_1FFF	Internal ROM.
	Off-Chip QSPI0 Flash Read	0x2000_0000 ~ 0x3FFF_FFFF	QSPI0 with XiP mode read-only address space.
Peripherals	GPIO	0x1001_2000 ~ 0x1001_2FFF	GPIO Unit address space.
	UART0	0x1001_3000 ~ 0x1001_3FFF	First UART address space.
	QSPI0	0x1001_4000 ~ 0x1001_4FFF	First QSPI address space.
	PWM0	0x1001_5000 ~ 0x1001_5FFF	First PWM address space.
	UART1	0x1002_3000 ~ 0x1002_3FFF	Second UART address space.
	QSPI1	0x1002_4000 ~ 0x1002_4FFF	Second QSPI address space.
	PWM1	0x1002_5000 ~ 0x1002_5FFF	Second PWM address space.
	QSPI2	0x1003_4000 ~ 0x1003_4FFF	Third QSPI address space.
	PWM2	0x1003_5000 ~ 0x1003_5FFF	Third PWM address space.
	I2C Master	0x1004_2000 ~ 0x1004_2FFF	I2C Master address space.
Default slave	The other space is write-ignored and read-as zero.		

6. Reset PC Address

After reset, the reset PC address of the Processor Core could be from external Flash (XIP mode), or from the internal ROM.

- In FPGA board, the reset PC address is from external Flash (XIP)
  - In the SoC, “QSPI Flash XIP Read” bus slave port have been allocated with address space of 0x2000_0000 ~ 0x3FFF_FFFF at system bus.
  - In the FPGA version of SoC RTL code, there is a macro check “`ifdef FPGA_SOURCE”, and in this case the input pin “reset_vector” of Processor Core is tied as value “0x2000_0000” , then after reset, the Core will start execution from “QSPI Flash XIP Read”, i.e., the external Flash.
  - Please refer to Chapter 2. for more information of the SoC diagram, and refer to Section 9.2. for more info of QSPI XIP mode.
- In SIMULATION Environment, the reset PC address is from internal ROM.
  - In the SoC, “internal ROM” bus slave port have been allocated with address space of 0x0000_1000 ~ 0x0000_1FFF at system bus.
  - In the SIMULATION version of SoC RTL code, there is a macro check “`ifndef FPGA_SOURCE”, and in this case the input pin “reset_vector” of Processor Core is tied as value “0x0000_1000” , then after reset, the Core will start execution from “Internal ROM”. There is just 1~2 instructions in ROM, after executed the instructions, Core will to jump to ILM (address at 0x8000_0000). That is to say, in SIMULATION environment, the Core will just jump to ILM after reset and start execution from ILM.
  - Please refer to Chapter 2. for more information of the SoC diagram.

7. Ports of SoC

The top level ports of the SoC are as shown in Table 9 -5.

Table 7-3 Ports of SoC

Direction	Name	Description
Input	JTAG TCK	JTAG TCK Signal
Output	JTAG TDO	JTAG TDO Signal
Output	JTAG_DRV_TDO	JTAG TDO Output Enable
Input	JTAG TMS	JTAG TMS Signal
Input	JTAG TDI	JTAG TDI Signal
Bidir	QSPI DQ 3	Quad SPI Data3
Bidir	QSPI DQ 2	Quad SPI Data2
Bidir	QSPI DQ 1	Quad SPI Data1
Bidir	QSPI DQ 0	Quad SPI Data0
Output	QSPI CS	Quad SPI Chip Select
Output	QSPI SCK	Quad SPI Clock
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir	GPIO_0	32 General Purpose I/O
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		
Bidir		

Bidir		
Bidir		
Bidir		
Bidir		
Bidir		

8. Interrupts of SoC

The interrupts of SoC is managed by the interrupt controller of the core, the interrupts are directly connected to the external interrupt interface of the processor Core.

In this SoC, only the GPIO connected to the Core as 32 external interrupt sources, as shown in Table 9 -5.

Table 8-4 Allocation of External Interrupts in SoC

External Interrupts to the Core	Source
0	gpio_0
…	…
31	gpio_31

9. Peripherals of SoC

The Chapter will shortly introduce the peripherals used in the SoC.

9.1. GPIO

There is a GPIO (General Purpose I/O) in the SoC. GPIO provide 32 general purpose I/O ports, the key points of which are:

- Each I/O can be programmed as input or output. If as output, the output value can also be programmed.
- Each I/O can be as the interrupt source, connecting to interrupt controller of core.
- Each I/O can be programmed to IOF (Hardware I/O Functions) mode, in this mode, the I/O is worked as Pin-Mux to serve internal peripherals, e.g., SPI, UART, PWM, etc. Each pin can be reused by two different sources, called IoF0 and IoF1. The Pin-Mux table is as shown in Table 9 -5.

The detailed of the GPIO is introduced in Chapter 6 of Chinese Book《RISC-V 架构嵌入式开发快速入门》. And for the English user, if want to know more info, please contact Nuclei.

Table 9-5 GPIO Pin-Mux

GPIO Pin Number	IOF0	IOF1
0		PWM0_0
1		PWM0_1
2	QSPI1:SS0	PWM0_2
3	QSPI1:SD0/MOSI	PWM0_3
4	QSPI1:SD1/MISO	
5	QSPI1:SCK	
6	QSPI1:SD2	
7	QSPI1:SD3	
8	QSPI1:SS1	
9	QSPI1:SS2	
10	QSPI1:SS3	PWM2_0
11		PWM2_1
12	I2C:SDA	PWM2_2
13	I2C:SCL	PWM2_3
14		
15		
16	UART0:RX	
17	UART0:TX	

18		
19		PWM1_1
20		PWM1_0
21		PWM1_2
22		PWM1_3
23		
24	UART1:RX	
25	UART1:TX	
26	QSPI2:SS	
27	QSPI2:SD0/MOSI	
28	QSPI2:SD1/MISO	
29	QSPI2:SCK	
30	QSPI2:SD2	
31	QSPI2:SD3	

9.2. QSPI

There are 3 independent QSPI in the SoC, which all use GPIO Pin-Mux to communicate with outside.

- One QSPI for Flash:
  - There is one QSPI master in the SoC which has dedicated SoC ports to communicate with external FLASH (e.g. Nor Flash with SPI interface).
  - This QSPI support the eXecute-In-Place (XIP) mode, and it is the default mode after reset. With XIP mode, external Flash can be treated as a read-only memory directly accessible. Hence, the instructions program can be saved at external Flash (will not be lost even after power down), and then after power-on reset, the Processor Core can fetch instruction directly from the external Flash with XIP mode through this QSPI master interface, and start to execute the programs.
- Another two QSPI :
  - There are two QSPI master (without XIP mode supported), which all use GPIO Pin-Mux to communicate with outside. One QSPI master has 4 CS (Chip Select) signals, another has 1 CS signal.

The detailed of the QSPI is introduced in Chapter 6 of Chinese Book《RISC-V 架构嵌入式开发快速入门》. And for the English user, if want to know more info, please contact Nuclei.

9.3. UART

There are 2 independent UART (Universal Asynchronous Receiver-Transmitter) in the SoC, which all use GPIO Pin-Mux to communicate with outside.

The detailed of the UART is introduced in Chapter 6 of Chinese Book《RISC-V 架构嵌入式开发快速入门》. And for the English user, if want to know more info, please contact Nuclei.

9.4. PWM

There are 3 independent PWM (Pulse-Width Modulator) in the SoC, two of them are 16bits wide, and another one is 8bits wide, which all use GPIO Pin-Mux to communicate with outside.

The detailed of the PWM is introduced in Chapter 6 of Chinese Book《RISC-V 架构嵌入式开发快速入门》. And for the English user, if want to know more info, please contact Nuclei.

9.5. I2C

There is an I2C master in the SoC, which use GPIO Pin-Mux to communicate with outside.

The detailed of this I2C master is introduced in Chapter 6 of Chinese Book《RISC-V 架构嵌入式开发快速入门》. And for the English user, if want to know more info, please contact Nuclei.