Gowin_EMPU_M3 Software Programming
Reference Manual

IPUG922-1.1E, 07/16/2021

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
04/03/2020	1.0E	Initial version published.
07/16/2021	1.1E	• Known issues of read and write for SPI full-duplex fixed; • Known issues of GPIO initialization fixed; • The version of MCU software updated.

Contents

List of Figures

List of Tables

1 Software Programming Library

Gowin_EMPU_M3 supports software programming library: Gowin_EMPU_M3\src\c_lib

Two Gowin_EMPU_M3 software programming methods are supported:

- Cortex-M3 Software Programming
- Embedded RTOS software programming

1.1 MCU Cortex-M3 Software Programming

Gowin_EMPU_M3 software programming library supports MCU Cortex-M3 software programming, as shown in Table 1-1.

Table 1-1 MCU Cortex-M3 Software Programming

File	Description
startup_GOWIN_M3.s	Startup
core_cm3.c	Cortex-M3 register definition
GOWIN_M3.h	Definition of interrupt vector table, register, and address mapping
system_GOWIN_M3.c	System initialization and system clock definition
GOWIN_M3_flash.ld	GMD EDA Flash linker
GOWIN_M3_gpio.c	GPIO drive function definition
GOWIN_M3_ethernet.c	Ethernet drive function definition
GOWIN_M3_ddr3.c	I2C drive function definition
GOWIN_M3_spi_flash.c	SPI-Flash read, write and erasure drive function definition
GOWIN_M3_timer.c	Timer0/1 drive function definition
GOWIN_M3_wdog.c	WatchDog drive function definition
GOWIN_M3_uart.c	UART0/1 drive function definition
GOWIN_M3_rtc.c	RTC driving function definition
GOWIN_M3_i2c.c	I2C Master drive function definition
GOWIN_M3_spi.c	SPI Master drive function definition
GOWIN_M3_misc.c	Interrupt priority management and SysTick

File	Description
GOWIN_M3_it.c	Interrupt handler function definition
retarget.c	UART0 printf function redirection definition
malloc.c	Dynamic memory management malloc and free function redirection definition

1.2 Embedded RTOS Programming

Gowin_EMPU_M3 supports software programming in following two operating systems:

- uC/OS-III
- FreeRTOS

2 Memory System

2.1 Standard Peripherals Memory Mapping

The standard peripherals memory mapping addresses for Gowin_EMPU_M3 are as shown in Table 2-1.

Table 2-1 Standard Peripheral Memory Mapping Definition

Standard Peripheral	Type	Address Mapping	Description
Instruction Memory	–	0x00000000	16KB, 32KB, 64KB, 128KB
Data Memory	–	0x20000000	16KB, 32KB, 64KB, 128KB
TIMER0	TIMER_TypeDef	0x50000000	Timer 0
TIMER1	TIMER_TypeDef	0x50001000	Timer1
UART0	UART_TypeDef	0x50004000	UART0
UART1	UART_TypeDef	0x50005000	UART1
Watch Dog	WDOG_TypeDef	0x50008000	WatchDog
RTC	RTC_RegDef	0x50006000	Real-time clock
SPI_FLASH	SPI_FLASH_RegDef	0x50003000	SPI Flash
I2C	I2C_TypeDef	0x5000A000	I2C
SPI	SPI_TypeDef	0x5000B000	Serial Peripheral Interface
GPIO0	GPIO_TypeDef	0x40000000	GPIO port
Ethernet	ETH_RegDef	0x46000000	Ethernet
DDR3	DDR3_RegDef	0x55000000	DDR3 Memory
APB2 Extension	–	0x51000000	APB2 extented interface
AHB2 Extension	–	0x52000000	AHB Extension Interface

2.2 Core System Memory Mapping

The Gowin_EMPU_M3 system memory mapping definition is as shown in Table 2-2.

Table 2-2 Core System Memory Mapping Definition

System Control	Type	Address Mapping	Description
ITM	ITM_Type	0xE0000000	ITM configuration struct
DWT	DWT_Type	0xE0001000	DWT configuration struct
CoreDebug	CoreDebug_Type	0xE000EDF0	Core Debug configuration struct
ETM	ETM_Type	0xE0041000	ETM configuration struct
SysTick	SysTick_Type	0xE000E010	SysTick configuration struct
NVIC	NVIC_BASE	0xE000E100	NVIC configuration struct
SCnSCB	SCnSCB_Type	0xE000E000	System control Register not in SCB
SCB	SCB_Type	0xE000ED00	SCB configuration struct
TPIU	TPIU_Type	0xE0040000	TPIU configuration struct
MPU	MPU_Type	0xE000ED90	MPU configuration struct

3 Interrupt Handler

The nested vector interrupt controller (NVIC) includes the following features:

- Supports up to 48 low latency interrupts
- Provides 16 interrupt handling signals available to users
- Supports programmable interrupt priorities of 8~ 256
- Low latency interrupts and exception handler
- Interrupt signal edge or pulse detection
- Interrupt priority adjustment

Gowin_EMPU_M3 interrupt controller definition is shown in Table 3-1.

Table 3-1 Interrupt Controller Definition

Address	Interrupt	Number	Description
0x00000000	__StackTop	–	Top of Stack
0x00000004	Reset_Handler	–	Reset Handler
0x00000008	NMI_Handler	-14	NMI Handler
0x0000000C	HardFault_Handler	-13	Hard Fault Handler
0x00000010	MemManage_Handler	-12	MPU Fault Handler
0x00000014	BusFault_Handler	-11	Bus Fault Handler
0x00000018	UsageFault_Handler	-10	Usage Fault Handler
0x0000001C	0	–	Reserved
0x00000020	0	–	Reserved
0x00000024	0	–	Reserved
0x00000028	0	–	Reserved
0x0000002C	SVC_Handler	-5	SVCall Handler
0x00000030	DebugMon_Handler	-4	Debug Monitor Handler
0x00000034	0	–	Reserved
0x00000038	PendSV_Handler	-2	PendSV Handler
0x0000003C	SysTick_Handler	-1	SysTick Handler
0x00000040	UART0_Handler	0	16+ 0: UART 0 RX and TX

Address	Interrupt	Number	Description
			Handler
0x00000044	UART1_Handler	1	16+ 1: UART 1 RX and TX Handler
0x00000048	TIMER0_Handler	2	16+ 2: Timer 0 Handler
0x0000004C	TIMER1_Handler	3	16+ 3: Timer 1 Handler
0x00000050	GPIO0_Handler	4	16+ 4: GPIO Port 0 Combined Handler
0x00000054	UARTOVF_Handler	5	16+ 5: UART 0,1 Overflow Handler
0x00000058	RTC_Handler	6	16+ 6: RTC Handler
0x0000005C	I2C_Handler	7	16+ 7: I2C Handler
0x00000060	Interrupt8_Handler	8	16+ 8: Interrupt 8 Handler
0x00000064	ETH_Handler	9	16+ 9: ETH Handler
0x00000068	Interrupt10_Handler	10	16+10: Interrupt 10 Handler
0x0000006C	Interrupt11_Handler	11	16+11: Interrupt 11 Handler
0x00000070	Interrupt12_Handler	12	16+12: Interrupt 12 Handler
0x00000074	Interrupt13_Handler	13	16+13: Interrupt 13 Handler
0x00000078	Interrupt14_Handler	14	16+14: Interrupt 14 Handler
0x0000007C	Interrupt15_Handler	15	16+15: Interrupt 15 Handler
0x00000080	GPIO0_0_Handler	16	16+16: GPIO0_0 Handler
0x00000084	GPIO0_1_Handler	17	16+17: GPIO0_1 Handler
0x00000088	GPIO0_2_Handler	18	16+18: GPIO0_2 Handler
0x0000008C	GPIO0_3_Handler	19	16+19: GPIO0_3 Handler
0x00000090	GPIO0_4_Handler	20	16+20: GPIO0_4 Handler
0x00000094	GPIO0_5_Handler	21	16+21: GPIO0_5 Handler
0x00000098	GPIO0_6_Handler	22	16+22: GPIO0_6 Handler
0x0000009C	GPIO0_7_Handler	23	16+23: GPIO0_7 Handler
0x000000A0	GPIO0_8_Handler	24	16+24: GPIO0_8 Handler
0x000000A4	GPIO0_9_Handler	25	16+25: GPIO0_9 Handler
0x000000A8	GPIO0_10_Handler	26	16+26: GPIO0_10 Handler
0x000000AC	GPIO0_11_Handler	27	16+27: GPIO0_11 Handler
0x000000B0	GPIO0_12_Handler	28	16+28: GPIO0_12 Handler
0x000000B4	GPIO0_13_Handler	29	16+29: GPIO0_13 Handler
0x000000B8	GPIO0_14_Handler	30	16+30: GPIO0_14 Handler
0x000000BC	GPIO0_15_Handler	31	16+31: GPIO0_15 Handler
0x000000C0	USER_INT0_Handler	32	16+32: User Interrupt 0 Handler
0x000000C4	USER_INT1_Handler	33	16+33: User Interrupt 1 Handler
0x000000C8	USER_INT2_Handler	34	16+34: User Interrupt 2 Handler
0x000000CC	USER_INT3_Handler	35	16+35: User Interrupt 3 Handler
0x000000D0	USER_INT4_Handler	36	16+36: User Interrupt 4 Handler

Address	Interrupt	Number	Description
0x000000D4	USER_INT5_Handler	37	16+37: User Interrupt 5 Handler
0x000000D8	USER_INT6_Handler	38	16+38: User Interrupt 6 Handler
0x000000DC	USER_INT7_Handler	39	16+39: User Interrupt 7 Handler
0x000000E0	USER_INT8_Handler	40	16+40: User Interrupt 8 Handler
0x000000E4	USER_INT9_Handler	41	16+41: User Interrupt 9 Handler
0x000000E8	USER_INT10_Handler	42	16+42: User Interrupt 10 Handler
0x000000EC	USER_INT11_Handler	43	16+43: User Interrupt 11 Handler
0x000000F0	USER_INT12_Handler	44	16+44: User Interrupt 12 Handler
0x000000F4	USER_INT13_Handler	45	16+45: User Interrupt 13 Handler
0x000000F8	USER_INT14_Handler	46	16+46: User Interrupt 14 Handler
0x000000FC	USER_INT15_Handler	47	16+47: User Interrupt 15 Handler

4 Universal Asynchronous Receiver/Transmitter

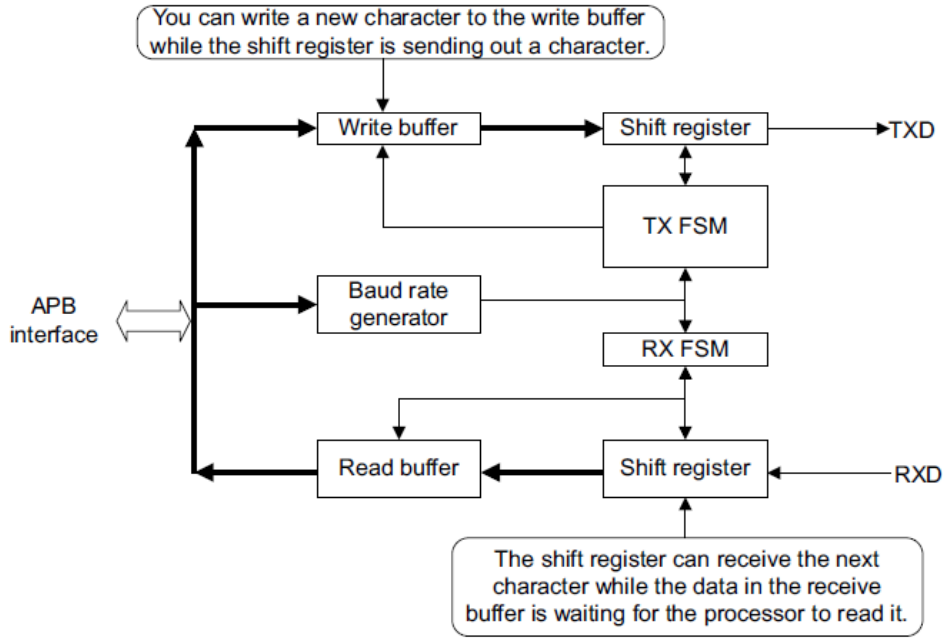
4.1 Features

There are two UARTs accessed by the APB in Gowin_EMPU_M3:

- The max. baud rate is 921.6Kbit/s
- No parity bit
- 8-bit data bit
- 1-bit stop bit

UART Buffering is as shown in .

Figure 4-1 UART Buffering



The UART supports the High Speed Test Mode (HSTM). When the register CTRL[6] is set to 1, the serial data is transmitted one bit per cycle, and the text information can be transmitted in a short time.

The baud rate divider register should be set when using UART. For example, if the APB1 bus frequency is running at 12MHz and the baud rate is required to be 9600, the baud rate divider register can be set to 12000000/9600=1250.

4.2 Register Description

The UART register description is as shown in Table 4-1.

Table 4-1 UART Register Description

Name	Address Offset	Type	Width	Initial Value	Description
DATA	0x000	RW	8	0x--	[7:0] Data Value
STATE	0x004	RW	4	0x0	[3] RX buffer overrun, write 1 to clear [2] TX buffer overrun, write 1 to clear [1] RX buffer full, read-only [0] TX buffer full, read-only
CTRL	0x008	RW	7	0x00	[6] High speed test mode for TX only [5] RX overrun interrupt enable [4] TX overrun interrupt enable [3] RX interrupt enable [2] TX interrupt enable [1] RX enable [0] TX enable
INTSTATUS /INTCLEAR	0x00C	RW	4	0x0	[3] RX overrun interrupt, write 1 to clear [2] TX overrun interrupt, write 1 to clear [1] RX interrupt, write 1 to clear [0] TX interrupt, write 1 to clear
BAUDDIV	0x010	RW	20	0x00000	[19:0] Baud rate divider, the minimum number is 16

4.3 Initialization Definition

The UART initialization definition is shown in Table 4-2.

Table 4-2 UART Initialization Definition

Name	Type	Value	Description
UART_BaudRate	uint32_t	Max 921.6Kbit/s	Baud rate
UART_Mode	UARTMode_TypeDef	ENABLE/DISABLE	Enable/Disable TX/RX mode
UART_Int	UARTInt_TypeDef	ENABLE/DISABLE	Enable/Disable TX/RX interrupt
UART_Ovr	UARTOvr_TypeDef	ENABLE/DISABLE	Enable/Disable TX/RX overrun interrupt
UART_Hstm	FunctionalState	ENABLE/DISABLE	Enable/Disable TX hisgh speed test mode

4.4 Usage of Drivers

The usage of UART drivers is shown in Table 4-3.

Table 4-3 Usage of UART Drivers

Name	Description
UART_Init	Initializes UARTx
UART_GetRxBufferFull	Returns UARTx RX buffer full status
UART_GetTxBufferFull	Returns UARTx TX buffer full status
UART_GetRxBufferOverrunStatus	Returns UARTx RX buffer overrun status
UART_GetTxBufferOverrunStatus	Returns UARTx TX buffer overrun status
UART_ClearRxBufferOverrunStatus	Clears Rx buffer overrun status
UART_ClearTxBufferOverrunStatus	Clears Tx buffer overrun status
UART_SendChar	Sends a character to UARTx TX buffer
UART_SendString	Sends a string to UARTx TX buffer
UART_ReceiveChar	Receives a character from UARTx RX buffer
UART_GetBaudDivider	Returns UARTx baud rate divider value
UART_GetTxIRQStatus	Returns UARTx TX interrupt status
UART_GetRxIRQStatus	Returns UARTx RX interrupt status
UART_ClearTxIRQ	Clears UARTx TX interrupt status
UART_ClearRxIRQ	Clears UARTx RX interrupt status
UART_GetTxOverrunIRQStatus	Returns UARTx TX overrun interrupt status
UART_GetRxOverrunIRQStatus	Returns UARTx RX overrun interrupt status
UART_ClearTxOverrunIRQ	Clears UARTx TX overrun interrupt request
UART_ClearRxOverrunIRQ	Clears UARTx RX overrun interrupt request
UART_SetHSTM	Sets UARTx TX high speed test mode
UART_ClrHSTM	Clears UARTx TX high speed test mode

4.5 Reference Design

Gowin_EMPU_M3 provides UART reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

5 Timer

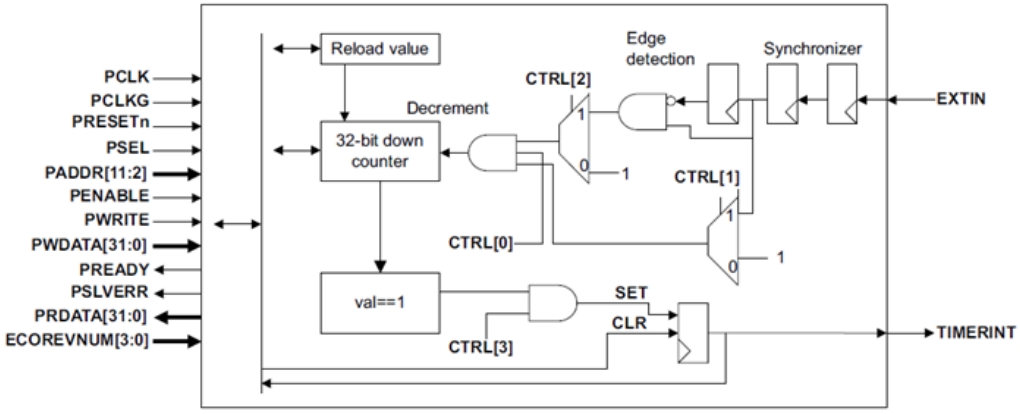
5.1 Features

Gowin_EMPU_M3 contains two synchronization standard timers accessed by APB:

- 32-bit counter
- Supports the interrupt request signal
- Supports the external input signal EXTIN to enable clock
- TIMER0: EXTIN connects to GPIO [1]
- TIMER1: EXTIN connects to GPIO [6]

The Timer structure is shown in Figure 5-1.

Figure 5-1 TIMER



5.2 Register Description

The Timer register definition is as shown in Table 5-1.

Table 5-1 Timer Register Definition

Name	Address Offset	Type	Width	Initial Value	Description
CTRL	0x000	RW	4	0x0	[3] Timer interrupt enable [2] Select external input as clock [1] Select external input as enable [0] Enable
VALUE	0x004	RW	32	0x000 00000	[31:0] Current value
RELOAD	0x008	RW	32	0x000 00000	[31:0] Reload value,writing to this register sets the current value
INTSTATUS /INTCLEAR	0x00C	RW	1	0x0	[0] Timer interrupt,write 1 to clear

5.3 Initialization Definition

The Timer initialization definition is as shown in Table 5-2.

Table 5-2 Timer Initialization Definition

Name	Type	Value	Description
Reload	uint32_t	–	Reload value
TIMER_Int	TIMERInt_TypeDef	SET/RESET	Enable/Disable interrupt
TIMER_Exti	TIMERExti_TypeDef	–	External input as enable or clock

5.4 Usage of Drivers

The usage of Timer drivers is shown in Table 5-3.

Table 5-3 Usage of Timer Drivers

Name	Description
TIMER_Init	Initializes TIMERx
TIMER_StartTimer	Starts TIMERx
TIMER_StopTimer	Stops TIMERx
TIMER_GetIRQStatus	Returns TIMERx interrupt status
TIMER_ClearIRQ	Clears TIMERx interrupt status
TIMER_GetReload	Returns TIMERx reload value
TIMER_SetReload	Sets TIMERx reload value
TIMER_GetValue	Returns TIMERx current value
TIMER_SetValue	Sets TIMERx current value
TIMER_EnableIRQ	Enable TIMERx interrupt request

Name	Description
TIMER_DisableIRQ	Disable TIMERx interrupt request

5.5 Reference Design

Gowin_EMPU_M3 provides Timer reference design in ARM Keil MDK (V5.26 and above0 and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

6 WatchDog

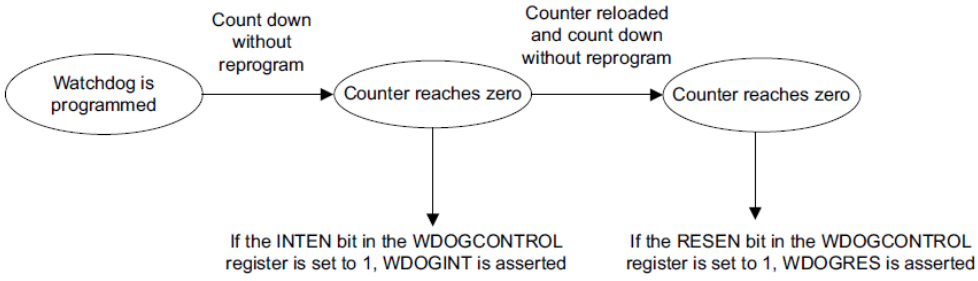
6.1 Features

There is one WatchDog accessed by the APB in Gowin_EMPU_M3:

- A 32-bit down-counter initialized by the LOAD register;
- Supports interrupt request;
- When the clock is enabled, the counter is decremented by the rising edge of the WDOGCLK signal;
- A reset request is generated and the counter is stopped when the counter is decremented to 0 to monitor interrupts;
- Provide reset caused by software crash and method.

WatchDog operation is shown in Figure 6-1.

Figure 6-1 WatchDog Operation



6.2 Register Description

The WatchDog register description is as shown in Table 6-1.

Table 6-1 WatchDog Register Definition

Name	Address Offset	Type	Width	Initial Value	Description
LOAD	0x00	RW	32	0xFFFF FFFF	The value from which the counter is to decrement
VALUE	0x04	RO	32	0xFFFF FFFF	The current value of the decrementing counter

Name	Address Offset	Type	Width	Initial Value	Description
CTRL	0x08	RW	2	0x0	[1] Enable reset output [0] Enable the interrupt
INTCLR	0x0C	WO	–	–	Clear the watchdog interrupt and reloads the counter
RIS	0x10	RO	1	0x0	Raw interrupt status from the counter
MIS	0x14	RO	1	0x0	Enable interrupt status from the counter
RESERVED	0xC00-0x014	–	–	–	Reserved
LOCK	0xC00	RW	32	0x0000 0000	[32:1] Enable register writes [0] Register write enable status
RESERVED	0xF00-0xC00	–	–	–	Reserved
ITCR	0xF00	RW	1	0x0	Integration test mode enable
ITOP	0xF04	WO	2	0x0	[1] Integration test WDOGRES value [0] Integration test WDOGINT value

6.3 Initialization Definition

The WatchDog initialization definition is as shown in Table 6-2.

Table 6-2 WatchDog Initialization Definition

Name	Type	Value	Description
WDOG_Reload	uint32_t	–	Reload value
WDOG_Lock	WDOGLock_TypeDef	SET/RESET	Enable/Disable lock register write access
WDOG_Res	WDOGRes_TypeDef	SET/RESET	Enable/Disable reset flag
WDOG_Int	WDOGInt_TypeDef	SET/RESET	Enable/Disable interrupt flag
WDOG_ITMode	WDOGMode_Typedef	SET/RESET	Enable/Disable integration test mode flag

6.4 Usage of Drivers

The usage of WatchDog drivers is shown in Table 6-3.

Table 6-3 Usage of WatchDog Drivers

Name	Description
WDOG_Init	Initializes WatchDog
WDOG_RestartCounter	Restart watchdog counter
WDOG_GetCounterValue	Returns counter value
WDOG_SetResetEnable	Sets reset enable

Name	Description
WDOG_GetResStatus	Returns reset status
WDOG_SetIntEnable	Sets interrupt enable
WDOG_GetIntStatus	Returns interrupt enable
WDOG_ClrIntEnable	Clears interrupt enable
WDOG_GetRawIntStatus	Returns raw interrupt status
WDOG_GetMaskIntStatus	Returns masked interrupt status
WDOG_LockWriteAccess	Disable write access all registers
WDOG_UnlockWriteAccess	Enable write access all registers
WDOG_SetITModeEnable	Sets integration test mode enable
WDOG_ClrITModeEnable	Clears integration test mode enable
WDOG_GetITModeStatus	Returns integration test mode status
WDOG_SetITOP	Sets integration test output reset or interrupt
WDOG_GetITOPResStatus	Returns integration test output reset status
WDOG_GetITOPIntStatus	Returns integration test output interrupt status
WDOG_ClrITOP	Clears integration test output reset or interrupt

6.5 Reference Design

Gowin_EMPU_M3 provides WatchDog reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

7 GPIO

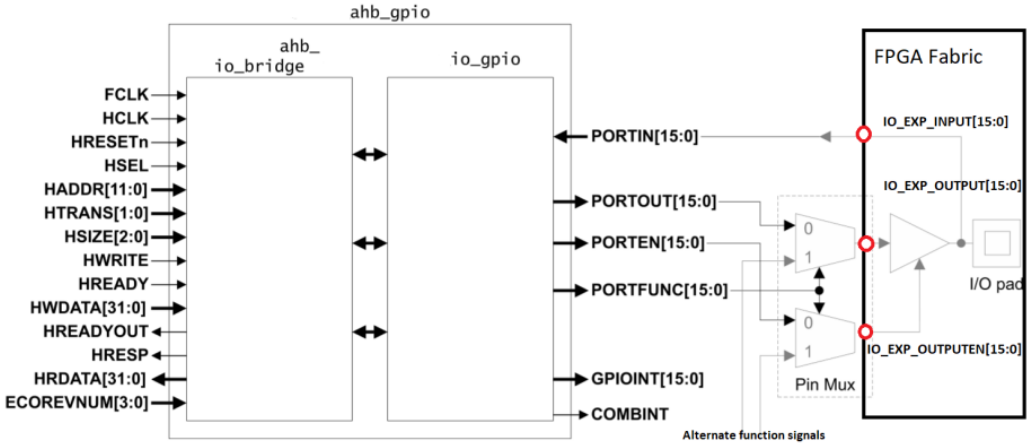
7.1 Features

Gowin_EMPU_M3 contains a GPIO module with a 16-bit input and output interface accessed by AHB:

- Connected with FPGA Fabric
- Supports bit mask
- Supports pin multiplexing

The GPIO block is shown in Figure 7-1.

Figure 7-1 GPIO Block



7.2 Register Description

The GPIO register description is as shown in Table 7-1.

Table 7-1 GPIO Register Description

Name	Address Offset	Type	Width	Initial Value	Description
DATA	0x0000	RW	16	0x----	[15:0] Data value Read Sampled at pin Write to data output register Read back value goes through double flip-flop synchronization logic with delay

Name	Address Offset	Type	Width	Initial Value	Description
					of two cycles.
DATAOUT	0x0004	RW	16	0x0000	[15:0] Data output register value Read current value of data output register write to data output register.
RESERVED	0x0008 -0x000C	–	–	–	Reserved
OUTENSET	0x0010	RW	16	0x0000	[15:0] Output enable set Write 1 to set the output enable bit Write 0 no effect Read back 0 indicates the signal direction as intput. 1 indicates the signal direction as output.
OUTENCLR	0x0014	RW	16	0x0000	[15:0] Output enable clear Write 1 to clear the output enable bit Write 0 no effect Read back 0 indicates the signal direction as intput. 1 indicates the signal direction as output.
ALTFUNCSET	0x0018	RW	16	0x0000	[15:0] Alernative function set Write 1 to set the ALTFUNC bit Write 0 no effect Read back 0 for I/O 1 for an alternate function
ALTFUNCCLR	0x001C	RW	16	0x0000	[15:0] Alernative function clear Write 1 to clear the ALTFUNC bit Write 0 no effect Read back 0 for I/O 1 for an alternate function
INTENSET	0x0020	RW	16	0x0000	[15:0] Interrupt enable set Write 1 to set the enable bit Write 0 no effect Read back 0 indicates interrupt disabled 1 indicates interrupt enabled.
INTENCLR	0x0024	RW	16	0x0000	[15:0] Interrupt enable clear Write 1 to clear the enable bit Write 0 no effect Read back 0 indicates interrupt disabled

Name	Address Offset	Type	Width	Initial Value	Description
					1 indicates interrupt enabled.
INTTYPESET	0x0028	RW	16	0x0000	[15:0] Interrupt type set Write 1 to set the interrupt type bit Write 0 no effect Read back 0 for LOW/HIGH level 1 for falling edge or rising edge
INTTYPECLR	0x002C	RW	16	0x0000	[15:0] Interrupt type clear Write 1 to clear the interrupt type bit Write 0 no effect Read back 0 for LOW/HIGH level 1 for falling edge or rising edge
INTPOLSET	0x0030	RW	16	0x0000	[15:0] Polarity-level, edge IRQ config Write 1 to set the interrupt polarity bit Write 0 no effect Read back 0 for LOW level or falling edge 1 for HIGH level or rising edge
INTPOLCLR	0x0034	RW	16	0x0000	[15:0] Polarity-level, edge IRQ config Write 1 to clear the interrupt polarity bit Write 0 no effect Read back 0 for LOW level or falling edge 1 for HIGH level or rising edge
INTSTATUS /INTCLEAR	0x0038	RW	16	0x0000	[15:0] Write IRQ status clear register Write 1 to clear interrupt request Write 0 no ffect Read back IRQ status register
MASKLOWBYTE	0x0400 -0x07FC	RW	16	0x----	Lower 8-bits masked access [9:2] of the address value are used as enable bit mask for the access [15:8] not used [7:0] Data for lower byte access,with [9:2] of address value used as enable mask for each bit
MASKHIGHBYTE	0x0800 -0x0BFC	RW	16	0x----	Higher 8-bits masked access [9:2] of the address value are

Name	Address Offset	Type	Width	Initial Value	Description
					used as enable bit mask for the access [15:8] Data for higher byte access, with [9:2] of address value used as enable mask for each bit [7:0] not used
RESERVED	0x0C00 -0x0FCF	–	–	–	Reserved

7.3 Initialization Definition

The GPIO initialization definition is shown in Table 7-2.

Table 7-2 GPIO Initialization Definition

Name	Type	Value	Description
GPIO_Pin	uint32_t	GPIO_Pin_0 GPIO_Pin_1 GPIO_Pin_2 GPIO_Pin_3 GPIO_Pin_4 GPIO_Pin_5 GPIO_Pin_6 GPIO_Pin_7 GPIO_Pin_8 GPIO_Pin_9 GPIO_Pin_10 GPIO_Pin_11 GPIO_Pin_12 GPIO_Pin_13 GPIO_Pin_14 GPIO_Pin_15	16 bits GPIO Pins
GPIO_Mode	GPIOMode_TypeDef	GPIO_Mode_IN GPIO_Mode_OUT GPIO_Mode_AF	16 bits GPIO Pins mode
GPIO_Int	GPIOInt_TypeDef	GPIO_Int_Disable GPIO_Int_Low_Level GPIO_Int_High_Level GPIO_Int_Falling_Edge GPIO_Int_Rising_Edge	16 bits GPIO Pins interrupt

7.4 Usage of Drivers

The usage of GPIO drivers is shown in Table 7-3.

Table 7-3 Usage of GPIO Drivers

Name	Description
GPIO_Init	Initializes GPIOx
GPIO_SetOutEnable	Sets GPIOx output enable
GPIO_ClrOutEnable	Clears GPIOx output enable
GPIO_GetOutEnable	Returns GPIOx output enable
GPIO_SetBit	GPIO output one
GPIO_ResetBit	GPIO output zero
GPIO_WriteBits	GPIO output
GPIO_ReadBits	GPIO input
GPIO_SetAltFunc	Sets GPIOx alternate function enable
GPIO_ClrAltFunc	Clears GPIOx alternate function enable
GPIO_GetAltFunc	Returns GPIOx alternate function enable
GPIO_IntClear	Clears GPIOx interrupt request
GPIO_GetIntStatus	Returns GPIOx interrupt status
GPIO_SetIntEnable	Sets GPIOx interrupt enable Returns GPIOx interrupt status
GPIO_ClrIntEnable	Clears GPIOx interrupt enable Returns GPIOx interrupt enable
GPIO_SetIntHighLevel	Setups GPIOx interrupt as high level
GPIO_SetIntRisingEdge	Setups GPIOx interrupt as rising edge
GPIO_SetIntLowLevel	Setups GPIOx interrupt as low level
GPIO_SetIntFallingEdge	Setups GPIOx interrupt as falling edge
GPIO_MaskedWrite	Setups GPIOx output value using masked access

7.5 Reference Design

Gowin_EMPU_M3 provides GPIO reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

8 I2C

8.1 Features

There is one I2C Master module accessed by the APB in Gowin_EMPU_M3:

- APB bus interface
- Compliant with I2C protocol;
- Bus arbitration and arbitration lost detection;
- Bus busy detection;
- Interrupt flag generation;
- Start / Stop / Repeated Start / Acknowledge generation;
- Start / Stop / Repeated Start detection;
- Supports 7-bit addressing mode.

8.2 Register Description

The I2C Master register description is as shown in Table 8-1.

Table 8-1 I2C Master Register Description

Name	Address Offset	Type	Width	Initial Value	Description
PRER	0x00	RW	32	0x000 0FFFF	Clock prescale register [31:15] Reserved [15:0] Prescale value = sys_clk/(5*SCL)-1
CTR	0x04	RW	32	0x000 00000	[31:8] Reserved [7] Enable I2C function [6] Enable I2C interrupt [5:0] Reserved
TXR	0x08	WO	32	0x000 00000	[31:8] Reserved [7:1] Next transmission data [0] Data direction
RXR	0x0C	RO	32	0x000 00000	[31:8] Reserved

Name	Address Offset	Type	Width	Initial Value	Description
					[7:0] Last received data
CR	0x010	WO	32	0x000 00000	[31:8] Reserved [7] STA, Start transmission status [6] STO, Over transmission status [5] RD, Read enable, read data from slave [4] WR, Write enable, write data to slave [3] Acknowledge [2:1] Reserved [0] Interrupt acknowledge
SR	0x14	RO	32	0x000 00000	[31:8] Reserved [7] Receive acknowledge signal from slave [6] I2C busy status [5] Arbitration loss [4:2] Reserved [1] Data transmission status flag [0] Interrupt flag

8.3 Usage of Drivers

The usage of I2C Master drivers is shown in Table 8-2.

Table 8-2 Usage of I2C Drivers

Name	Description
I2C_Init	I2C Initialization
I2C_SendByte	Send a byte to I2C bus
I2C_SendBytes	Send multiple bytes to I2C bus
I2C_SendData	Send multiple bytes to I2C bus once time
I2C_ReceiveByte	Read a byte from I2C bus
I2C_ReadBytes	Read multiple bytes from I2C bus
I2C_ReceiveData	Read multiple bytes from I2C bus once time
I2C_Rate_Set	Set I2C traffic rate
I2C_Enable	Enable I2C bus
I2C_UnEnable	Disable I2C bus
I2C_InterruptOpen	Open I2C interrupt
I2C_InterruptClose	Close I2C interrupt

8.4 Reference Design

Gowin_EMPU_M3 provides I2C Master reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

9 Serial Peripheral Interface

9.1 Features

There is one SPI Master module accessed by APB in Gowin_EMPU_M3:

- APB bus interface
- Full duplex synchronous serial data transmission
- Supports Master working mode
- Supports configurable clock polarity and phase
- Configurable serial clock frequency generated by SPI
- 8-bit width for data receive register and data transmission register

9.2 Register Description

The definition of SPI Master register is as shown in Table 9-1.

Table 9-1 Definition of SPI Master Register

Name	Address Offset	Type	Width	Initial Value	Description
RDATA	0x00	RO	32	0x000 00000	Read data register [31:8] Reserved [7:0] Read data
WDATA	0x04	WO	32	0x000 00000	Write data register [31:8] Reserved [7:0] Write data
STATUS	0x08	RW	32	0x000 00000	[31:8] Reserved [7] Overflow error status [6] Receive ready status [5] Transmit ready status [4] Be transmitting [3] Transmit overrun error status [2] Receive overrun error status [1:0] Reserved
SSMASK	0x0C	RW	32	0x000	[31:1] Reserved

Name	Address Offset	Type	Width	Initial Value	Description
				00000	[0] Select and enable slave
CTRL	0x10	RW	32	0x000 00000	[31:5] Reserved [4:3] Clock selected, CLK_I / 2/4/6/8 [2] Clock polarity [1] Clock phase [0] Direction, 1 is MSB first

9.3 Initialization Definition

The SPI Master initialization definition is shown in Table 9-2.

Table 9-2 SPI Master Initialization Definition

Name	Type	Value	Description
DIRECTION	uint8_t	1/0	MSB/LSB first transmission 0: MSB first; 1: MSB first.
PHASE	uint8_t	1/0	Posedge/Negedge transmit data 0: Sample at posedge edge; 1: Sample at negedge edge.
POLARITY	uint8_t	1/0	Initialize ploarity to one/zero 0: Idle sclk low; 1: Idle sclk high.
CLKSEL	uint8_t	CLKSEL_CLK_DIV_2 CLKSEL_CLK_DIV_4 CLKSEL_CLK_DIV_6 CLKSEL_CLK_DIV_8	Select clock divided 2/4/6/8

9.4 Usage of Drivers

The usage of SPI Master drivers is shown in Table 9-3.

Table 9-3 Usage of SPI Master Drivers

Name	Description
SPI_Init	Initializes SPI
SPI_SetDirection	Sets direction
SPI_ClrDirection	Clears direction
SPI_GetDirection	Returns direction
SPI_SetPhase	Sets phase
SPI_ClrPhase	Clears phase
SPI_GetPhase	Returns phase
SPI_SetPolarity	Sets polarity
SPI_ClrPolarity	Clears polarity
SPI_GetPolarity	Returns polarity

Name	Description
SPI_SetClkSel	Sets clock selection
SPI_GetClkSel	Returns clock selection
SPI_GetToeStatus	Reads transmit overrun error status
SPI_GetRoeStatus	Reads receive overrun error status
SPI_GetTmtStatus	Reads transmitting status
SPI_GetTrdyStatu	Reads transmit ready status
SPI_GetRrdyStatus	Reads receive ready error status
SPI_GetErrStatus	Reads error status
SPI_ClrToeStatus	Clears transmit overrun error status
SPI_ClrRoeStatus	Clears eceive overrun error status
SPI_ClrErrStatus	Clears error status
SPI_ReadWriteByte	Full duplex read and write a byte
SPI_WriteData	Writes data
SPI_ReadData	Reads data
SPI_Select_Slave	Select slave

9.5 Reference Design

Gowin_EMPU_M3 provides SPI Master reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

10 Real-time Clock

10.1 Features

Gowin_EMPU_M3 contains a 32-bit real-time clock (RTC) module accessed by APB:

- APB bus interface
- 32-bit counter
- 32-bit Match register
- 32-bit comparator

MCU reads data, control, and status messages via APB bus interface and RTC. At the rising edge of continuous input clock CLK1HZ (3.072MHz clock input must be provided to RTCSRCCL. The division in RTC is 1Hz.) ,32-bit counter increases.

This counter is not synchronous and can not be overloaded. When system resets, this counter counts from 1 to the max. value (0xFFFFFFFF) and then go back to 0 and keep increasing.

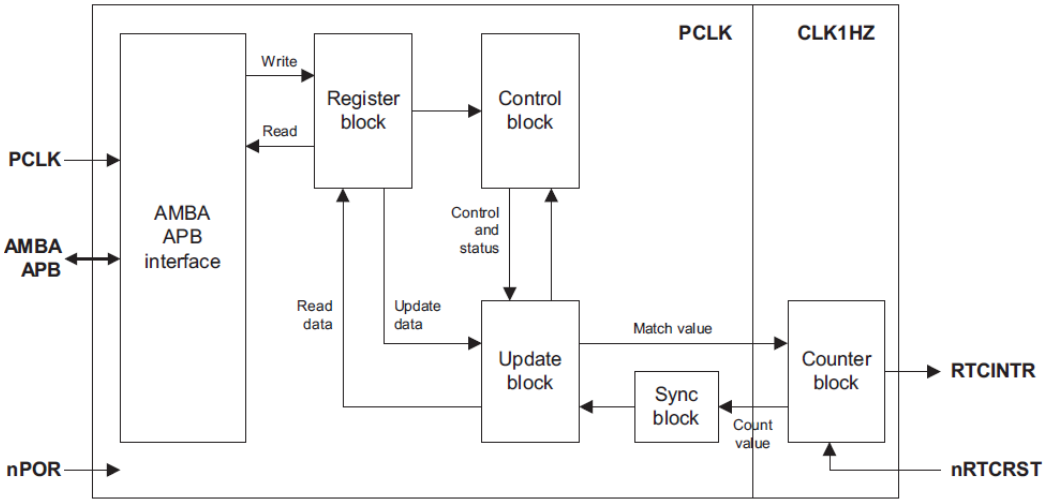
Realizes RTC load or update via the write load register RTC_LOAD_VALUE

Obtains RTC current clock via the read data register RTC_CURRENT_DATA

Programs Match register via the register of write RTC_MATCH_VALUE

The RTC structure is shown in Figure 10-1.

Figure 10-1 RTC Block



10.2 Register Description

The RTC register definition is shown in Table 10-1.

Table 10-1 RTC Register Definition

Name	Address Offset	Type	Width	Initial Value	Description
RTC_CURRENT_DATA	0x000	RO	32	0x000 00000	Data Register [31:0] Current value
RTC_MATCH_VALUE	0x004	RW	32	0x000 00000	Match Register If current value equals match register's value, generate interrupt. [31:0] Match data
RTC_LOAD_VALUE	0x008	RW	32	0x000 00000	Load Register Initialized value, start counter based on this value [31:0] Load data
RTC_CTROLLER_REG	0x00C	RW	32	0x000 00000	Control Register Start RTC counter [31:1] Reserved [0] Start RTC counter
RTC_IMSC	0x010	RW	32	0x000 00000	Interrupt mask set and clear register Enable or disable interrupt [31:1] Reserved [0] Enable interrupt
RTC_RIS	0x014	RO	32	0x000 00000	Raw interrupt status register Get current raw unmasked interrupt status [31:1] Reserved [0] Current raw unmasked interrupt status
RTC_MIS	0x018	RO	32	0x000	Masked interrupt status

Name	Address Offset	Type	Width	Initial Value	Description
				00000	register Get current masked interrupt status [31:1] Reserved [0] Current masked interrupt status
RTC_INTR_CLEAR	0x01C	WO	32	0x000 00000	Interrupt clear register Clear current interrupt [31:1] Reserved [0] Clear current interrupt

10.3 Usage of Drivers

The usage of RTC drivers is shown in Table 10-2.

Table 10-2 Usage of RTC Drivers

Name	Description
RTC_init	Initialize RTC
Get_Current_Value	Get RTC current value of data register
Set_Match_Value	Set RTC match value of match register
Get_Match_Value	Get RTC match value of match register
Set_Load_Value	Set RTC load value of load register
Get_Load_Value	Get RTC load value of load register
Start_RTC	Start RTC counter
Close_RTC	Cloase RTC counter
RTC_Inter_Mask_Set	Set RTC interrupt mask
Get_RTC_Control_value	Get value of control register
RTC_Inter_Mask_Clr	Clear RTC interrupt mask
Get_RTC_Inter_Mask_value	Get RTC interrupt mask
Clear_RTC_interrupt	Clear RTC interrupt

10.4 Reference Design

Gowin_EMPU_M3 provides RTC reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

11 Ethernet

11.1 Features

Gowin_EMPU_M3 contains one Ethernet module accessed by AHB:

- AHB interface
- Realize the function description of MAC layer in the IEEE802.3 protocol
- RGMII/GMII/MII interface
- Supports 10/100/1000M rate
- Supports full duplex and half duplex mode, and conflict detection can be supported in half duplex mode
- Supports users to choose whether to automatically add and verify CRC
- Supports to add pad function automatically
- Supports Ethernet frame classification statistics
- Supports Ethernet frame error statistics
- Supports IFG configurable functions
- Supports Jumbo mode
- Supports Flow Control in full duplex mode
- Supports Management interface mdc, mdio

11.2 Register Description

The Ethernet register definition is shown in Table 12-1.

Table 12-1 Definition of Ethernet Register

Name	Address Offset	Type	Width	Initial Value	Description
ETH_TX_DATA	0x000-0x5FF	WO	32	0x000 00000	Transmit data registers
ETH_RX_DATA	0x000-0x5FFF	RO	32	0x000 00000	Receive data registers
ETH_TX_LENGTH	0x600	RW	32	0x000 00000	Transmit data length [31:11] Reserved [10:0] TX data length
ETH_TX_EN	0x604	RW	32	0x000 00000	Transmit enable [31:1] Reserved [0] Enable TX
ETH_TX_FAIL	0x608	RW	32	0x000 00000	Transmit failed status [31:3] Reserved [2] TX late [1] TX excessive [0] TX failed
ETH_TX_IS	0x60C	RO	32	0x000 00000	Transmit interrupt status [31:1] Reserved [0] TX interrupt status
ETH_TX_IC	0x610	WO	32	0x000 00000	Transmit interrupt clear [31:1] Reserved [0] Clear TX interrupt
ETH_TX_IE	0x614	RW	32	0x000 00000	Transmit interrupt enable [31:1] Reserved [0] Enable TX interrupt
RESERVED_1	0x618-0x67F	-	-	-	Reserved
ETH_RX_LEGHT	0x680	RO	32	0x000 00000	Receive data length
ETH_RX_IS	0x684	RO	32	0x000 00000	Receive interrupt status [31:1] Reserved [0] RX interrupt status
ETH_RX_IE	0x688	RW	32	0x000 00000	Receive interrupt enable [31:1] Reserved [0] Enable RX interrupt
ETH_RX_IC	0x68C	WO	32	0x000 00000	Receive interrupt clear [31:1] Reserved [0] Clear RX interrupt
RESERVED_2	0x690-0x6FFF	-	-	-	Reserved

Name	Address Offset	Type	Width	Initial Value	Description
MIIM_OP_MODE	0x700	RW	32	0x000 00000	MIIM operation mode [31:1] Reserved [0] MIIM operation mode
MIIM_PHY_ADDR	0x704	RW	32	0x000 00000	MIIM PHY address [31:5] Reserved [4:0] MIIM PHY address
MIIM_REG_ADDR	0x708	RW	32	0x000 00000	MIIM reg address [31:5] Reserved [4:0] MIIM reg address
MIIM_WR_DATA	0x70C	RW	32	0x000 00000	MIIM write data [31:16] Reserved [15:0] MIIM write data
MIIM_RD_DATA	0x710	RO	32	0x000 00000	MIIM read data [31:16] Reserved [15:0] MIIM read data
MIIM_IS	0x714	RO	32	0x000 00000	MIIM interrupt status [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid
MIIM_IE	0x718	RW	32	0x000 00000	MIIM interrupt enable [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid
MIIM_IC	0x71C	WO	32	0x000 00000	MIIM interrupt clear [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid
MIIM_OP_EN	0x720	RW	32	0x000 00000	MIIM operation enable [31:1] Reserved [0] Enable MIIM operation
ETH_MODE	0x724	RW	32	0x000 00000	Ethernet operation mode [31:3] Reserved [2:0] duplex mode and speed 000 = full duplex 100M 001 = full duplex 1000M 010 = full duplex 10M 100 = half duplex 100M 110 = half duplex 10M

11.3 Usage of Drivers

The usage of Ethernet drivers is shown in Table 12-2.

Table 12-2 Usage of Ethernet Drivers

Name	Description
eth_init	Initialize Ethernet
tx_int_event	TX interrupt
rx_int_event	RX interrupt
eth_tx	Ethernet TX
eth_set_mode	Set Ethernet duplex mode and speed
miim_wr_int_event	MIIM interface transmits interrupt
miim_rd_int_event	MIIM interface receives interrupt
miim_write	MIIM interface transmits data
miim_receive	MIIM interface receives data

11.4 Reference Design

Gowin_EMPU_M3 provides Ethernet reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

12 DDR3 Memory

12.1 Features

Gowin_EMPU_M3 contains one DDR3 Memory module accessed by AHB:

- AHB interface
- Can connect with DDR3 SDRAM device meeting industrial standard and with the module compatible with JESD79-3F specification
- Supports memory data path width of 16 bits
- Supports UDIMM memory module
- Supports memory chip of x8 data width;
- Programmable burst length 4
- Supports 1:2 clock ratio

12.2 Register Description

The definition of DDR3 register is as shown in Table 13-1.

Table 13-1 DDR3 Register Definition

Name	Address Offset	Type	Width	Initial Value	Description
RESERVED	0x0000	-	-	-	Reserved
WR_ADDR	0x0004	RW	32	0x0	Write address register
WR_DATA	0x0008-0x0014	WO	128	0x0	Write data register
RD_ADDR	0x0018	RW	32	0x0	Read address register
RD_EN	0x001c	RW	32	0x0	Read enable register [31:1] Reserved [0] Read enable 1 = Enable 0 = Disable
RD_DATA	0x0020-0x002c	RO	128	0x0	Read data register
INIT	0x0030	RW	32	0x0	Initialized completely flag register

Name	Address Offset	Type	Width	Initial Value	Description
					[31:1] Reserved [0] Initialzed completely flag
WR_EN	0x0034	RW	32	0x0	Write enable and ending flag register [31:1] Reserved [0] Write enable and ending flag 1 = enable 0 = ending

12.3 Usage of Drivers

The usage of DDR3 drivers is shown in Table 13-2.

Table 13-2 Usage of DDR3 Drivers

Name	Description
DDR3_Init	Initialize DDR3
DDR3_Read	Read data from DDR3
DDR3_Write	Write data into DDR3

12.4 Reference Design

Gowin_EMPU_M3 provides DDR3 reference design in ARM Keil MDK (V5.26 and above0 and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

13 SPI-Flash

13.1 Features

Gowin_EMPU_M3 contains one SPI-Flash module accessed by AHB:

- SPI-Flash download is at AHB bus interface
- SPI-Flash Memory read, write, and erase functions are at APB bus interface
- The default is Gowin on-board Winbond W25Q64

13.2 Register Description

The definition of SPI-Flash registers is shown in Table 14-1.

Table 14-1 Definition of SPI-Flash Registers

Name	Address Offset	Type	Width	Initial Value	Description
IDREV	0x00	RO	32	0x020 02000	ID and revision register [31:8] ID number [7:4] Major revision number [3:0] Minor revision number
RESERVED0[3]	0x04-0x0C	-	-	-	Reserved
TRANSFMT	0x10	RW	32	0x000 20780	SPI transfer format register [31:18] Reserved [17:16] Address length in bytes 00 = 1 byte 01 = 2 bytes 10 = 3 bytes 11 = 4 bytes [15:13] Reserved [12:8] Data length [7] Enable data merge mode [6:5] Reserved [4] Bi-directional MOSI in single mode 0 = MOSI is uni-directional signal

Name	Address Offset	Type	Width	Initial Value	Description
					1 = MOSI is bi-directional signal [3] Transfer data with the lease significant bit first 0 = Most significant bit first 1 = Least significant bit first [2] SPI master/slave mode selection 0 = Master mode 1 = Slave mode [1] SPI clock polarity 0 = SCLK is LOW in the idle states 1 = SCLK is HIGH in the idle states [0] SPI clock phase 0 = Sampling data at odd SCLK edges 1 = Sampling data at even SCLK edges
DIRECTIO	0x14	RW	32	0x0	SPI direct IO control register [31:25] Reserved [24] Enable direct IO 0 = Disable 1 = Enable [23:22] Reserved [21] Output enable for SPI-Flash hold signal [20] Output enable for SPI-Flash write protect signal [19] Output enable for the SPI MISO signal [18] Output enable for the SPI MOSI signal [17] Output enable for SPI SCLK signal [16] Output enable for SPI CS signal [15:14] Reserved [13] Output value for SPI-Flash hold signal [12] Output value for SPI-Flash write protect signal [11] Output value for SPI MISO signal [10] Output value for SPI MOSI signal [9] Output value for SPI SCLK signal [8] Output value for SPI CS signal [7:6] Reserved

Name	Address Offset	Type	Width	Initial Value	Description
					[5] Status of SPI-Flash hold signal [4] Status of SPI-Flash write protect signal [3] Status of SPI MISO signal [2] Status of SPI MOSI signal [1] Status of SPI SCLK signal [0] Status of SPI CS signal
RESERVED1[2]	0x18-0x1C	-	-	-	Reserved
TRANSCTRL	0x20	RW	32	0x0	SPI transfer control register [31] Reserved [30] SPI command phase enable 0 = Disable the command phase 1 = Enable the command phase (Master mode only) [29] SPI address phase enable 0 = Disable the address phase 1 = Enable the address phase (Master mode only) [28] SPI address phase format 0 = Address phase is single mode 1 = The format of the address phase is the same as the DualQuad data phase (Master mode only) [27:24] Transfer mode 0000 = Write and read at the same time 0001 = Write only 0010 = Read only 0011 = Write, Read 0100 = Read, Write 0101 = Write, Dummy, Read 0110 = Read, Dummy, Write 0111 = None data 1000 = Dummy, Write 1001 = Dummy, Read 1010~1111 = Reserved [23:22] SPI data phase format 00 = Single mode 01 = Dual I/O mode 10 = Quad I/O mode 11 = Reserved [21] Append and one-byte special token following the address phase for SPI read transfers

Name	Address Offset	Type	Width	Initial Value	Description
					[20:12] Transfer count for write data [11] The value of the one-byte special token following the address phase for SPI read transfers 0 = token value is 0x00 1 = token value is 0x69 [10:9] Dummy data count [8:0] Transfer count for read data
CMD	0x24	RW	32	0x0	SPI command register [31:8] Reserved [7:0] SPI command
ADDR	0x28	RW	32	0x0	SPI address register [31:0] SPI address (Master mode only)
DATA	0x2C	RW	32	0x0	SPI data register [31:0] Data to transmit or the received data
CTRL	0x30	RW	32	0x0	SPI controller register [31:21] Reserved [20:16] Transmit FIFO threshold [15:13] Reserved [12:8] Receive FIFO threshold [7:5] Reserved [4] TX DMA enable [3] RX DMA enable [2] Transmit FIFO reset [1] Receive FIFO reset [0] SPI reset
STATUS	0x34	RO	32	0x0	SPI status register [31:24] Reserved [23] Transmit FIFO full flag [22] Transmit FIFO empty flag [21] Reserved [20:16] Number of valid entries int the transmit FIFO [15] Receive FIFO full flag [14] Receive FIFO empty flag [13] Reserved [12:8] Number of valid entries in the receive FIFO [7:1] Reserved [0] SPI register programming is in progress
INTREN	0x38	RW	32	0x0	SPI interrupt enable register

Name	Address Offset	Type	Width	Initial Value	Description
					[31:6] Reserved [5] Enable the slave command interrupt [4] Enable the end of SPI transfer interrupt [3] Enable the SPI transmit FIFO threshold interrupt [2] Enable the SPI receive FIFO threshold interrupt [1] Enable SPI transmit FIFO underrun interrupt (Slave mode only) [0] Enable SPI receive FIFO overrun interrupt (Slave mode only)
INTRST	0x3C	WO	32	0x0	SPI interrupt status register [31:6] Reserved [5] Slave command interrupt (Slave mode only) [4] End of SPI transfer interrupt [3] TX FIFO threshold interrupt [2] RX FIFO threshold interrupt [1] TX FIFO underrun interrupt (Slave mode only) [0] RX FIFO overrun interrupt (Slave mode only)
TIMING	0x40	RW	32	0x0	SPI interface timing register [31:14] Reserved [13:12] The minimum time between the edges of SPI CS and the edges of SCLK [11:8] The minimum time the SPI CS should stay HIGH [7:0] The clock frequency ratio between the clock source and SPI interface SCLK
RESERVED2[3]	0x44-0x4c	-	-	-	Reserved
MEMCTRL	0x50	RW	32	0x0	SPI memory access control register [31:9] Reserved [8] This bit is set when “MEMCTRL” / ”TIMING” is written [7:4] Reserved [3:0] Selects the SPI command
RESERVED3[3]	0x54-0x5C	-	-	-	Reserved
SLVST	0x60	RW	32	0x0	SPI slave status register

Name	Address Offset	Type	Width	Initial Value	Description
					[31:19] Reserved [18] Data underrun occurs in the last transaction [17] Data overrun occurs in the last transaction [16] SPI is ready for data transaction [15:0] User defined status flags
SLVDATACNT	0x64	RO	32	0x0	SPI slave data count register [31:25] Reserved [24:16] Slave transmitted data count [15:9] Reserved [8:0] Slave received data count
RESERVED4[5]	0x68-0x78	-	-	-	Reserved
CONFIG	0x7C	RO	32	0x0	Configuration register [31:15] Reserved [14] Support for SPI slave mode [13] Reserved [12] Support for memory-mapped access through AHB bus [11] Support for direct SPI IO [10] Reserved [9] Support for Quad I/O SPI [9] Support for Dual I/O SPI [7:6] Reserved [5:4] Depth of TX FIFO 00 = 2 words 01 = 4 words 10 = 8 words 11 = 16 words [3:2] Reserved [1:0] Depth of RX FIFO 00 = 2 words 01 = 4 words 10 = 8 words 11 = 16 words

13.3 Usage of Drivers

The usage of SPI-Flash drivers is shown in Table 14-2.

Table 14-2 Usage of SPI-Flash Drivers

Name	Description
spi_flash_init	Initialize SPI-Flash
change_mode_spi_flash	Switch SPI-Flash mode between download and read, write, erase memory
spi_flash_read	Read data from SPI-Flash
spi_flash_write	Write data into SPI-Flash
spi_flash_write_read	Write data into SPI-Flash and read data from SPI-Flash once time
spi_flash_page_program	Write data into SPI-Flash with pages
spi_flash_sector_erase	Erase SPI-Flash with sector

13.4 Reference Design

Gowin_EMPU_M3 provides SPI-Flash reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\peripherals_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_peripherals_app

14 Embedded Real-time Operating System

Gowin_EMPU_M3 supports the embedded operating system of uC/OS-III and FreeRTOS.

14.1 uC/OS-III

14.1.1 Features

- The uC/OS-III is an extensible, romable, and preemptive real-time core. There is no limit to the number of managed tasks.
- uC/OS-III is the third generation core. It provides a real-time core functions, including resource management, synchronization, and inter-task communication, etc.
- UC/os-iii provides many features that other real-time cores do not have. For example, it can measure operating performance during runtime and send signals or messages to tasks directly. Tasks can also wait for multiple semaphores and message queues simultaneously.
- Gowin_EMPU_M3 offers uC/OS-III reference designs.
- uC/OS-III source code is available at Micrium website: http://www.micrium.com.

14.1.2 Operating System Version

Gowin_EMPU_M3 reference design uses uC/ os-iii V3.03.00 version.

14.1.3 Operating System Configuration

- Users can modify UCOSIII_CONFIG\os_cfg.h and os_cfg_app.h to configure uC/OS-III.
- Users can modify UCOS_BSP\bsp.c and bsp.h to support the development board used.

14.1.4 Reference Design

Gowin_EMPU_M3 provides uC/OS-III reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\ucos_iii_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_ucos_iii_app

14.2 FreeRTOS

14.2.1 Features

- FreeRTOS is a lightweight real-time operating system.
- FreeRTOS is a lightweight operating system. It offers functions of task management, time management, semaphore, message queue, memory management, recording, software timer, coroutines, etc. It can basically meet the needs of small systems.；
- FreeRTOS is a free operating system. It has features of open source, portability, reducibility, and flexible scheduling policy.；
- Gowin_EMPU_M3 offers FreeRTOS reference designs.
- FreeRTOS source code is available at FreeRTOS website: http://www.FreeRTOS.org.

14.2.2 Operating System Version

Gowin_EMPU_M3 reference designs use FreeRTOS (V10.2.1).

14.2.3 Operating System Configuration

Users can modify include\FreeRTOSConfig.h to configure FreeRTOS.

14.2.4 Reference Design

Gowin_EMPU_M3 provides FreeRTOS reference design in ARM Keil MDK (V5.26 and above) and GOWIN MCU Designer (V1.1 and above).

- Gowin_EMPU_M3\ref_design\MCU_RefDesign\Keil_RefDesign\freertos_app
- Gowin_EMPU_M3\ref_design\MCU_RefDesign\GMD_RefDesign\sm3_freertos_app


GOWIN
PROGRAMMING FOR THE FUTURE