Gowin_EMPU_M1

软件编程参考手册

IPUG533-1.1,2019-07-18

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和／或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改文档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	说明
2019/02/19	1.0	初始版本。
2019/07/18	1.1	MCU 软件编程设计支持扩展外部设备 CAN、Ethernet、SPI-Flash、RTC、DualTimer、TRNG、I2C、SPI、SD-Card。

目录

图目录

表目录

1 软件编程库

Gowin_EMPU_M1 提供软件编程库：

- src\c_lib

Gowin_EMPU_M1 提供两种软件编程方法：

- Cortex-M1 软件编程
- 嵌入式操作系统软件编程

1.1 Cortex-M1 软件编程

Gowin_EMPU_M1 软件编程库中提供的 Cortex-M1 软件编程方法，如表 1-1 所示。

表 1-1 Cortex-M1 软件编程

文件	描述
startup_GOWIN_M1.s	Cortex-M1 启动引导程序
core_cm1.h	Cortex-M1 内核定义
GOWIN_M1.h	中断向量表、寄存器和地址映射定义
system_GOWIN_M1.c	Cortex-M1 系统初始化和系统时钟定义
GOWIN_M1_flash.ld	GNU 工具链中 Flash 链接脚本
GOWIN_M1_gpio.c	GPIO 驱动函数定义
GOWIN_M1_can.c	CAN 驱动函数定义
GOWIN_M1_ethernet.c	Ethernet 驱动函数定义
GOWIN_M1_timer.c	Timer 驱动函数定义
GOWIN_M1_wdog.c	Watch Dog 驱动函数定义
GOWIN_M1_uart.c	UART 驱动函数定义
GOWIN_M1_rtc.c	RTC 驱动函数定义
GOWIN_M1_trng.c	TRNG 驱动函数定义
GOWIN_M1_dualtimer.c	DualTimer 驱动函数定义
GOWIN_M1_i2c.c	I2C 驱动函数定义
GOWIN_M1_spi.c	SPI 驱动函数定义
GOWIN_M1_sdcard.c	SD-Card 驱动函数定义
GOWIN_M1_misc.c	中断优先级管理和 SysTick
GOWIN_M1_it.c	中断处理函数定义
retarget.c	printf 函数重定向
malloc.c	动态内存管理

1.2 嵌入式操作系统软件编程

Gowin_EMPU_M1 支持以下两种操作系统软件编程：

- uC/OS-III
- FreeRTOS

2 存储系统

2.1 标准外设内存映射

Gowin_EMPU_M1 标准外设内存映射地址如表 2-1 所示。

表 2-1 Gowin_EMPU_M1 标准外设内存映射

标准外设	类型	地址映射	描述
ITCM		0x00000000	1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB
DTCM		0x20000000	1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB
TIMER0	TIMER_TypeDef	0x50000000	定时器 0
TIMER1	TIMER_TypeDef	0x50001000	定时器 1
UART0	UART_TypeDef	0x50004000	通用异步收发器 0
UART1	UART_TypeDef	0x50005000	通用异步收发器 1
Watch Dog	WDOG_TypeDef	0x50008000	看门狗
RTC	RTC_RefDef	0x50006000	实时时钟
TRNG	TRNG_RefDef	0x5000F000	真随机数发生器
DualTimer	DUALTIMER_RegDef	0x50002000	双重定时器
I2C	I2C_TypeDef	0x5000A000	内部集成电路
SPI	SPI_TypeDef	0x5000B000	串行外设接口
SD-Card	SDCard_TypeDef	0x50009000	安全数字存储卡
GPIO0	GPIO_TypeDef	0x40000000	通用输入输出端口
CAN	CAN_RegDef	0x45000000	控制器局域网络
Ethernet	ETH_RegDef	0x46000000	以太网
APB2 Extension		0xB0000000	APB 扩展接口
AHB2 Extension		0xA0000000	AHB 扩展接口

2.2 内核系统内存映射

Cortex-M1 内核系统内存映射如表 2-2 所示。

表 2-2 Cortex-M1 内核系统内存映射

系统控制	类型	地址映射	描述
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

3 中断处理

嵌套向量中断控制器包括以下特征：

- 提供最多 32 个用户可用的中断处理信号，用户可配置 1 或 8 或 16 或 32 个外部中断处理信号
- 支持 0-3 级可编程优先级

Cortex-M1 中断控制器如表 3-1 所示。

表 3-1 Cortex-M1 中断控制器

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	0		Reserved
0x00000014	0		Reserved
0x00000018	0		Reserved
0x0000001C	0		Reserved
0x00000020	0		Reserved
0x00000024	0		Reserved
0x00000028	0		Reserved
0x0000002C	SVC_Handler	-5	SVCall Handler
0x00000030	0		Reserved
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 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	CAN_Handler	8	16+ 8: CAN Handler
0x00000064	ETH_Handler	9	16+ 9: ETH Handler
0x00000068	Interrupt10_Handler	10	16+10: Interrupt 10 Handler

Address	Interrupt	Number	Description
0x0000006C	DTimer_Handler	11	16+11: DualTimer Handler
0x00000070	TRNG_Handler	12	16+12: TRNG 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	Interrupt16_Handler	16	16+16: Interrupt 16 Handler
0x00000084	Interrupt17_Handler	17	16+17: Interrupt 17 Handler
0x00000088	Interrupt18_Handler	18	16+18: Interrupt 18 Handler
0x0000008C	Interrupt19_Handler	19	16+19: Interrupt 19 Handler
0x00000090	Interrupt20_Handler	20	16+20: Interrupt 20 Handler
0x00000094	Interrupt21_Handler	21	16+21: Interrupt 21 Handler
0x00000098	Interrupt22_Handler	22	16+22: Interrupt 22 Handler
0x0000009C	Interrupt23_Handler	23	16+23: Interrupt 23 Handler
0x000000A0	Interrupt24_Handler	24	16+24: Interrupt 24 Handler
0x000000A4	Interrupt25_Handler	25	16+25: Interrupt 25 Handler r
0x000000A8	Interrupt26_Handler	26	16+26: Interrupt 26 Handler
0x000000AC	Interrupt27_Handler	27	16+27: Interrupt 27 Handler
0x000000B0	Interrupt28_Handler	28	16+28: Interrupt 28 Handler r
0x000000B4	Interrupt29_Handler	29	16+29: Interrupt 29 Handler
0x000000B8	Interrupt30_Handler	30	16+30: Interrupt 30 Handler
0x000000BC	Interrupt31_Handler	31	16+31: Interrupt 31 Handler

4 通用异步收发器

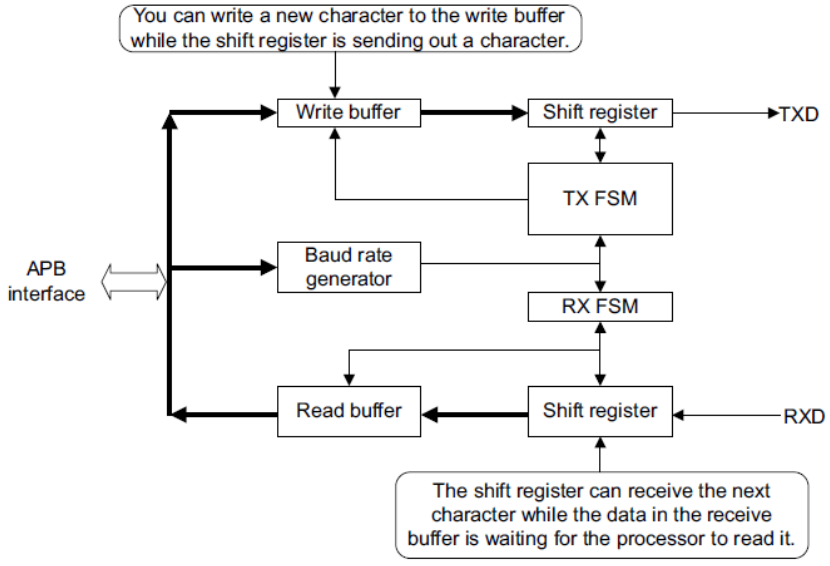
4.1 特征

Gowin_EMPU_M1 包含 2 个通过 APB 总线访问的通用异步收发器 UART：

- 最大波特率为 921.6Kbit/s
- 无奇偶校验位
- 8 位数据位
- 1 位停止位

UART Buffering 如图 4-1 所示。

图 4-1 UART Buffering



UART 支持高速测试模式 HSTM（High Speed Test Mode），当寄存器 CTRL[6]设置为 1 时，串行数据每个周期传输 1 位，可以在很短时间内传输文本信息。

用户在使能 UART 时，必须设置波特率分频寄存器，例如，如果 APB1 总线频率运行在 12MHz，需要波特率为 9600，则可以设置波特率分频寄存

器为 12000000/9600=1250。

4.2 寄存器定义

UART 寄存器定义如表 4-1 所示。

表 4-1 UART 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
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 初始化定义

UART 初始化定义如表 4-2 所示。

表 4-2 UART 初始化定义

名称	类型	数值	描述
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 函数库使用方法

UART 函数库使用方法如表 4-3 所示。

表 4-3 UART 函数库使用方法

名称	描述
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

名称	描述
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 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 UART 参考设计：

- MCU_RefDesign\Keil_RefDesign\uart
- MCU_RefDesign\GNU_RefDesign\cm1_uart

5 定时器

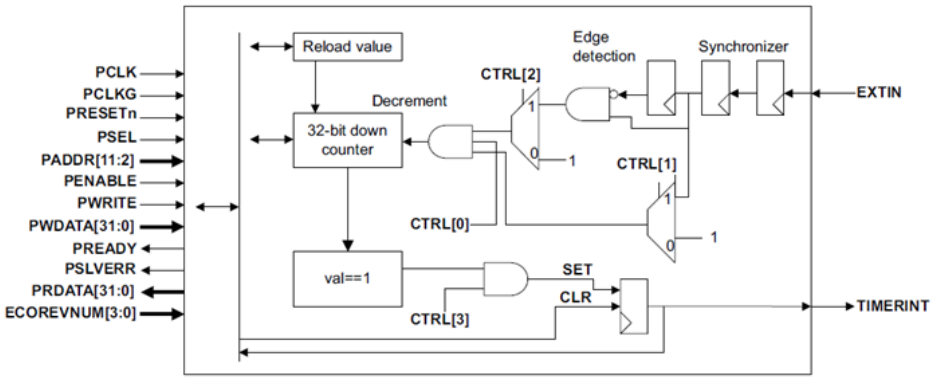
5.1 特征

Gowin_EMPU_M1 包含 2 个通过 APB 总线访问的同步标准定时器：

- 32 位计数器
- 可以产生中断请求信号
- 可以使用外部输入信号 EXTIN 使能时钟
- TIMER0: EXTIN 连接 GPIO[1]
- TIMER1: EXTIN 连接 GPIO[6]

Timer 结构如图 5-1 所示。

图 5-1 Timer



5.2 寄存器定义

Timer 寄存器定义如表 5-1 所示。

表 5-1 Timer 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
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	0x00000000	[31:0] Current value

寄存器名称	地址偏移	类型	宽度	初始值	描述
RELOAD	0x008	RW	32	0x00000000	[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 初始化定义

Timer 初始化定义如表 5-2 所示。

表 5-2 Timer 初始化结构

名称	类型	数值	描述
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 函数库使用方法

Timer 函数库使用方法如表 5-3 所示。

表 5-3 Timer 函数库使用方法

名称	描述
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
TIMER_DisableIRQ	Disable TIMERx interrupt request

5.5 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 Timer 参考设计：

- MCU_RefDesign\Keil_RefDesign\timer
- MCU_RefDesign\GNU_RefDesign\cm1_timer

6 看门狗

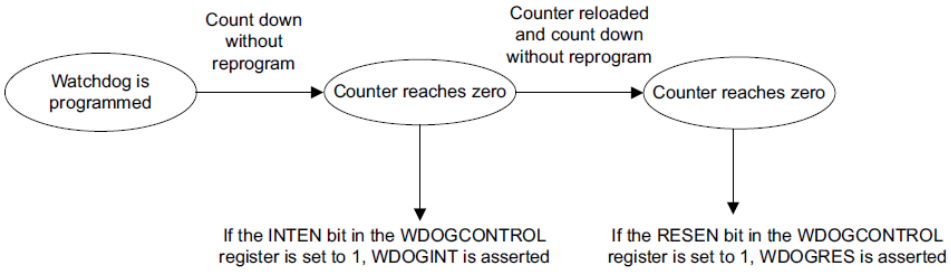
6.1 特征

Gowin_EMPU_M1 包含 1 个通过 APB 总线访问的看门狗 Watch Dog：

- 基于由 LOAD 寄存器初始化的 32 位逐减计数器
- 产生中断请求
- 当时钟使能，由 WDOGCLK 信号上升沿触发计数器递减
- 监视中断，当计数器递减到 0 时，产生复位请求，计数器停止
- 响应软件崩溃引起的复位，提供软件恢复方法

Watch Dog 操作如图 6-1 所示。

图 6-1 Watch Dog Operation



6.2 寄存器定义

Watch Dog 寄存器定义如表 6-1 所示。

表 6-1 Watch Dog 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
LOAD	0x00	RW	32	0xFFFFFFFF	The value from which the counter is to decrement
VALUE	0x04	RO	32	0xFFFFFFFF	The current value of the decrementing counter
CTRL	0x08	RW	2	0x0	[1] Enable reset output [0] Enable the interrupt
INTCLR	0x0C	WO			Clear the watchdog interrupt and reloads the counter

寄存器名称	地址偏移	类型	宽度	初始值	描述
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	0x00000000	[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 初始化定义

Watch Dog 初始化定义如表 6-2 所示。

表 6-2 Watch Dog 初始化定义

名称	类型	数值	描述
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 函数库使用方法

Watch Dog 函数库使用方法如表 6-3 所示。

表 6-3 Watch Dog 函数库使用方法

名称	描述
WDOG_Init	Initializes WatchDog
WDOG_RestartCounter	Restart watchdog counter
WDOG_GetCounterValue	Returns counter value
WDOG_SetResetEnable	Sets reset enable
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 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 Watch Dog 参考设计：

- MCU_RefDesign\Keil_RefDesign\watchdog
- MCU_RefDesign\GNU_RefDesign\cm1_watchdog

7 通用输入输出

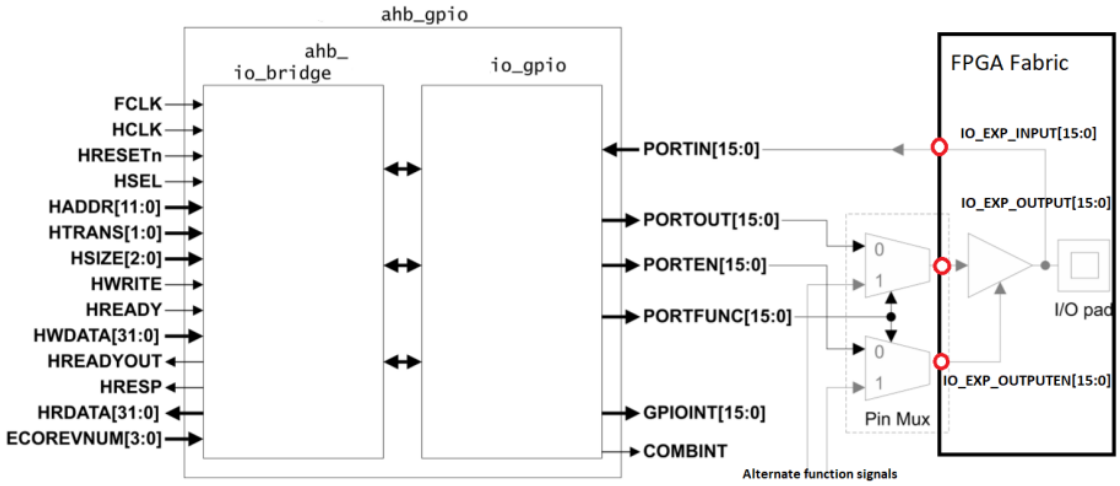
7.1 特征

Gowin_EMPU_M1 包含 1 个通过 AHB 总线访问的 16 位输入输出接口的 GPIO 模块：

- 与 FPGA Fabric 连接
- 支持位掩码
- 管脚复用功能

GPIO 结构如图 7-1 所示。

图 7-1 GPIO Block



7.2 寄存器定义

GPIO 寄存器定义如表 7-1 所示。

表 7-1 GPIO 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
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

寄存器名称	地址偏移	类型	宽度	初始值	描述
					synchronization logic with delya of two cycle
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 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

寄存器名称	地址偏移	类型	宽度	初始值	描述
					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 effectx 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 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 初始化定义

GPIO 初始化定义如表 7-2 所示。

表 7-2 GPIO 初始化定义

名称	类型	数值	描述
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	16 bits GPIO Pins interrupt

名称	类型	数值	描述
		GPIO_Int_Rising_Edge	

7.4 函数库使用方法

GPIO 函数库使用方法如表 7-3 所示。

表 7-3 GPIO 函数库使用方法

名称	描述
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 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 GPIO 参考设计：

- MCU_RefDesign\Keil_RefDesign\led
- MCU_RefDesign\GNU_RefDesign\cm1_led

8 内部集成电路

8.1 特征

Gowin_EMPU_M1 包含一个通过 APB 总线访问的内部集成电路 I2C Master 模块：

- APB 总线接口
- 符合业界标准的 I2C 总线协议
- 总线仲裁及仲裁丢失检测
- 总线忙状态检测
- 产生中断标志
- 产生起始、终止、重复起始和应答信息
- 支持起始、终止和重复起始检测
- 支持 7 位寻址模式

8.2 寄存器定义

I2C 寄存器定义如表 8-1 所示。

表 8-1 I2C 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
PRER	0x00	RW	32	0x0000FFFF	Clock prescale register [31:15] Reserved [15:0] Prescale value = sys_clk/(5*SCL)-1
CTR	0x04	RW	32	0x00000000	[31:8] Reserved [7] Enable I2C function [6] Enable I2C interrupt [5:0] Reserved
TXR	0x08	WO	32	0x00000000	[31:8] Reserved [7:1] Next transmission data [0] Data direction
RXR	0x0C	RO	32	0x00000000	[31:8] Reserved [7:0] Last received data
CR	0x010	WO	32	0x00000000	[31:8] Reserved [7] STA, Start transmission status [6] STO, Over transmission status

寄存器名称	地址偏移	类型	宽度	初始值	描述
					[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	0x00000000	[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 函数库使用方法

I2C 函数库使用方法如表 8-2 所示。

表 8-2 I2C 函数库使用方法

名称	描述
I2C_Init	I2C Initialization
I2C_SendByte	Send a byte to I2C bus
I2C_SendBytes	Send multiple bytes to I2C bus
I2C_ReceiveByte	Read a byte from I2C bus
I2C_ReadBytes	Read multiple bytes from I2C bus
I2C_Rate_Set	Set I2C traffic rate
I2C_Enable	Enable I2C bus
I2C_UnEnable	Disable I2C bus

8.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 I2C 参考设计：

- MCU_RefDesign\Keil_RefDesign\i2c
- MCU_RefDesign\GNU_RefDesign\cm1_i2c

9 串行外设接口

9.1 特征

Gowin_EMPU_M1 包含一个通过 APB 总线访问的串行外设接口 SPI Master 模块：

- APB 总线接口
- 全双工同步串行数据传输
- 支持 Master 工作模式
- 支持可配置的时钟极性和相位
- SPI 产生的串行时钟频率可配置
- 数据接收寄存器和数据发送寄存器 8 位宽

9.2 寄存器定义

SPI 寄存器定义如表 9-1 所示。

表 9-1 SPI 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
RDATA	0x00	RO	32	0x0000 0000	Read data register [31:8] Reserved [7:0] Read data
WDATA	0x04	WO	32	0x0000 0000	Write data register [31:8] Reserved [7:0] Write data
STATUS	0x08	RW	32	0x0000 0000	[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	0x0000 0000	[31:1] Reserved [0] Select and enable slave
CTRL	0x10	RW	32	0x0000 0000	[31:5] Reserved [4:3] Clock selected, CLK_I / 2/4/6/8 [2] Clock polarity

寄存器名称	地址偏移	类型	宽度	初始值	描述
					[1] Clock phase [0] Direction, 1 is MSB first

9.3 初始化定义

SPI 初始化定义如表 9-2 所示。

表 9-2 SPI 初始化定义

名称	类型	数值	描述
DIRECTION	FunctionalState	ENABLE/DISABLE	MSB/LSB first transmission
PHASE	FunctionalState	ENABLE/DISABLE	Posedge/Negedge transmit data
POLARITY	FunctionalState	ENABLE/DISABLE	Initialize ploarity to one/zero
CLKSEL	uint32_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 函数库使用方法

SPI 函数库使用方法如表 9-3 所示。

表 9-3 SPI 函数库使用方法

名称	描述
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
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	Clear receive overrun error status
SPI_ClrErrStatus	Clears error status
SPI_WriteData	Writes data
SPI_ReadData	Reads data
SPI_Select_Slave	Select slave

9.5 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 SPI 参考设计：

- MCU_RefDesign\Keil_RefDesign\spi

- MCU_RefDesign\GNU_RefDesign\cm1_spi

10 实时时钟

10.1 特征

Gowin_EMPU_M1 包含 1 个通过 APB 总线访问的 32 位实时时钟 RTC 模块：

- APB 总线接口
- 32-bit 计数器
- 32-bit Match 寄存器
- 32-bit 比较器

MCU 通过 APB 总线接口与 RTC 读写数据、控制和状态信息。在连续输入时钟 CLK1HZ 上升沿时，32-bit 计数器递增。

此计数器是不同步计数器，不可重载。在系统复位时，此计数器从 1 开始计数，递增到最大值 0xFFFFFFFF，然后回绕到 0 开始继续递增。

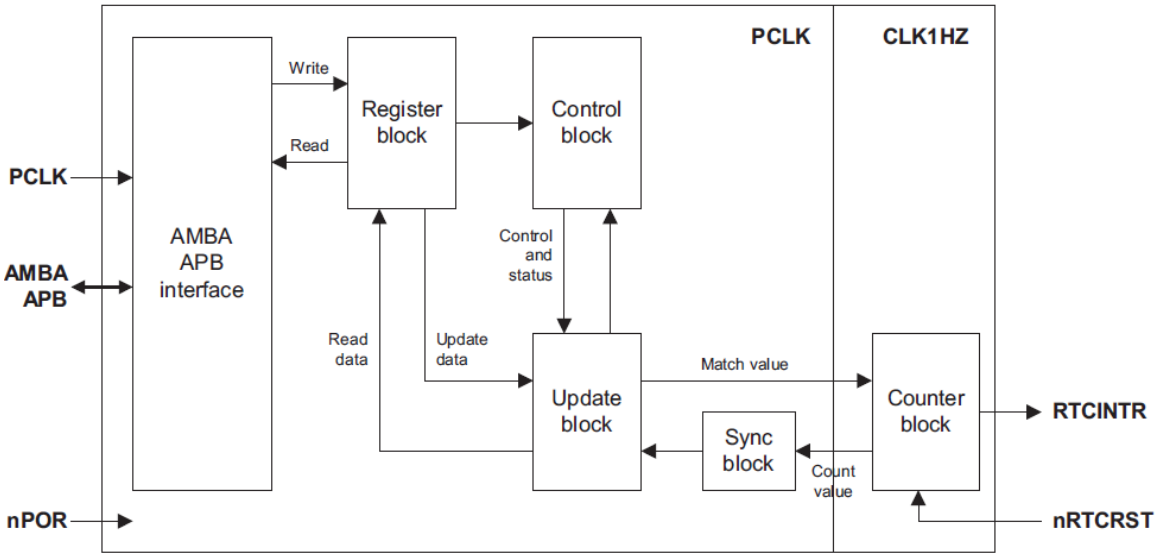
通过写 Load 寄存器 RTC_LOAD_VALUE，实现 RTC 加载或更新。

通过读 Data 寄存器 RTC_CURRENT_DATA，获取 RTC 当前时钟。

通过写 RTC_MATCH_VALUE 寄存器，编程 Match 寄存器。

RTC 结构如图 10-1 所示。

图 10-1 RTC Block



10.2 寄存器定义

RTC 寄存器定义如表 10-1 所示。

表 10-1 RTC 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
RTC_CURRENT_DATA	0x000	RO	32	0x00000000	Data Register [31:0] Current value
RTC_MATCH_VALUE	0x004	RW	32	0x00000000	Match Register If current value equals match register’s value, generate interrupt [31:0] Match data
RTC_LOAD_VALUE	0x008	RW	32	0x00000000	Load Register Initialized value, start counter based on this value [31:0] Load data
RTC_CTROLLER_REG	0x00C	RW	32	0x00000000	Control Register Start RTC counter [31:1] Reserved [0] Start RTC counter
RTC_IMSC	0x010	RW	32	0x00000000	Interrupt mask set and clear register Enable or disable interrupt [31:1] Reserved [0] Enable interrupt
RTC_RIS	0x014	RO	32	0x00000000	Raw interrupt status register Get current raw unmasked interrupt status [31:1] Reserved [0] Current raw unmasked interrupt status
RTC_MIS	0x018	RO	32	0x00000000	Masked interrupt status register Get current masked interrupt status [31:1] Reserved [0] Current masked interrupt status
RTC_INTR_CLEAR	0x01C	WO	32	0x00000000	Interrupt clear register Clear current interrupt [31:1] Reserved

寄存器名称	地址偏移	类型	宽度	初始值	描述
					[0] Clear current interrupt

10.3 函数库使用方法

RTC 函数库使用方法如表 10-2 所示。

表 10-2 RTC 函数库使用方法

名称	描述
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 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 RTC 参考设计：

- MCU_RefDesign\Keil_RefDesign\rtc
- MCU_RefDesign\GNU_RefDesign\cm1_rtc

11 真随机数发生器

11.1 特征

Gowin_EMPU_M1 包含 1 个通过 APB 总线访问的 32 位真随机数发生器 TRNG 模块：

- 数字逻辑产生和采集一个真正的随机数比特流
- 包含一个基于数字逆变器链的内部熵源
- 如果 MCU 内核 200MHz 运行，产生一个 10K bits/s 的熵
- APB 总线接口

11.2 寄存器定义

TRNG 寄存器定义如表 11-1 所示。

表 11-1 TRNG 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
RESERVE1	0x000-0x0FC	-	-	-	Reserved
RNG_IMR	0x100	RW	32	0x00000 00F	Interrupt mask register [31:4] Reserved [3] Mask the Von Neumann error [2] Mask the CRNGT error [1] Mask the Autocorrelation error [0] Mask when the TRNG has collected 192 bits
RNG_ISR	0x104	RO	32	0x00000 000	Interrupt status register [31:4] Reserved [3] A Von Neumann error [2] A Continuos Random Number Generation Testing (CRNGT) error [1] The Autocorrelation test failed four timers in a row [0] Set to 1 When 192 bits have been collected, and EHR_DATA[0-5] registers are ready to be read
RNG_ICR	0x108	WO	32	0x00000 000	Interrupt clear register [31:4] Reserved [3] Clear a Von Nenumann error

寄存器名称	地址偏移	类型	宽度	初始值	描述
					[2] Clear a CRNGT error [1] Software cannot clear this bit, only a TRNG reset can clear this bit [0] Set to 1 after EHR_DATA[0-5] have been read
TRNG_CONFIG	0x10C	RW	32	0x00000000	Configuration register [31:2] Reserved [1:0] Selects the number of inverters: 0b00 = Selects the shortest inverter chain length 0b01 = Selects the short inverter chain length 0b10 = Selects the long inverter chain length 0b11 = Selects the longest inverter chain length
TRNG_VALID	0x110	RO	32	0x00000000	Valid register [31:1] Reserved [0] TRNG is complete, data can be read from EHR_DATA[0-5]
EHR_DATA0 EHR_DATA1 EHR_DATA2 EHR_DATA3 EHR_DATA4 EHR_DATA5	0x114—x128	RO	32	0x00000000	Entropy holding register data register Return 32 bits from the 192-bit EHR DATA0 returns bits[31:0] DATA1 returns bits[63:32] DATA2 returns bits[95:64] DATA3 returns bits[127:96] DATA4 returns bits[159:128] DATA5 returns bits[191:160]
RND_SOURCE_ENABLE	0x12C	RW	32	0x00000000	Random source enable register [31:1] Reserved [0] 1 = enable entropy source; 0 = disable entropy source
SAMPLE_CNT1	0x130	RW	32	0x0000FFFF	Sample count register [31:0] Sets the number of rng_clk cycles
AUTOCORR_STATISTIC	0x134	RW	32	0x00000000	Autocorrelation register [31:22] Reserved [21:14] Count each time an autocorrelation test fails [13:0] Count each time an autocorrelation test starts
TRNG_DEBUG_CONTROL	0x138	RO	32	0x00000000	Debug control register [31:4] Reserved [3] The autocorrelation test is bypassed [2] The CRNGT test is bypassed [1] The Von Neumann balancer is bypassed [0] Reserved
RESERVE2	0x13C	-	-	-	Reserved
TRNG_SW_RESET	0x140	WO	32	0x00000000	Reset register [31:1] Reserved [0] Writing 1 to this register causes an internal TRNG reset
RESERVE3	0x144-0x1B4	-	-	-	Reserved
TRNG_BUSY	0x1B8	RO	32	0x00000000	Busy register [31:1] Reserved [0] Relects the status of rng_busy signal
RST_BIT_COUNT	0x1BC	WO	32	0x00000000	Reset bits counter register [31:1] Reserved [0] Write any value to this bit resets the bits counter

寄存器名称	地址偏移	类型	宽度	初始值	描述
					and TRNG valid registers
RESERVE4	0x1C0-0x1DC	-	-	-	Reserved
RNG_BIST_CNTR0	0x1E0-0x1E8	RO	32	0x00000000	BIST counter registers Return the collected BIST results [31:22] Reserved [21:0] Returns the results of the TRNG BIST counter
RNG_BIST_CNTR1					
RNG_BIST_CNTR2					

11.3 函数库使用方法

TRNG 函数库使用方法如表 11-2 所示。

表 11-2 TRNG 函数库使用方法

名称	描述
Init_TRNG	Initialized TRNG
Set_Interrupt_Mask	Set interrupt mask
Get_Int_State	Get interrupt status
Clear_Int	Clear interrupt
Set_Config	Set config register
Get_EHR_Data	Get Entropy holding data
Set_Random_Source_Enable	Set random source enable register
Clr_Random_Source_Enable	Clear random source enable register
Set_Sample_Count	Set sample count register
Trng_SW_Reset	Reset TRNG
Get_TRNG_State	Get TRNG state
Reset_Bit_Count	Reset bit count register
Get_BIT_Counter	Get bits count register
Set_Debug_Control	Set debug control register
Fail_Start_State_times	Get autocorrelation register
Clr_Fail_Start_State_register	Clear autocorrelation register

11.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 TRNG 参考设计：

- MCU_RefDesign\Keil_RefDesign\trng
- MCU_RefDesign\GNU_RefDesign\cm1_trng

12 双重定时器

12.1 特征

Gowin_EMPU_M1 包含 1 个通过 APB 总线访问的 32 位和 16 位双重定时器 DualTimer 模块：

- APB 总线接口
- 包含两个可编程的 32-bit 或 16-bit 倒数器
- 倒数器倒数到 0 时能够产生中断

12.2 寄存器定义

DualTimer 寄存器定义如表 12-1 所示。

表 12-1 DualTimer 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
TIMER1LOAD	0x00	RW	32	0x00000000	Timer1 load register [31:0] Timer1 load value
TIMER1VALUE	0x04	RO	32	0xFFFFFFFF	Timer1 current value register [31:0] Timer1 current value
TIMER1CONTROL	0x08	RW	32	0x00000020	Timer1 control register [31:8] Reserved [7] Timer enable [6] Timer mode [5] Interrupt enable [4] Reserved [3:2] Timer prescale 00 = clock is divided by 1 01 = clock is divided by 16 10 = clock is divided by 256 11 Undefined [1] Timer size 0 = 16-bit counter, default 1 = 32-bit counter [0] One-shot count 0 = wrapping mode, default 1 = on-shot mode

寄存器名称	地址偏移	类型	宽度	初始值	描述
TIMER1INTCLR	0x0C	WO	-	-	Timer1 interrupt clear register Any write clears the interrupt output of the counter
TIMER1RIS	0x10	RO	32	0x00000000	Timer1 raw interrupt status register [31:1] Reserved [0] Raw interrupt status from the counter
TIMER1MIS	0x14	RO	32	0x00000000	Timer1 interrupt status register [31:1] Reserved [0] Enable interrupt status from the counter
TIMER1BGLOAD	0x18	RW	32	0x00000000	Timer1 background load register [31:0] The value used to reload the counter
RESERVE1	-	-	-	-	Reserved
TIMER2LOAD	0x00	RW	32	0x00000000	Timer2 load register [31:0] Timer2 load value
TIMER2VALUE	0x04	RO	32	0xFFFFFFFF	Timer2 current value register [31:0] Timer2 current value
TIMER2CONTROL	0x08	RW	32	0x00000020	Timer2 control register [31:8] Reserved [7] Timer enable [6] Timer mode [5] Interrupt enable [4] Reserved [3:2] Timer prescale 00 = clock is divided by 1 01 = clock is divided by 16 10 = clock is divided by 256 11 Undefined [1] Timer size 0 = 16-bit counter, default 1 = 32-bit counter [0] One-shot count 0 = wrapping mode, default 1 = on-shot mode
TIMER2INTCLR	0x0C	WO	-	-	Timer2 interrupt clear register Any write clears the interrupt output of the counter
TIMER2RIS	0x10	RO	32	0x00000000	Timer2 raw interrupt status register [31:1] Reserved [0] Raw interrupt status from the counter
TIMER2MIS	0x14	RO	32	0x00000000	Timer2 interrupt status register [31:1] Reserved [0] Enable interrupt status from the counter
TIMER2BGLOAD	0x18	RW	32	0x00000000	Timer2 background load register [31:0] The value used to reload the counter

12.3 函数库使用方法

DualTimer 函数库使用方法如表 12-2 所示。

表 12-2 DualTimer 函数库使用方法

名称	描述
DUALTIMER1_Init	Initialized DualTimer1
DUALTIMER2_Init	Intiialized DualTimer2
Clear_DULATIMER_interrupt	Clear DualTimer interrupt

Dtimer_MODE_function	Set timer mode of DualTimer1 or DualTimer2
Dtimer_PRE_function	Set timer prescale of DualTimer1 or DualTimer2
INIT_NUM_load_function	Set load value of DualTimer1 or DualTimer2
ENABLE_interrupt_Dtimer_function	Enable interrupt of DualTimer1 or DualTimer2
TIMER_SIZE_function	Set timer size of DualTimer1 or DualTimer2
ENABLE_Dtimer_function	Enable DualTimer1 or DualTime2
Get_DULATIMER_interrupt_num	Get timer ID of DualTimer1 or DualTimer2

12.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 DualTimer 参考设计：

- MCU_RefDesign\Keil_RefDesign\dualtimer
- MCU_RefDesign\GNU_RefDesign\cm1_dualtimer

13 安全数字存储卡

13.1 特征

Gowin_EMPU_M1 包含 1 个通过 APB 总线访问的安全数字存储卡 SD-Card 模块：

- APB 总线接口
- 只读模式

13.2 寄存器定义

SD-Card 寄存器定义如表 13-1 所示。

表 13-1 SD-Card 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
TRANSDATA	0x000-0x1FF	RW	32	0x00000000	Data registers
BASEADDR	0x200	RW	32	0x00000000	Base address register
READEN	0x204	RW	32	0x00000000	Read enable register [31:1] Reserved [0] Enable read
INITIALISED	0x208	RW	32	0x00000000	Initialization register

13.3 函数库使用方法

SD-Card 函数库使用方法如表 13-2 所示。

表 13-2 SD-Card 函数库使用方法

名称	描述
sd_sector_read	Read a sector data
SD_ReadDisk	Read a package data
sdcard_read_package	Read whole sd-card data

13.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 SD-Card 参考设计：

- MCU_RefDesign\Keil_RefDesign\sdcard
- MCU_RefDesign\GNU_RefDesign\cm1_sdcard

14 控制器局域网络

14.1 特征

Gowin_EMPU_M1 包含 1 个通过 AHB 总线访问的控制器局域网络 CAN 模块：

- AHB 总线接口
- 符合 CAN2.0A 和 CAN2.0B 协议及 ISO 11898-1 标准
- CAN FD 协议
- 独立系统时钟和 CAN 总线时钟
- 灵活的共享缓存方案，实现最佳缓存区大小，以便在给定的应用程序中存储发送和接收消息
- 接收滤波器可配置为 1-16 个
- 可编程波特率预分频器

14.2 寄存器定义

CAN 寄存器定义如表 14-1 所示。

表 14-1 CAN 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
SRST	0x0000	RW	32	0x00000000	Software reset register [31:1] Reserved [0] control reset 1 = start hard reset 0 = cancel reset
CMD	0x0004	RW	32	0x00000000	Command register [31:1] Reserved [0] Enable 1 = working mode 0 = command mode
BRP	0x0008	RW	32	0x00000000	Baud rate prescalar register [31:8] Reserved [7:0] baud rate prescalar
BTN	0x000C	RW	32	0x00000000	Bit timing (nominal) register [28:24] sjw_nom

寄存器名称	地址偏移	类型	宽度	初始值	描述
					[13:8] phseg2_nom, PHASE_SEG2 window's width [5:0] phseg1_nom, PROP_SEG+PHASE_SEG1 window's width
BTD	0x0010	RW	32	0x00000000	Bit timing (data) register [26:24] sjw_d [11:8] phseg2_d, PHASE_SEG2 window's depth [3:0] phseg1_d, PROP_SEG+PHASE_SEG1 window's depth
RSVD0	0x001C	-	-	-	Reserved
IS	0x0020	RO	32	0x00000000	Interrupt status register [31] Bus off status [27] TX message successfully [26] TX message retry [25] TX message failed [23] TX high-priority message successfully [22] TX high-priority message retry [21] TX high-priority message failed [8] Error status [5] TX high-priority fifo overflow [4] TX fifo overflow [1] RX fifo overflow [0] RX fifo valid
IE	0x0024	RW	32	0x00000000	Interrupt enable register [31] Enable us off [27] Enable TX message successfully [26] Enable TX message retry [25] Enable TX message failed [23] Enable TX high-priority message successfully [22] Enabe TX high-priority message retry [21] Enable TX high-priority message failed [8] Enable Error status [5] Enabe TX high-priority fifo overflow [4] Enable TX fifo overflow [1] Enable RX fifo overflow [0] Enable RX fifo valid
IC	0x0028	WO	32	0x00000000	Interrupt clear register [31] Clear bus off status [27] Clear TX message successfully status [26] Clear TX message retry status [25] Clear TX message failed status [23] Clear TX high-priority message successfully status [22] Clear TX high-priority message retry status [21] Clear TX high-priority message failed status [8] Clear Error status status [5] Clear TX high-priority fifo overflow status [4] Clear TX fifo overflow status [1] Clear RX fifo overflow status [0] Clear RX fifo valid status
RSVD1	0x002C	-	-	-	Reserved
CFG	0x0030	RW	32	0x00000000	Configuration register [4] Configure disprotexceponres

寄存器名称	地址偏移	类型	宽度	初始值	描述
					1 = ‘res’ is FORM-ERROR 0 = ‘res’ is exception [0] Configure isofd 1 = ISO FD mode 0 = non ISO FD mode
RSVD2	0x0034-0x003C	-	-	-	Reserved
RXBCFG	0x0040	RW	32	0x00000000	RX buffer/fifo configuration register [31:16] RX buffer’s ending offset [15:0] RX buffer’s start offset
TXBCFG	0x0044	RW	32	0x00000000	TX buffer/fifo configuration register [31:16] TX buffer’s ending offset [15:0] TX buffer’s start offset
TXHBCFG	0x0048	RW	32	0x00000000	TX high-priority/fifo configuration register [31:16] TX high-priority buffer’s ending offset [15:0] TX high-priority buffer’s start offset
RSVD3	0x004C	-	-	-	Reserved
TXBRETRY	0x0050	RW	32	0x00000000	TX buffer retry counter
TXHBRETRY	0x0054	RW	32	0x00000000	TX high-priority buffer retry counter
TXMSGSTS	0x0058	RO	32	0x00000000	Transmit message status register [31:30] TX message status 00 = successfully 10 = retry 11 = failed [28:0] Message ID
TXHMSGSTS	0x005C	RO	32	0x00000000	Transmit high-priority message status register [31:30] TX high-priority message status 00 = successfully 10 = retry 11 = failed [28:0] Message ID
ERRSTS	0x0060	RW	32	0x00000000	Error status register [4] CRC error [3] ACK error [2] FORM error [1] BIT error [0] STUFF error
ERRCNTR	0x0064	RO	32	0x00000000	Error counter register [24:16] TX error counter [8:0] RX error counter
RSVD4	0x0068-0x00fc	-	-	-	Reserved
AF	0x0100-0x100+(4*N)	RW	32	0x00000000	Receive acceptance filter register [31] Enable 1 = valid 0 = invalid [30] IDE 1 = extended frame 0 = normal frame [29] Extended data length

寄存器名称	地址偏移	类型	宽度	初始值	描述
					1 = match FD frame 0 = match normal frame [28:18] Basic ID [17:0] ID Extension
AFM	0x0140-0x140+(4*N)	RW	32	0x00000000	Receive acceptance filter mask register [28:18] Basic ID mask [17:0] ID Extension mask
RSVD5	0x0180-0x01FC	-	-	-	Reserved
RXB	0x0200	RO	32	0x00000000	Receive buffer/fifo window register
TXB	0x0204	WO	32	0x00000000	Transmit buffer/fifo window register
TXHB	0x0208	WO	32	0x00000000	Transmit high-priority buffer/fifo window register
TXBSTS	0x020C	RO	32	0x00000000	Transmit buffer/fifo status [31] txbwerr [15:0] txbspace
TXHBSTS	0x0210	RO	32	0x00000000	Transmit high-priority buffer/fifo status [31] txhbwerr [15:0] txhbspace
RXBSTS	0x0214	RO	32	0x00000000	Receive buffer/fifo status [15:0] rxbdepth

14.3 函数库使用方法

CAN 函数库使用方法如表 14-2 所示。

表 14-2 CAN 函数库使用方法

名称	描述
can_srst	Start hard reset
can_set_cmd	Enable working mode
can_set_brp	Set baud rate prescalar
can_set_btn_phseg1_nom	Set PROP_SEG+PHASE_SEG1 window’s width
can_set_btn_phseg2_nom	Set PHASE_SEG2 window’s width
can_set_btn_sjw_nom	Set sjw_nom
can_set_btn	Set BTN register
can_read_btn_phseg1_nom	Get PROP_SEG+PHASE_SEG1 window’s width
can_read_btn_phseg2_nom	Get PHASE_SEG2 window’s width
can_read_btn_sjw_nom	Get sjw_nom
can_set_btd_phseg1_d	Set PROP_SEG+PHASE_SEG1 window’s depth
can_set_btd_phseg2_d	Set PHASE_SEG2 window’s depth
can_set_btd_sjw_d	Set sjw_d
can_set_btd	Set BTD register
can_read_btd_phseg1_d	Get PROP_SEG+PHASE_SEG1 window’s depth
can_read_btd_phseg2_d	Get PHASE_SEG2 window’s depth
can_read_btd_sjw_d	Get sjw_d
can_read_is_bit	Get IS register bits function
can_set_ie_bit	Set IE register bits function
can_clear_ie_bit	Clear IE register bits function
can_read_ie_bit	Get IE register bits function

can_set_ic_bit	Set IC register bits function
can_set_cfg_bit_as_one	Set CFG register bits function
can_set_cfg_bit_as_zero	Clear CFG register bits function
can_read_cfg_bit	Get CFG register bits function
can_set_rxbcfg_rxb_start	Set RX buffer start offset in RXBCFG
can_read_rxbcfg_rxb_start	Get RX buffer start offset in RXBCFG
can_set_rxbcfg_rxb_end	Set RX buffer ending offset in RXBCFG
can_read_rxbcfg_rxb_end	Get RX buffer ending offset in RXBCFG
set_rxbcfg	Set RX buffer start and ending offset
can_set_txbcfg_txb_start	Set TX buffer start offset in TXBCFG
can_read_txbcfg_txb_start	Get TX buffer start offset in TXBCFG
can_set_txbcfg_txb_end	Set TX buffer ending offset in TXBCFG
can_read_txbcfg_txb_end	Get TX buffer ending offset in TXBCFG
set_txbcfg	Set TX buffer start and ending offset
can_set_txhbcfg_txhb_start	Set TX high-priority buffer start offset in TXHBCFG
can_read_txhbcfg_txhb_start	Get TX high-priority buffer start offset in TXHBCFG
can_set_txhbcfg_txhb_end	Set TX high-priority buffer ending offset in TXHBCFG
can_read_txhbcfg_txhb_end	Get TX high-priority buffer ending offset in TXHBCFG
set_txhbcfg	Set TX high-priority buffer start and ending offset
can_set_txbretry	Set TX buffer retry
can_read_txbretry	Get TX buffer retry
can_set_txhbretry	Set TX high-priority buffer retry counter
can_read_txhbretry	Get TX high-priority buffer retry counter
can_read_txmsgsts	Get TX message status
can_read_txmsgid	Get TX message ID
can_read_txhmsgsts	Get TX high-priority message status
can_read_txhmsgid	Get TX high-priority message ID
can_read_errsts	Get error status
can_read_errcntr_rec	Get RX error counter
can_read_errcntr_tec	Get TX error counter
can_set_af_bit_as_one	Set AF bits function
can_set_af_bit_as_zero	Clear AF bits function
can_read_af_bit	Get AF bits function
can_set_af_ie	Set AF ID Extension
can_read_af_ie	Get AF ID Extension
can_set_af_bid	Set AF basic ID
can_read_af_bid	Get AF basic ID
can_set_afm_iem	Set AFM ID Extension mask
can_read_afm_iem	Get AFM ID Extension mask
can_set_afm_bidm	Set AFM basic ID mask
can_read_afm_bid	Get AFM basic ID mask
can_read_rxb	Get RXB register
can_set_txb	Set TXB register
can_set_txhb	Set TXHB register
can_read_txbsts_tdbspace	Get txbspace
can_read_txbsts_txbwerr	Get txbwerr
can_read_txhbsts_tdbspace	Get txhbspace
can_read_txhbsts_txbwerr	Get txhbwerr
can_read_rxbsts	Get rxbdepth

14.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 CAN 参考设计：

- MCU_RefDesign\Keil_RefDesign\can
- MCU_RefDesign\GNU_RefDesign\cm1_can

15 以太网

15.1 特征

Gowin_EMPU_M1 包含 1 个通过 AHB 总线访问的以太网 Ethernet 模块：

- AHB 总线接口
- 实现 IEEE802.3 协议中对 Ethernet MAC 层的功能描述
- 支持 RGMII/GMII/MII 接口
- 支持 10/100/1000M 速率
- 支持全双工和半双工模式，半双工模式下支持冲突检测
- 支持用户可选是否自动添加和校验 CRC
- 支持自动添加 pad 功能
- 支持以太网帧分类统计
- 支持以太网帧错误统计
- 支持 IFG 可配置功能
- 支持 Jumbo 模式
- 支持全双工模式下的 Flow Control
- 支持 Management 接口 mdc、mdio

15.2 寄存器定义

Ethernet 寄存器定义如表 15-1 所示。

表 15-1 Ethernet 寄存器定义

寄存器名称	地址偏移	类型	宽度	初始值	描述
ETH_TX_DATA	0x000-0x5FF	WO	32	0x00000000	Transmit data registers
ETH_RX_DATA	0x000-0x5FFF	RO	32	0x00000000	Receive data registers
ETH_TX_LENGTH	0x600	RW	32	0x00000000	Transmit data length [31:11] Reserved [10:0] TX data length
ETH_TX_EN	0x604	RW	32	0x00000000	Transmit enable [31:1] Reserved [0] Enable TX

寄存器名称	地址偏移	类型	宽度	初始值	描述
ETH_TX_FAIL	0x608	RW	32	0x00000000	Transmit failed status [31:3] Reserved [2] TX late [1] TX excessive [0] TX failed
ETH_TX_IS	0x60C	RO	32	0x00000000	Transmit interrupt status [31:1] Reserved [0] TX interrupt status
ETH_TX_IC	0x610	WO	32	0x00000000	Transmit interrupt clear [31:1] Reserved [0] Clear TX interrupt
ETH_TX_IE	0x614	RW	32	0x00000000	Transmit interrupt enable [31:1] Reserved [0] Enable TX interrupt
RESERVED_1	0x618-0x67F	-	-	-	Reserved
ETH_RX_LEGHT	0x680	RO	32	0x00000000	Receive data length
ETH_RX_IS	0x684	RO	32	0x00000000	Receive interrupt status [31:1] Reserved [0] RX interrupt status
ETH_RX_IE	0x688	RW	32	0x00000000	Receive interrupt enable [31:1] Reserved [0] Enable RX interrupt
ETH_RX_IC	0x68C	WO	32	0x00000000	Receive interrupt clear [31:1] Reserved [0] Clear RX interrupt
RESERVED_2	0x690-0x6FFF	-	-	-	Reserved
MIIM_OP_MODE	0x700	RW	32	0x00000000	MIIM operation mode [31:1] Reserved [0] MIIM operation mode
MIIM_PHY_ADDR	0x704	RW	32	0x00000000	MIIM PHY address [31:5] Reserved [4:0] MIIM PHY address
MIIM_REG_ADDR	0x708	RW	32	0x00000000	MIIM reg address [31:5] Reserved [4:0] MIIM reg address
MIIM_WR_DATA	0x70C	RW	32	0x00000000	MIIM write data [31:16] Reserved [15:0] MIIM write data
MIIM_RD_DATA	0x710	RO	32	0x00000000	MIIM read data [31:16] Reserved [15:0] MIIM read data
MIIM_IS	0x714	RO	32	0x00000000	MIIM interrupt status [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid
MIIM_IE	0x718	RW	32	0x00000000	MIIM interrupt enable [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid

寄存器名称	地址偏移	类型	宽度	初始值	描述
MIIM_IC	0x71C	WO	32	0x00000000	MIIM interrupt clear [31:2] Reserved [1] MIIM operation end [0] MIIM read data valid
MIIM_OP_EN	0x720	RW	32	0x00000000	MIIM operation enable [31:1] Reserved [0] Enable MIIM operation
ETH_MODE	0x724	RW	32	0x00000000	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

15.3 函数库使用方法

Ethernet 函数库使用方法如表 15-2 所示。

表 15-2 Ethernet 函数库使用方法

名称	描述
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

15.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 Ethernet 参考设计：

- MCU_RefDesign\Keil_RefDesign\ethernet
- MCU_RefDesign\GNU_RefDesign\cm1_ethernet

16 嵌入式实时操作系统

Gowin_EMPU_M1 支持 uC/OS-III 和 FreeRTOS 嵌入式实时操作系统。

16.1 uC/OS-III

16.1.1 特征

- uC/OS-III 是一个可扩展的，可固化的，抢占式的实时内核，管理的任务个数不受限制
- uC/OS-III 是第三代内核，提供了现代实时内核所期望的功能，包括资源管理、同步、任务间通信等
- uC/OS-III 提供了很多其它实时内核所没有的特性，比如能在运行时测量运行性能，直接发送信号或消息给任务，任务能同时等待多个信号量和消息队列
- Gowin_EMPU_M1 已成功移植 uC/OS-III 参考设计
- uC/OS-III 源代码请在 uC/OS 官网 http://www.micrium.com 下载

16.1.2 操作系统版本

Gowin_EMPU_M1 参考设计使用的 uC/OS-III 版本为 V3.03.00。

16.1.3 操作系统配置

- 用户可以通过修改 UCOSIII_CONFIG\os_cfg.h 和 os_cfg_app.h 来配置 uC/OS-III
- 用户可以通过修改 UCOS_BSP\bsp.c 和 bsp.h 来支持所用开发板

16.1.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 uC/OS-III 参考设计：

- MCU_RefDesign\Keil_RefDesign\ucos_iii
- MCU_RefDesign\GNU_RefDesign\cm1_ucos_iii

16.2 FreeRTOS

16.2.1 特征

- FreeRTOS 是一个轻量级的实时操作系统
- FreeRTOS 作为一个轻量级的操作系统，功能包括：任务管理、时间管理、信号量、消息队列、内存管理、记录功能、软件定时器、协程等，可基本满足较小系统的需要
- FreeRTOS 操作系统是完全免费的操作系统，具有源码公开、可移植、可裁减、调度策略灵活的特点
- Gowin_EMPU_M1 已成功移植 FreeRTOS 参考设计
- FreeRTOS 源代码请在 FreeRTOS 官网 http://www.FreeRTOS.org 下载

16.2.2 操作系统版本

Gowin_EMPU_M1 参考设计使用的 FreeRTOS 版本为 V9.0.0。

16.2.3 操作系统配置

用户可以通过修改 include\FreeRTOSConfig.h 来配置 FreeRTOS。

16.2.4 参考设计

Gowin_EMPU_M1 支持 Keil 和 GNU 软件环境的 FreeRTOS 参考设计：

- MCU_RefDesign\Keil_RefDesign\free_rtos
- MCU_RefDesign\GNU_RefDesign\cm1_freertos


GOWIN高云
智慧逻辑 定制未来