

Gowin_EMPU(GW1NS-4C)ソフトウェア ユーザーガイド

IPUG931-2.1J, 2025-08-15

著作権について(2025)

著作権に関する全ての権利は、**Guangdong Gowin Semiconductor Corporation** に留保されています。

 **GOWIN** 及び **Gowin** は、当社により、中国、米国特許商標庁、及びその他の国において登録されています。商標又はサービスマークとして特定されたその他全ての文字やロゴは、それぞれの権利者に帰属しています。何れの団体及び個人も、当社の書面による許可を得ず、本文書の内容の一部もしくは全部を、いかなる視聴覚的、電子的、機械的、複写、録音等の手段によりもしくは形式により、伝搬又は複製をしてはなりません。

免責事項

当社は、GOWINSEMI Terms and Conditions of Sale (GOWINSEMI取引条件) に規定されている内容を除き、(明示的か又は黙示的に拘わらず) いかなる保証もせず、また、知的財産権や材料の使用によりあなたのハードウェア、ソフトウェア、データ、又は財産が被った損害についても責任を負いません。当社は、事前の通知なく、いつでも本文書の内容を変更することができます。本文書を参照する何れの団体及び個人も、最新の文書やエラッタ (不具合情報) については、当社に問い合わせる必要があります。

バージョン履歴

日付	バージョン	説明
2020/04/20	1.0J	初版。
2021/02/08	1.1J	● 周辺機器 AHB PSRAM Memory Interface のサポートを追加。 ● 周辺機器 AHB HyperRAM Memory Interface のサポートを追加。 ● 周辺機器 APB Nor Flash のサポートを追加。 ● GPIO が複数のポートタイプをサポートするように更新。 ● I²C が複数のポートタイプをサポートするように更新。 ● ソフトウェアバージョンと SDK をアップグレード。
2021/06/21	1.2J	既知の SPI 全二重読み出しおよび書き込みの問題を修正。
2022/12/16	1.3J	● 既知の SPI 分周係数の問題を修正。 ● RT-Thread Nano RTOS リファレンス・デザインを追加。 ● SDK を更新。
2024/01/16	2.0J	ソフトウェアプログラミング・ライブラリとソフトウェア・リファレンス・デザインを更新。
2025/08/15	2.1J	● SPI-Flash のドライバー関数を更新。 ● GPIO のドライバー関数を更新。 ● GMD の V2025.01 バージョンに適合するようにリファレンス・デザインを更新。 ● Code Coverage 機能と Profiling 機能のリファレンス・デザインを追加。 ● FIFO 周辺機器のリファレンス・デザインを追加。 ● ユーザー割り込みのリファレンス・デザインを追加。 ● 関連ソフトウェアのバージョン情報を更新。

目次

目次	i
図一覧	v
表一覧	vi
1 ソフトウェアプログラミング・ライブラリ	1
1.1 MCU ソフトウェアプログラミング	1
1.2 組み込み RTOS ソフトウェアプログラミング	3
2 メモリシステム	4
2.1 標準周辺機器のメモリマッピング	4
2.2 コアシステムのメモリマッピング	5
3 割り込み処理	7
4 UART	10
4.1 特徴	10
4.2 レジスタの定義	11
4.3 初期化の定義	12
4.4 ドライバーの使用方法	13
5 Timer	14
5.1 特徴	14
5.2 レジスタの定義	14
5.3 初期化の定義	15
5.4 ドライバーの使用方法	15
6 Watch Dog	17
6.1 特徴	17
6.2 レジスタの定義	18
6.3 初期化の定義	19
6.4 ドライバーの使用方法	19

7 GPIO.....	21
7.1 特徴.....	21
7.2 レジスタの定義	21
7.3 初期化の定義	24
7.4 ドライバーの使用方法.....	25
8 RTC	26
8.1 特徴.....	26
8.2 レジスタの定義	27
8.3 ドライバーの使用方法.....	28
9 SPI Master	30
9.1 特徴.....	30
9.2 レジスタの定義	30
9.3 初期化の定義	31
9.4 ドライバーの使用方法.....	31
10 System Controller.....	33
10.1 レジスタの定義	33
10.2 ドライバーの使用方法.....	33
11 I²C Master	35
11.1 特徴.....	35
11.2 レジスタの定義.....	35
11.3 ドライバーの使用方法	36
12 SysTick	38
12.1 特徴.....	38
12.2 レジスタの定義	38
12.3 ドライバーの使用方法.....	39
13 メモリ管理.....	40
13.1 特徴.....	40
13.2 ドライバーの使用方法.....	40
14 SPI-Flash Memory.....	41
14.1 特徴.....	41
14.2 レジスタの定義	41

14.3 ドライバーの使用方法.....	48
15 PSRAM Memory Interface	49
15.1 特徴.....	49
15.2 レジスタの定義	49
15.3 ドライバーの使用方法.....	51
16 HyperRAM Memory Interface.....	52
16.1 特徴.....	52
16.2 レジスタの定義	52
16.3 ドライバーの使用方法.....	54
17 RTOS.....	55
17.1 uC/OS-III	55
17.1.1 特徴	55
17.1.2 バージョン	55
17.1.3 構成	55
17.2 FreeRTOS	56
17.2.1 特徴	56
17.2.2 バージョン	56
17.2.3 構成	56
17.3 RT-Thread Nano	56
17.3.1 特徴	56
17.3.2 バージョン	57
17.3.3 構成	57
18 アプリケーションプログラム	58
18.1 UART	58
18.2 Timer.....	59
18.3 Watch Dog.....	59
18.4 GPIO.....	59
18.5 I ² C Master.....	60
18.6 SPI Master.....	60
18.7 RTC	60
18.8 SPI-Flash Memory.....	60
18.9 PSRAM Memory.....	61
18.10 HyperRAM Memory.....	61

18.11 Interrupt	61
18.12 DMM	61
18.13 AHB Master	62
18.14 APB Master	62
18.15 User Interrupt	62
18.16 FIFO	62
18.17 Code Coverage	63
18.18 Profiling	63
18.19 uC/OS-III	63
18.20 FreeRTOS	63
18.21 RT-Thread Nano	63

図一覧

図 4-1 UART の構造	11
図 5-1 Timer の構造	14
図 6-1 Watch Dog の動作フロー	17
図 7-1 GPIO の構造	21
図 8-1 RTC の構造	27

表一覧

表 1-1 MCU ソフトウェアプログラミング	1
表 2-1 標準周辺機器のメモリマッピングの定義	4
表 2-2 コアシステムのメモリマッピングの定義	5
表 3-1 割り込みコントローラの定義	7
表 4-1 UART レジスタの定義	11
表 4-2 UART 初期化の定義	12
表 4-3 UART ドライバーの使用方法	13
表 5-1 Timer レジスタの定義	15
表 5-2 Timer 初期化の定義	15
表 5-3 Timer ドライバーの使用	15
表 6-1 Watch Dog レジスタの定義	18
表 6-2 WatchDog 初期化の定義	19
表 6-3 WatchDog ドライバーの使用	19
表 7-1 GPIO レジスタの定義	22
表 7-2 GPIO 初期化の定義	24
表 7-3 GPIO ドライバーの使用方法	25
表 8-1 RTC レジスタの定義	27
表 8-2 RTC ドライバーの使用	28
表 9-1 SPI Master レジスタの定義	30
表 9-2 SPI Master 初期化の定義	31
表 9-3 SPI Master ドライバーの使用方法	32
表 10-1 SYSCON レジスタの定義	33
表 10-2 SYSCON ドライバーの使用	33
表 11-1 I ² C Master レジスタの定義	35
表 11-2 I ² C Master ドライバーの使用	36
表 12-1 SysTick レジスタの定義	38
表 12-2 SysTick ドライバーの使用	39

表 13-1 メモリ管理ドライバーの使用	40
表 14-1 SPI-Flash Memory レジスタの定義	41
表 14-2 SPI-Flash Memory ドライバーの使用	48
表 15-1 PSRAM Memory Interface レジスタの定義	50
表 15-2 PSRAM Memory Interface ドライバーの使用	51
表 16-1 HyperRAM Memory Interface レジスタの定義	53
表 16-2 HyperRAM Memory Interface ドライバーの使用	54

1 ソフトウェアプログラミング・ライブラリ

Gowin_EMPU(GW1NS-4C)のソフトウェアプログラミング・ライブラリ : …¥library。

Gowin_EMPU(GW1NS-4C)は、次のソフトウェアプログラミング方法をサポートしています。

- MCU ソフトウェアプログラミング
- 組込み RTOS ソフトウェアプログラミング

1.1 MCU ソフトウェアプログラミング

Gowin_EMPU(GW1NS-4C)の MCU ソフトウェアプログラミングのファイル一覧を表 1-1 に示します。

表 1-1 MCU ソフトウェアプログラミング

ファイル	説明
システムの定義	
startup_gw1ns4c.s	Cortex-M3コアのスタートアッププログラム
core_cm3.c core_cm3.h	Cortex-M3コアレジスタの定義
gw1ns4c.h	割り込みベクターテーブル、レジスタ、およびアドレスマッピングの定義
system_gw1ns4c.c	システム初期化とシステムクロックの定義
Linkerの定義	
gw1ns4c_flash.ld	GMD IDE Flashリンカー
周辺機器ドライバーの定義	
gw1ns4c_gpio.c	GPIOドライバー関数の定義

ファイル	説明
gw1ns4c_gpio.h	
gw1ns4c_uart.c gw1ns4c_uart.h	UART ドライバー関数の定義
gw1ns4c_timer.c gw1ns4c_timer.h	Timer ドライバー関数の定義
gw1ns4c_wdog.c gw1ns4c_wdog.h	Watch Dog ドライバー関数の定義
gw1ns4c_spi.c gw1ns4c_spi.h	SPI Master ドライバー関数の定義
gw1ns4c_i2c.c gw1ns4c_i2c.h	I ² C Master ドライバー関数の定義
gw1ns4c_rtc.c gw1ns4c_rtc.h	RTC ドライバー関数の定義
gw1ns4c_misc.c gw1ns4c_misc.h	割り込み優先度およびSysTickの定義
gw1ns4c_syscon.c gw1ns4c_syscon.h	システム制御ドライバ関数の定義
ミドルウェアの定義	
malloc.c malloc.h	動的メモリ管理mallocとfree関数の再定義
retarget.c uart.c uart.h	UART printf関数の再定義 UART初期化の定義
delay.c delay.h	遅延関数の定義
gpio.c gpio.h	GPIO初期化の定義
psram.c psram.h	PSRAM Memory Interface ドライバー関数の定義
hyper_ram.c hyper_ram.h	HyperRAM Memory Interface ドライバー関数の定義
spi_flash.c spi_flash.h	SPI-Flash Memory ドライバー関数の定義

1.2 組み込み RTOS ソフトウェアプログラミング

Gowin_EMPU(GW1NS-4C)は、次の組み込み RTOS ソフトウェアプログラミングをサポートしています。

- uC/OS-III
- FreeRTOS
- RT-Thread Nano

2メモリシステム

2.1 標準周辺機器のメモリマッピング

Gowin_EMPU(GW1NS-4C)標準周辺機器のメモリマッピングのアドレスの定義を表 2-1 に示します。

表 2-1 標準周辺機器のメモリマッピングの定義

標準周辺機器	タイプ	アドレスマッピング	説明
Flash	-	0x00000000	32KB
SRAM	-	0x20000000	2KB、4KB、8KB、または16KB
TIMER0	TIMER_TypeDef	0x40000000	タイマー0(Timer 0)
TIMER1	TIMER_TypeDef	0x40001000	タイマー1(Timer 1)
UART0	UART_TypeDef	0x40004000	汎用非同期送受信回路0(UART0)
UART1	UART_TypeDef	0x40005000	汎用非同期送受信回路1(UART1)
RTC	RTC_TypeDef	0x40006000	リアルタイムクロック(RTC)
WatchDog	WDog_TypeDef	0x40008000	ウォッチドッグ (Watch Dog)
GPIO0	GPIO_TypeDef	0x40010000	汎用入出力ポート(GPIO)
SYSCON	SYSCON_TypeDef	0x4001F000	システムコントローラ(System Controller)
I2C	I2C_TypeDef	0x40002000	I ² C マスター (I ² C Master)

標準周辺機器	タイプ	アドレスマッピング	説明
SPI	SPI_TypeDef	0x40002200	SPI マスター (SPI Master)
APB2 Master 1	-	0x40002400	APB2 Master [1]
APB2 Master 2	-	0x40002500	APB2 Master [2]
APB2 Master 3	-	0x40002600	APB2 Master [3]
APB2 Master 4	-	0x40002700	APB2 Master [4]
APB2 Master 5	-	0x40002800	APB2 Master [5]
APB2 Master 6	-	0x40002900	APB2 Master [6]
APB2 Master 7	-	0x40002A00	APB2 Master [7]
APB2 Master 8	-	0x40002B00	APB2 Master [8]
APB2 Master 9	-	0x40002C00	APB2 Master [9]
APB2 Master 10	-	0x40002D00	APB2 Master [10]
APB2 Master 11	-	0x40002E00	APB2 Master [11]
APB2 Master 12	-	0x40002F00	APB2 Master [12]
AHB2 Master	-	0xA0000000	AHB2 Master

2.2 コアシステムのメモリマッピング

Gowin_EMPU(GW1NS-4C)のコアシステムメモリマッピングの定義を表 2-2 に示します。

表 2-2 コアシステムのメモリマッピングの定義

システム制御	タイプ	アドレスマッピング	説明
ITM	ITM_Type	0xE0000000	ITM configuration structure
DWT	DWT_Type	0xE0001000	DWT configuration structure
CoreDebug	CoreDebug_Type	0xE000EDF0	Core Debug configuration structure
ETM	ETM_Type	0xE0041000	ETM configuration structure
SysTick	SysTick_Type	0xE000E010	SysTick configuration structure
NVIC	NVIC_BASE	0xE000E100	NVIC configuration structure
SCnSCB	SCnSCB_Type	0xE000E000	System control Register not in SCB
SCB	SCB_Type	0xE000ED00	SCB configuration structure

システム制御	タイプ	アドレスマッピング	説明
TPIU	TPIU_Type	0xE0040000	TPIU configuration structure

3 割り込み処理

ネスト型ベクタ割り込みコントローラの特徴は以下のとおりです：

- 最大 **32** の低遅延割り込みをサポート
- ユーザーが利用可能な **6** つの割り込み処理信号(**USER_INT0~5**)を提供
- **0~7** のプログラマブル割り込み優先度をサポート
- 低遅延割り込みと例外処理
- 割り込み信号エッジまたはパルス検出
- 割り込み優先度の動的調整

割り込みコントローラの定義を表 **3-1** に示します。

表 3-1 割り込みコントローラの定義

アドレス	Interrupt	番号	説明
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

アドレス	Interrupt	番号	説明
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	USER_INT0_Handler	1	16+ 1: USER_INT0
0x00000048	UART1_Handler	2	16+ 2: UART 1 RX and TX Handler
0x0000004C	USER_INT1_Handler	3	16+ 3: USER_INT1
0x00000050	USER_INT2_Handler	4	16+ 4: USER_INT2
0x00000054	RTC_Handler	5	16+ 5: RTC Handler
0x00000058	PORT0_COMB_Handler	6	16+ 6: GPIO Port 0 Combined Handler
0x0000005C	USER_INT3_Handler	7	16+ 7: USER_INT3
0x00000060	TIMER0_Handler	8	16+ 8: TIMER 0 Handler
0x00000064	TIMER1_Handler	9	16+ 9: TIMER 1 Handler
0x00000068	0	-	16+10: Reserved
0x0000006C	I2C_Handler	11	16+11: I2C Handler
0x00000070	UARTOVF_Handler	12	16+12: UART 0,1 Overflow Handler
0x00000074	USER_INT4_Handler	13	16+13: USER_INT4
0x00000078	USER_INT5_Handler	14	16+14: USER_INT5
0x0000007C	Spare15_Handler	15	16+ 15: Not Used
0x00000080	PORT0_0_Handler	16	16+16: GPIO Port 0 pin 0 Handler
0x00000084	PORT0_1_Handler	17	16+17: GPIO Port 0 pin 1 Handler
0x00000088	PORT0_2_Handler	18	16+18: GPIO Port 0 pin 2 Handler
0x0000008C	PORT0_3_Handler	19	16+19: GPIO Port 0 pin 3 Handler
0x00000090	PORT0_4_Handler	20	16+20: GPIO Port 0 pin 4 Handler
0x00000094	PORT0_5_Handler	21	16+21: GPIO Port 0 pin 5 Handler
0x00000098	PORT0_6_Handler	22	16+22: GPIO Port 0 pin 6 Handler
0x0000009C	PORT0_7_Handler	23	16+23: GPIO Port 0 pin 7 Handler
0x000000A0	PORT0_8_Handler	24	16+24: GPIO Port 0 pin 8 Handler
0x000000A4	PORT0_9_Handler	25	16+25: GPIO Port 0 pin 9 Handler
0x000000A8	PORT0_10_Handler	26	16+26: GPIO Port 0 pin 10 Handler
0x000000AC	PORT0_11_Handler	27	16+27: GPIO Port 0 pin 11 Handler

アドレス	Interrupt	番号	説明
0x000000B0	PORT0_12_Handler	28	16+28: GPIO Port 0 pin 12 Handler
0x000000B4	PORT0_13_Handler	29	16+29: GPIO Port 0 pin 13 Handler
0x000000B8	PORT0_14_Handler	30	16+30: GPIO Port 0 pin 14 Handler
0x000000BC	PORT0_15_Handler	31	16+31: GPIO Port 0 pin 15 Handler

4UART

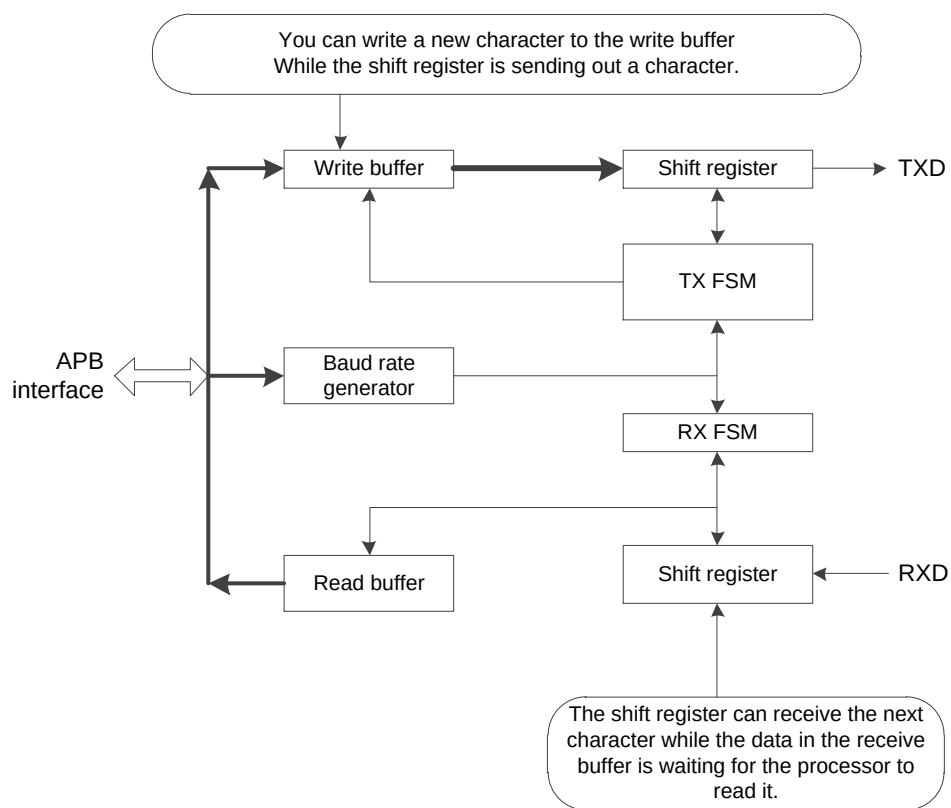
4.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 2 つの UART 周辺機器が含まれています。この 2 つの UART 周辺機器の機能は：

- 最大ボーレートは 921.6Kbit/s
- パリティビットなし
- 8 ビットのデータビット
- 1 ビットのストップビット

UART の構造を図 4-1 に示します。

図 4-1 UART の構造



UART は HSTM(高速テストモード)をサポートしています。レジスタ CTRL[6]に 1 をセットすると、シリアル・データはサイクルごとに 1 ビットで送信されるため、テキスト情報は短時間で送信できます。

ユーザーが UART を有効にする場合、ボーレート分周レジスタを設定する必要があります。たとえば、APB1 バス周波数が 12MHz で動作し、ボーレートが 9600 である必要がある場合は、ボーレート分周レジスタを $12000000/9600 = 1250$ に設定します。

4.2 レジスタの定義

UART レジスタの定義を表 4-1 に示します。UART レジスタの定義は、library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 4-1 UART レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[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
INTSTATU/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 初期化の定義

UART 初期化の定義を表 4-2 に示します。UART 初期化の定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_uart.h` にあります。

表 4-2 UART 初期化の定義

名称	タイプ	値	説明
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 high speed test mode

4.4 ドライバーの使用法

UART ドライバーの使用法を表 4-3 に示します。UART ドライバーの定義は、`library\libraries\drivers\inc\gw1ns4c_uart.c` にあります。

表 4-3 UART ドライバーの使用法

名称	説明
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

5 Timer

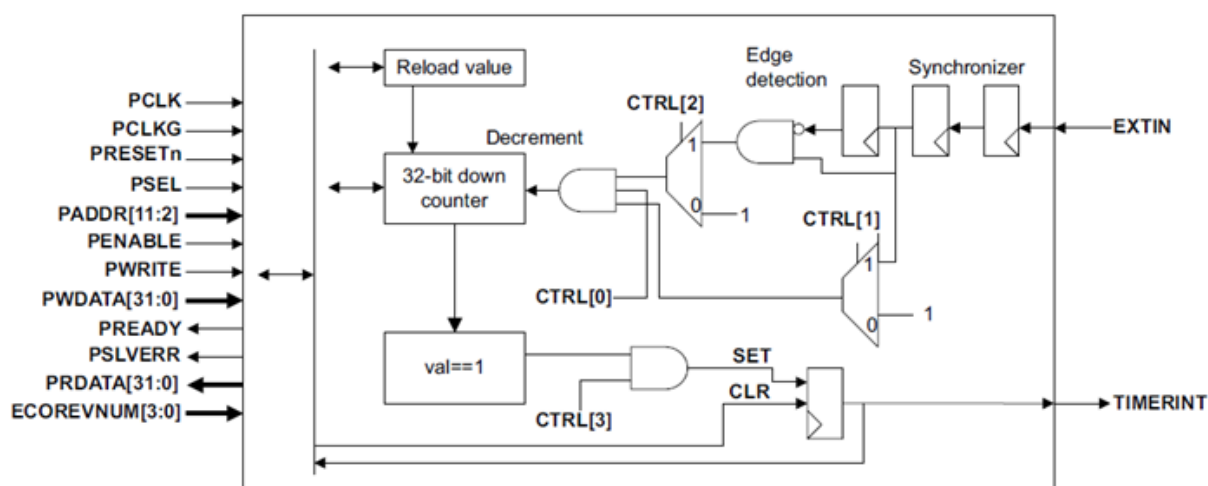
5.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 2 つの同期標準 Timer 周辺機器が含まれています。

- 32 ビットカウンタ
- 割り込み要求信号を生成可能
- 外部入力信号 EXTIN でクロックを有効にすることをサポート
- Timer0: EXTIN は GPIO [1]に接続
- Timer1: EXTIN は GPIO [6]に接続

Timer の構造を図 5-1 に示します。

図 5-1 Timer の構造



5.2 レジスタの定義

Timer レジスタの定義を表 5-1 に示します。Timer レジスタの定義は、

library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 5-1 Timer レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
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
RELOAD	0x008	RW	32	0x00000000	[31:0] Reload value, writing to this register sets the current value
INTSTATUS/ NTCLEAR	0x00C	RW	1	0x0	[0] Timer interrupt, write 1 to clear

5.3 初期化の定義

Timer 初期化の定義を表 5-2 に示します。Timer 初期化の定義は、library¥libraries¥drivers¥inc¥gw1ns4c_timer.h にあります。

表 5-2 Timer 初期化の定義

名称	タイプ	値	説明
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 ドライバーの使用方法

Timer ドライバーの使用方法を表 5-3 に示します。Timer ドライバーの定義は、library¥libraries¥drivers¥inc¥gw1ns4c_timer にあります。

表 5-3 Timer ドライバーの使用

名称	説明
TIMER_Init	Initializes TIMEx
TIMER_StartTimer	Starts TIMEx
TIMER_StopTimer	Stops TIMEx
TIMER_GetIRQStatus	Returns TIMEx interrupt status
TIMER_ClearIRQ	Clears TIMEx interrupt status
TIMER_GetReload	Returns TIMEx reload value

名称	説明
TIMER_SetReload	Sets TIMEx reload value
TIMER_GetValue	Returns TIMEx current value
TIMER_SetValue	Sets TIMEx current value
TIMER_EnableIRQ	Enable TIMEx interrupt request
TIMER_DisableIRQ	Disable TIMEx interrupt request

6 Watch Dog

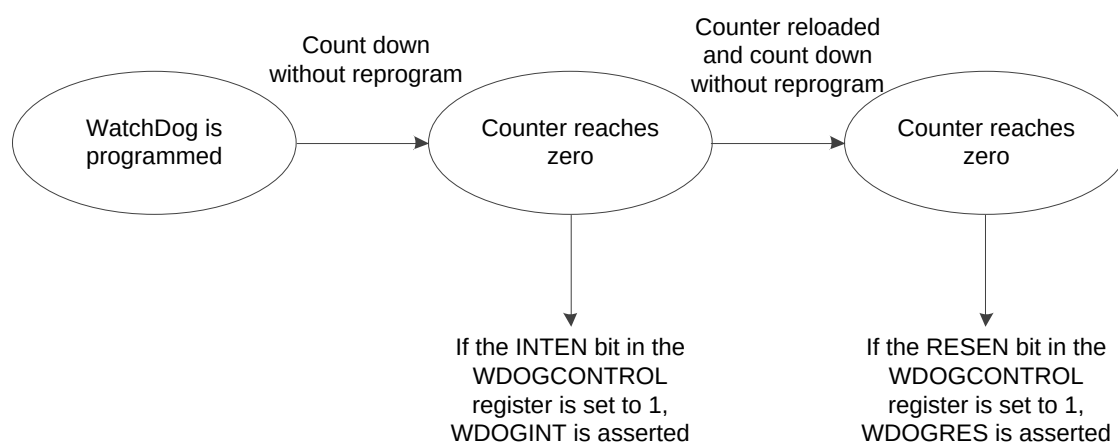
6.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 1 つの Watch Dog 周辺機器が含まれています。

- LOAD レジスタによって初期化された 32 ビットダウンカウンタに基づく
- 割り込み要求を生成
- クロックがイネーブルされた場合、ダウンカウンタは WDOGCLK 信号の立ち上がりエッジでトリガ。
- 割り込みを監視。カウンタが 0 になると、リセット要求が生成され、カウンタが停止
- ソフトウェア障害時のリセットに対応

Watch Dog の動作フローを図 6-1 に示します。

図 6-1 Watch Dog の動作フロー



6.2 レジスタの定義

Watch Dog レジスタの定義を表 6-1 に示します。Watch Dog レジスタの定義は、library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 6-1 Watch Dog レジスタの定義

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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
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 WDOGIN value

6.3 初期化の定義

Watch Dog 初期化の定義を表 6-2 に示します。Watch Dog 初期化の定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_wdog.h` にあります。

表 6-2 WatchDog 初期化の定義

名称	タイプ	値	説明
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 ドライバーの使用方法

WatchDog ドライバーの使用方法を表 6-3 に示します。Watch Dog ドライバーの定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_wdog.c` にあります。

表 6-3 WatchDog ドライバーの使用

名称	説明
WDOG_Init	Initializes Watch Dog
WDOG_RestartCounter	Restart Watch Dog 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

名称	説明
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

7 GPIO

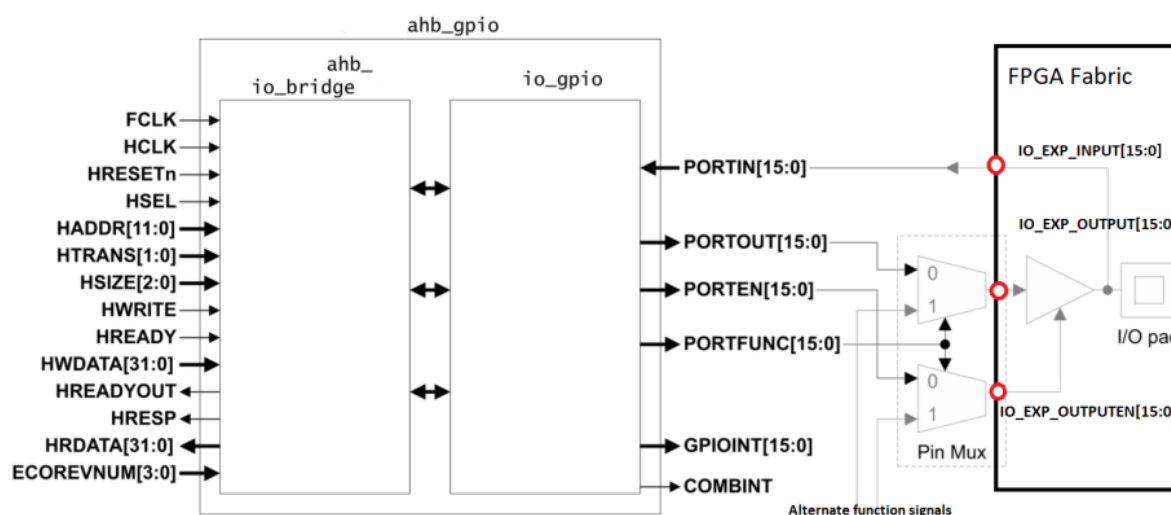
7.1 特徴

Gowin_EMPU(GW1NS-4C)には、AHB バスを介してアクセスされる 1 つの 16 ビット GPIO 周辺機器が含まれています：

- FPGA コアシステムに接続
- すべての IO ピンは割り込みを生成可能
- ビットマスクをサポート
- ピン多重化

GPIO の構造を図 7-1 に示します。

図 7-1 GPIO の構造



7.2 レジスタの定義

GPIO レジスタの定義を表 7-1 に示します。GPIO レジスタの定義は、`library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h` にあります。

表 7-1 GPIO レジスタの定義

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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 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 input 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 input 1 indicates the signal direction as output
ALTFUNCS ET	0x0018	RW	16	0x0000	[15:0] Alternative function set Write 1 to set the ALTFUNC bit Write 0 no effect Read back 0 for I/O 1 for an alternate function
ALTFUNCC LR	0x001C	RW	16	0x0000	[15:0] Alternative 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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 configuration 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 configuration 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 effect Read back IRQ status register
MASKLOW	0x0400	RW	16	0x----	Lower 8-bits masked access

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
BYTE	-0x07FC				[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
MASKHIGH BYTE	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 初期化の定義

GPIO 初期化の定義を表 7-2 に示します。GPIO 初期化の定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_gpio.h` にあります。

表 7-2 GPIO 初期化の定義

名称	タイプ	値	説明
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	16 bits GPIO Pins

名称	タイプ	値	説明
		GPIO_Pin_15	
GPIO_Mode	GPIO_Mode_TypeDef	GPIO_Mode_IN GPIO_Mode_OUT GPIO_Mode_AF	16 bits GPIO Pins mode
GPIO_Int	GPIO_Int_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 ドライバーの使用方法

GPIO ドライバーの使用方法を表 7-3 に示します。GPIO ドライバーの定義は、`library\libraries\drivers\inc\gw1ns4c_gpio.c` にあります。

表 7-3 GPIO ドライバーの使用方法

名称	説明
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_MaskedWrite	Setups GPIOx output value using masked access

8_RTC

8.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 1 つの 32 ビット RTC 周辺機器が含まれています。この 2 つの RTC 周辺機器の機能は：

- APB バスインターフェース
- 32-bit カウンター
- 32-bit Match レジスタ
- 32 ビットコンパレータ

MCU は、APB バスインターフェースを介して RTC と通信します。32 ビットカウンタは、クロック CLK1HZ(ポート rtc_src_clk からの 3.072MHz のクロック入力、RTC 内部で 1Hz に分周されます)の連続した立ち上がりエッジでインクリメントされます。

このカウンタは非同期カウンタであり、リロード不可です。システムがリセットされると、このカウンタは 1 から最大 0xFFFFFFFFF までインクリメントされ、その後 0 に戻ってインクリメントされ続けます。

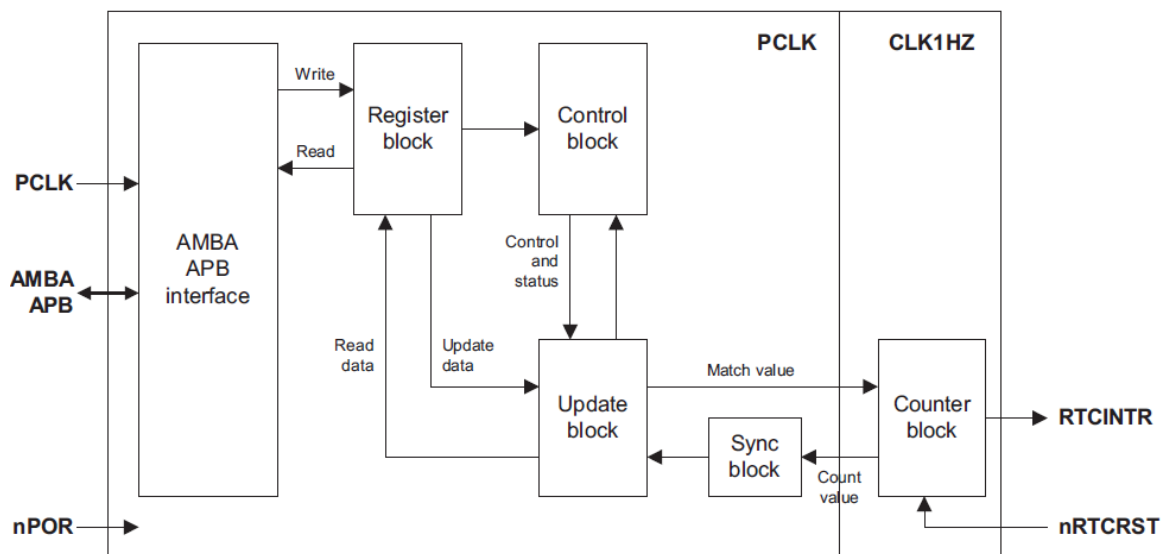
Load レジスタ RTC_LOAD_VALUE に書き込むことで RTC をロード・更新します。

Data レジスタ RTC_CURRENT_DATA を読み出すことで RTC の現在のクロックを取得します。

RTC_MATCH_VALUE レジスタに書き込むことで Match レジスタをプログラムします。

RTC の構造を図 8-1 に示します。

図 8-1 RTC の構造



8.2 レジスタの定義

RTC レジスタの定義を表 8-1 に示します。RTC レジスタの定義は、`library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h` にあります。

表 8-1 RTC レジスタの定義

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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_CTLLOLER_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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[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 [0] Clear current interrupt

8.3 ドライバーの使用方法

RTC ドライバーの使用方法を表 8-2 に示します。RTC ドライバーの定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_rtc.c` にあります。

表 8-2 RTC ドライバーの使用

名称	説明
RTC_init	Initializes RTC
Get_Current_Value	Gets RTC current value of data register
Set_Match_Value	Sets RTC match value of match register
Get_Match_Value	Gets RTC match value of match register
Set_Load_Value	Sets RTC load value of load register
Get_Load_Value	Gets RTC load value of load register
Start_RTC	Starts RTC counter
Close_RTC	Closes RTC counter
RTC_Inter_Mask_Set	Sets RTC interrupt mask
Get_RTC_Control_value	Gets value of control register

名称	説明
RTC_Inter_Mask_Clr	Clears RTC interrupt mask
Get_RTC_Inter_Mask_value	Gets RTC interrupt mask
Clear_RTC_interrupt	Clears RTC interrupt

9 SPI Master

9.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 1 つの SPI Master 周辺機器が含まれています。

- APB バスインターフェース
- 全二重同期シリアル・データ転送
- Master 動作モードをサポート
- クロックの極性と位相は構成可能
- SPI によって生成されるシリアル・クロック周波数は構成可能
- データ受信レジスタとデータ送信レジスタのデータ幅は、8 ビット

9.2 レジスタの定義

SPI Master レジスタの定義を表 9-1 に示します。SPI Master レジスタの定義は、library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 9-1 SPI Master レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
RDATA	0x00	RO	8	0x00	Reads data register
WDATA	0x04	WO	8	0x00	Writes data register
STATUS	0x08	RW	8	0x00	[7] Error status [6] Receives ready status [5] Transmits ready status [4] Be transmitting [3] Transmits overrun error

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					status [2] Receives overrun error status [1:0] Reserved
SSMASK	0x0C	RW	8	0x00	Unused selected slave address
CTRL	0x10	RW	5	0x00	[4:3] Clock selected [2] Polarity [1] Phase [0] Direction

9.3 初期化の定義

SPI Master 初期化の定義を表 9-2 に示します。SPI Master 初期化の定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_spi.h` にあります。

表 9-2 SPI Master 初期化の定義

名称	タイプ	値	説明
DIRECTION	uint8_t	1/0	MSB/LSB first transmission 0: MSB first; 1: LSB 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 polarity 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 $CLKDIV = (CLKSEL + 1) * 2$

9.4 ドライバーの使用方法

SPI Master ドライバーの使用方法を表 9-3 に示します。SPI Master ドライバーの定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_spi.c` にあります。

表 9-3 SPI Master ドライバーの使用法

名称	説明
SPI_Init	Initializes SPI Master
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_ReadWriteByte	Full duplex read and write a byte
SPI_WriteData	Writes data
SPI_ReadData	Reads data
SPI_Select_Slave	Selects slave

10 System Controller

10.1 レジスタの定義

SYSCON レジスタの定義を表 10-1 に示します。SYSCON レジスタの定義は、library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 10-1 SYSCON レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
REMAP	0x000	RW	1	0x0	Remap control register
PMUCTRL	0x004	RW	1	0x0	PMU control register
RESETOP	0x008	RW	1	0x0	reset option register
RESERVED0	0x00C	–	–	–	Reserved
RSTINFO	0x010	RW	3	0x0	[2] Lockup reset [1] Watchdog reset request [0] System reset request

10.2 ドライバーの使用方法

SYSCON ドライバーの使用方法を表 10-2 に示します。SYSCON ドライバーの定義は、library¥libraries¥drivers¥inc¥gw1ns4c_syscon.c にあります。

表 10-2 SYSCON ドライバーの使用

名称	説明
SYSCON_Init	Initializes SYSCON
SYSCON_GetRemap	Returns REMAP

名称	説明
SYSCON_GetPmuctrlEnable	Returns PMUCTRL Enable
SYSCON_GetResetopLockuprst	Returns RESETOP LOCKUPRST
SYSCON_GetRstinfoSysresetreq	Returns RSTINFO SYSRESETREQ
SYSCON_GetRstinfoWdogresetreq	Returns RSTINFO SYSRESETREQ
SYSCON_GetRstinfoLockreset	Returns RSTINFO SYSRESETREQ

11 I²C Master

11.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB バスを介してアクセスされる 1 つの I²C Master 周辺機器が含まれています。

- APB バスインターフェース
- 業界標準の I²C プロトコルに準拠
- バスアービトラレーションおよびアービトラレーションロストを検出
- バスビジー状態を検出
- 割込み標識を生成
- 開始、停止、繰り返し開始、及び応答情報を生成
- 開始、停止、及び繰り返し開始の検出をサポート
- 7 ビットアドレッシングモードをサポート

11.2 レジスタの定義

I²C Master レジスタの定義を表 11-1 に示します。I²C Master レジスタの定義は、library¥libraries¥cmsis¥cm3¥device_support¥gw1ns4c.h にあります。

表 11-1 I²C Master レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
PRER	0x00	RW	16	0xFFFF	Clock prescale register [15:0] Prescale value = sys_clk/(5*SCL)-1
CTR	0x04	RW	8	0x00	[7] Enable I ² C function [6] Enable I ² C interrupt

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[5:0] Reserved
TXR	0x08	WO	8	0x00	[7:1] Next transmission data [0] Data direction
RXR	0x0C	RO	8	0x00	[7:0] Last received data
CR	0x010	WO	8	0x00	[7] Start transmission status [6] Over transmission status [5] Read enable, read data from slave [4] Write enable, write data to slave [3] Acknowledge [2:1] Reserved [0] Interrupt acknowledge
SR	0x14	RO	8	0x00	[7] Receive acknowledge signal from slave [6] I2C busy status [5] Arbitration loss [4:2] Reserved [1] Data transmission status flag [0] Interrupt flag

11.3 ドライバーの使用法

I²C Master ドライバーの使用法を表 11-2 に示します。I²C Master ドライバーの定義は、`library¥libraries¥drivers¥inc¥gw1ns4c_i2c.c` にあります。

表 11-2 I²C Master ドライバーの使用

名称	説明
I2C_Init	I ² C Initialization
I2C_SendByte	Sends a byte to I ² C bus
I2C_SendBytes	Sends multiple bytes to I ² C bus
I2C_SendData	Sends multiple bytes to I ² C bus once time
I2C_ReceiveByte	Reads a byte from I ² C bus
I2C_ReadBytes	Reads multiple bytes from I ² C bus
I2C_ReceiveData	Reads multiple bytes from I ² C bus once time
I2C_Rate_Set	Sets I ² C traffic rate

名称	説明
I2C_Enable	Enable I ² C bus
I2C_UnEnable	Disable I ² C bus
I2C_InterruptOpen	Opens I ² C interrupt
I2C_InterruptClose	Closes I ² C interrupt

12SysTick

12.1 特徴

Gowin_EMPU(GW1NS-4C)には、自動リロードおよびオーバーフロー割り込み機能を備えた 24 ビットのシステムティックタイマー(SysTick)が含まれています。このタイマーは一定時間間隔を提供します。

12.2 レジスタの定義

SysTick レジスタの定義を表 12-1 に示します。SysTick レジスタの定義は、library¥libraries¥cmsis¥cm3¥core_support¥mdk(gmd)¥core_cm3.h にあります。

表 12-1 SysTick レジスタの定義

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
CTRL	0x00	RW	17	0x00000	[16] Counts down to zero flag [2] External clock source [1] Enable interrupt request [0] Enable SysTick
LOAD	0x04	RW	24	0x000000	[23:0] Reloads value
VAL	0x08	RW	24	0x000000	[23:0] Returns current count down value
CALIB	0x0C	RW	32	0x00000000	[31] Whether a separate reference clock is provided [30] Whether the TENMS value is exact [23:0] 10ms calibration value

12.3 ドライバーの使用方法

SysTick ドライバーの使用を表 12-2 に示します。SysTick ドライバーの定義は、library¥libraries¥drivers¥inc¥gw1ns4c_misc.c にあります。

表 12-2 SysTick ドライバーの使用

名称	説明
SysTick_Config	Initializes and starts the SysTick counter and interrupt
SysTick_CLKSourceConfig	Configure the SysTick clock source

13メモリ管理

13.1 特徴

Gowin_EMPU(GW1NS-4C)は、`malloc` と `free` 関数の再定義による動的メモリ管理をサポートします。

13.2 ドライバーの使用方法

メモリ管理ドライバーの一覧を表 13-1 に示します。メモリ管理ドライバーの定義は、`library¥middlewares¥dmm¥malloc.c` にあります。

表 13-1 メモリ管理ドライバーの使用

名称	説明
<code>mem_init</code>	Initializes memory management
<code>mymemset</code>	Sets the values of memory space
<code>mymemcpy</code>	Copies from source to destination
<code>mymemcmp</code>	Compares source with destination
<code>mymalloc</code>	Allocates a memory space dynamically
<code>myfree</code>	Frees a memory space

14SPI-Flash Memory

14.1 特徴

Gowin_EMPU(GW1NS-4C)には、APB2 Master インターフェースを通じて拡張された SPI-Flash Memory 周辺機器が含まれています。

- シリアル不揮発性 Flash。
- APB インターフェースによる構成をサポート。
- 標準の SPI インターフェースをサポート。
- 読み出し、書き込み、および消去機能をサポート。
- GW1NSR-4C/GW1NSER-4C デバイス QN48G パッケージをサポート。

14.2 レジスタの定義

SPI-Flash Memory レジスタの定義を表 14-1 に示します。SPI-Flash Memory レジスタの定義は、library¥middlewares¥spi_flash¥spi_flash.h にあります。

表 14-1 SPI-Flash Memory レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
IDREV	0x00	RO	32	0x02002000	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	0x00020780	SPI transfer format register [31:18] Reserved [17:16] Address length in bytes

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					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 1 = MOSI is bi-directional signal [3] Transfer data with the least 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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					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 [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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					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 [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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[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 in 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 [31:6] Reserved [5] Enable the slave command interrupt [4] Enable the end of SPI transfer interrupt

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
RESERVED3[3]	0x54-0x5C	-	-	-	Reserved
SLVST	0x60	RW	32	0x0	SPI slave status register [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 [8] 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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					10 = 8 words 11 = 16 words

14.3 ドライバーの使用法

SPI-Flash Memory ドライバーの使用法を表 14-2 に示します。
SPI-Flash Memory ドライバーの定義は、
library¥middlewares¥spi_flash¥spi_flash.c にあります。

表 14-2 SPI-Flash Memory ドライバーの使用法

名称	説明
spi_flash_init	Initializes SPI-Flash Memory
spi_flash_read	Read data from SPI-Flash Memory
spi_flash_write	Write data into SPI-Flash Memory
spi_flash_page_program	Write data into SPI-Flash Memory with pages
spi_flash_4ksector_erase	Erase SPI-Flash Memory with a 4k sector
spi_flash_64ksector_erase	Erase SPI-Flash Memory with a 64k sector

15 PSRAM Memory Interface

15.1 特徴

Gowin_EMPU(GW1NS-4C)には、AHB2 Master インターフェースを通じて拡張された PSRAM Memory Interface 周辺機器が含まれています：

- AHB インターフェースによる構成をサポート。
- 標準の PSRAM インターフェースと互換性あり。
- GW1NSR-4C デバイス MG64P パッケージをサポート。
- 8 ビットの PSRAM width をサポート。
- 16 ビットの DQ bus width をサポート。
- プログラム可能なバースト長：16、32、64、および 128。
- クロック比 1：2 をサポート。
- 初期遅延 6 をサポート。
- 固定遅延モードをサポート。
- 電源オフオプションをサポート。
- 構成可能なドライブ強度
- 構成可能なセルフリフレッシュ領域
- 構成可能なリフレッシュレート

15.2 レジスタの定義

PSRAM Memory Interface レジスタの定義を表 15-1 に示します。
PSRAM Memory Interface レジスタの定義は、
library¥middlewares¥psram¥psram.h にあります。

表 15-1 PSRAM Memory Interface レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
CMD	0x00	RW	1	0x0	Command register [0] Operation type 0 = Read operation 1 = Write operation
ADDRESS	0x04	RW	21	0x0	Address register [20:0] Address of reading and writing data
WR_DATA0	0x08	RW	32	0x0	Write data register 0 [31:0] Write first 32bit data
WR_DATA1	0x0C	RW	32	0x0	Write data register 1 [31:0] Write second 32bit data
WR_DATA2	0x10	RW	32	0x0	Write data register 2 [31:0] Write third 32bit data
WR_DATA3	0x14	RW	32	-	Write data register 3 [31:0] Write fourth 32bit data
CMD_EN	0x18	WO	1	-	Command enable register [0] Enable PSRAM
READ_DONE	0x1C	RW	1	-	Read status register [0] Read done flag, auto set 1 if it is done, and need MCU to clear
RD_DATA0	0x20	RO	32	-	Read data register 0 [31:0] Read first 32bit data
RD_DATA1	0x24	RO	32	-	Read data register 1 [31:0] Read second 32bit data
RD_DATA2	0x28	RO	32	-	Read data register 2 [31:0] Read third 32bit data
RD_DATA3	0x2C	RO	32	-	Read data register 3 [31:0] Read fourth 32bit data
INTI_DONE	0x30	RO	1	-	Initialization done register [0] PSRAM hardware initialization done flag 0 = Initialization failed 1 = Initialization done

15.3 ドライバーの使用法

PSRAM Memory Interface ドライバーの使用を表 15-2 に示します。
PSRAM Memory Interface ドライバーの定義は、
library¥middlewares¥psram¥psram.c にあります。

表 15-2 PSRAM Memory Interface ドライバーの使用

名称	説明
PSRAM_Check_Init_Status	Check the status of PSRAM Memory initialization
PSRAM_Mode_Set	Set the mode for PSRAM Memory write and read
PSRAM_Address_Set	Set the address of PSRAM Memory and save data into this address
PSRAM_Read_Data_Buff	Read data from the buffer of PSRAM Memory
PSRAM_Cmd_Enable	Enable the command of PSRAM Memory
PSRAM_Read_Done_Flag	Get the flag of read PSRAM Memory done
PSRAM_Clear_Read_Done_Flag	Clear the flag of read PSRAM Memory done
PSRAM_Write_Data_Buff	Write data into the buffer of PSRAM Memory
PSRAM_Cmd_Unable	Disable the command of PSRAM Memory
PSRAM_Write_Data_Package	Write a package data into PSRAM Memory
PSRAM_Read_Data_Package	Read a package data from PSRAM Memory

16HyperRAM Memory Interface

16.1 特徴

Gowin_EMPU(GW1NS-4C)には、AHB2 Master インターフェースを通じて拡張された HyperRAM Memory Interface 周辺機器が含まれています:

- AHB インターフェースによる構成をサポート。
- 標準の HyperRAM インターフェースと互換性あり。
- GW1NSR-4C/GW1NSER-4C デバイス QN48P パッケージをサポート。
- 8 ビットの PSRAM Width をサポート。
- 8 ビットの DQ bus Width をサポート。
- プログラム可能なバースト長 : 16、32、64、および 128。
- クロック比 1 : 2 をサポート。
- 初期遅延 6 をサポート。
- 固定遅延モードをサポート。
- 電源オフオプションをサポート。
- 構成可能なドライブ強度
- 構成可能なセルフリフレッシュ領域
- 構成可能なリフレッシュレート

16.2 レジスタの定義

HyperRAM Memory Interface レジスタの定義を表 16-1 に示します。
HyperRAM Memory Interface レジスタの定義は、
library¥middlewares¥hyper_ram¥hyper_ram.h にあります。

表 16-1 HyperRAM Memory Interface レジスタの定義

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
CMD	0x00	RW	1	0x0	Command register [0] Operation type 0 = Read operation 1 = Write operation
ADDRESS	0x04	RW	21	0x0	Address register [20:0] Address of reading and writing data
WR_DATA0	0x08	RW	32	0x0	Write data register 0 [31:0] Write first 32bit data
WR_DATA1	0x0C	RW	32	0x0	Write data register 1 [31:0] Write second 32bit data
WR_DATA2	0x10	RW	32	0x0	Write data register 2 [31:0] Write third 32bit data
WR_DATA3	0x14	RW	32	-	Write data register 3 [31:0] Write fourth 32bit data
CMD_EN	0x18	WO	1	-	Command enable register [0] Enable HyperRAM Memory
READ_DONE	0x1C	RW	1	-	Read status register [0] Read done flag, auto set 1 if it is done, and need MCU to clear
RD_DATA0	0x20	RO	32	-	Read data register 0 [31:0] Read first 32bit data
RD_DATA1	0x24	RO	32	-	Read data register 1 [31:0] Read second 32bit data
RD_DATA2	0x28	RO	32	-	Read data register 2 [31:0] Read third 32bit data
RD_DATA3	0x2C	RO	32	-	Read data register 3 [31:0] Read fourth 32bit data
INTI_DONE	0x30	RO	1	-	Initialization done register [0] HyperRAM Memory hardware initialization done flag 0 = Initialization failed 1 = Initialization done

16.3 ドライバーの使用法

HyperRAM Memory Interface ドライバーの使用を表 16-2 に示します。
HyperRAM Memory Interface ドライバーの定義は、
library¥middlewares¥hyper_ram¥hyper_ram.c にあります。

表 16-2 HyperRAM Memory Interface ドライバーの使用

名称	説明
HPRAM_Check_Init_Status	Check the status of HyperRAM Memory initialization
HPRAM_Mode_Set	Set the mode for HyperRAM Memory write and read
HPRAM_Address_Set	Set the address of HyperRAM Memory and save data into this address
HPRAM_Read_Data_Buff	Read data from the buffer of HyperRAM Memory
HPRAM_Cmd_Enable	Enable the command of HyperRAM Memory
HPRAM_Read_Done_Flag	Get the flag of read HyperRAM Memory done
HPRAM_Clear_Read_Done_Flag	Clear the flag of read HyperRAM Memory done
HPRAM_Write_Data_Buff	Write data into the buffer of HyperRAM Memory
HPRAM_Cmd_Unable	Disable the command of HyperRAM Memory
HPRAM_Write_Data_Package	Write a package data into HyperRAM Memory
HPRAM_Read_Data_Package	Read a package data from HyperRAM Memory

17 RTOS

Gowin_EMPU(GW1NS-4C)支持 uC/OS-III、FreeRTOS 和 RT-Thread Nano 版本的 RTOS。

17.1 uC/OS-III

17.1.1 特徴

- 管理可能なタスクの数に制限のない、拡張・ROM 化可能でプリエンブティブなリアルタイムコア
- リソース管理、同期、タスク間通信など、期待される機能を提供する第 3 世代のコア
- 実行時の動作性能を測定したり、信号やメッセージをタスクに直接送信したり、タスクが複数のセマフォとメッセージキューを同時に待機できるなど、他のリアルタイムコアにはない多くの特性を提供。
- Gowin_EMPU(GW1NS-4C)は uC/OS-III をサポート。
- uC/OS-III のソースコードは Micrium のホームページ <http://www.micrium.com> からダウンロード可能。

17.1.2 バージョン

Gowin_EMPU(GW1NS-4C)は、バージョン V3.03.00 の uC/OS-III をサポートしています。

17.1.3 構成

- UCOSIII_CONFIG ¥ os_cfg.h および os_cfg_app.h を変更することにより、uC/OS-III を構成することが可能。
- UCOS_BSP ¥ bsp.c および bsp.h を変更することにより、開発ボードをサポートすることが可能。

17.2 FreeRTOS

17.2.1 特徴

- 軽量のリアルタイム・オペレーティングシステム。
- タスク管理、時間管理、セマフォ、メッセージキュー、メモリ管理、ロギング機能、ソフトウェアタイマー、コルーチンなどの機能を備えており、基本的に小規模システムのニーズを満たすことが可能
- オープンソース、ポータブル、スケーラブル、および柔軟なスケジューリングを備えた完全に無料のオペレーティングシステム。
- Gowin_EMPU(GW1NS-4C)はFreeRTOSをサポート。
- FreeRTOSのソースコードはFreeRTOSのホームページ <http://www.FreeRTOS.org> からダウンロード可能。

17.2.2 バージョン

Gowin_EMPU(GW1NS-4C)は、バージョン V10.2.1 の FreeRTOS をサポートします。

17.2.3 構成

`include¥FreeRTOSConfig.h` を変更することにより、FreeRTOS を構成することができます。

17.3 RT-Thread Nano

17.3.1 特徴

- RT-Thread Nano は、C 言語で開発され、オブジェクト指向の、洗練されたハードリアルタイム・カーネルです。RT-Thread Nano はプリエンプティブなリアルタイム・マルチタスク RTOS で、簡単にカスタマイズできます。
-
- RT-Thread Nanod は、わずかなメモリ資源しか必要とせず、タスク処理、ソフトウェアタイマ、セマフォ、メールボックス、リアルタイム・スケジューリングなど、比較的完成度の高いリアルタイム OS としての機能を備えています。
- Apache ライセンス 2.0 に準拠しており、オープンソースで無料です。RTOS カーネルとすべてのオープンソース・コンポーネントは、アプリケーションのソースコードを公開する必要がなく、潜在的な商業的

リスクなしに商用製品で自由に使用することができます。

- Gowin_EMPU(GW1NS-4C)は、RT-Thread Nano をサポートします。
- RT-Thread Nano のソースコードは、RT-Thread のホームページ <https://www.rt-thread.org> からダウンロードできます。

17.3.2 バージョン

Gowin_EMPU(GW1NS-4C)は、バージョン V3.1.5 の RT-Thread Nano をサポートします。

17.3.3 構成

- `bsp¥cm3¥rtconfig.h` を変更することにより、RT-Thread Nano を構成することができます。
- `bsp¥cm3¥drivers¥board.c` を変更することにより、開発ボードをサポートするようにできます。

18 アプリケーションプログラム

Gowin_EMPU(GW1NS-4C)は、ARM Keil MDK IDE(テスト済みバージョン : V5.26)および GMD IDE(テスト済みバージョン : V2025.01)を使用したアプリケーションプログラムを提供します。アプリケーションプログラムのダウンロード URL :

[cdn.gowinsemi.com.cn/Gowin_EMPU\(GW1NS-4C\)_V2.1.zip](http://cdn.gowinsemi.com.cn/Gowin_EMPU(GW1NS-4C)_V2.1.zip)

18.1 UART

UART アプリケーションプログラム :

```
...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥printf
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥printf
...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥uart_rx
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥uart_rx
...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥uart_rx_intr
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥uart_rx_intr
```

18.2 Timer

Timer アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥time
r

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
timer

18.3 Watch Dog

Watch Dog アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥wdo
g

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
wdog

18.4 GPIO

GPIO アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥led

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
led

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥gpio
_in_intr

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
gpio_in_intr

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥key
scan

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
keyscan

18.5 I²C Master

I²C Master アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥i2c_
master

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
i2c_master

18.6 SPI Master

SPI Master アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥spi_
master

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
spi_master

18.7 RTC

RTC アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥rtc

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
rtc

18.8 SPI-Flash Memory

SPI-Flash Memory アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥spi_
flash

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
spi_flash

18.9 PSRAM Memory

PSRAM Memory アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥psram

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
psram

18.10 HyperRAM Memory

HyperRAM Memory アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥hyper_ram

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
hyper_ram

18.11 Interrupt

Interrupt アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥interrupt

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
interrupt

18.12 DMM

Dynamic Memory Management アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥dmm

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
dmm
```

18.13 AHB Master

AHB Master アプリケーションプログラム :

...

```
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥ahb
2_master
```

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
ahb2_master
```

18.14 APB Master

APB Master アプリケーションプログラム :

...

```
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥project¥apb
2_master
```

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥project¥
apb2_master
```

18.15 User Interrupt

User Interrupt アプリケーションプログラム :

```
...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥proje
ct¥user_intr
```

```
...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥p
roject¥user_intr
```

18.16 FIFO

FIFO アプリケーションプログラム :

```
...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_demo¥projec
t¥fifo
```

```
...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_demo¥src¥p
roject¥fifo
```

18.17 Code Coverage

Code Coverage 機能アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_gcov_demo

18.18 Profiling

Profiling 機能アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_gprof_demo

18.19 uC/OS-III

RTOS uC/OS-III アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_ucos_iii

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_ucos_iii

18.20 FreeRTOS

RTOS FreeRTOS アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_freertos

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_freertos

18.21 RT-Thread Nano

RT-Thread Nano アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm3_rtthread_nano

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm3_rtthread_nano

