

Gowin_EMPU_M1 ソフトウェアプログラミング リファレンスマニュアル

IPUG533-2.3J, 2025-08-08

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
2019/02/19	1.0J	初版。
2019/07/18	1.1J	MCU ソフトウェア設計では、周辺機器 CAN、Ethernet、SPI-Flash、RTC、DualTimer、TRNG、I2C、SPI、SD-Card の拡張をサポート。
2019/08/18	1.2J	● MCU ハードウェア設計とソフトウェア設計では、DDR3 Memory の拡張をサポート。 ● 既知の ITCM、DTCM Size、および IDE の問題を修正。
2019/09/27	1.3J	● MCU ハードウェア設計およびソフトウェア設計では、オフチップ SPI-Flash の読み出し、書き込み、および消去をサポート。 ● MCU ソフトウェア設計は、外部デバイス I2C の連続したマルチバイト読み出しおよび書き込みをサポート。 ● 既知の MCU ソフトウェア設計の AHB2 拡張インターフェースと APB2 拡張インターフェースのアドレスマッピングの問題を修正。 ● 既知の MCU ソフトウェア設計の DDR3 Memory の連続した読み出し/書き込みの問題を修正。
2020/01/16	1.4J	● MCU ハードウェア設計とソフトウェア設計では、周辺機器 PSRAM の拡張をサポート。 ● MCU コンパイラ GMD V1.0 を更新。 ● RTOS リファレンス・デザインを更新。 ● AHB2 および APB2 拡張バスインターフェースハードウェアおよびソフトウェアのリファレンス・デザインを追加。
2020/03/05	1.5J	MCU ソフトウェア設計での SD-Card の読み出しと書き込みをサポート。
2020/06/12	1.6J	● MCU が外部命令メモリをサポート。 ● MCU が外部データメモリをサポート。 ● AHB バスインターフェースを 6 個拡張。 ● APB バスインターフェースを 16 個拡張。 ● GPIO が複数のインターフェースタイプをサポート。 ● I2C が複数のインターフェースタイプをサポート。
2021/07/16	1.7J	● 既知の SPI 全二重読み出しおよび書き込みの問題を修正。 ● MCU ソフトウェアのバージョン情報を更新。
2023/05/11	1.8J	● 拡張外部割り込み入力をサポートを追加。 ● Arora V FPGA 製品のサポートを追加。 ● ソフトウェアプログラミングの開発ライブラリを更新。 ● Bootloader を更新。 ● ソフトウェア・リファレンス・デザインを更新。 ● RT-Thread Nano のソフトウェア・リファレンス・デザインを追加。 ● TCP/IP プロトコルスタックのソフトウェア・リファレンス・デザインを追加。

日付	バージョン	説明
2023/07/21	1.9J	ソフトウェアのバージョン情報を更新。
2024/03/07	2.0J	● ソフトウェアプログラミング・ライブラリを更新。 ● ソフトウェア・リファレンス・デザインを更新。
2024/07/12	2.1J	● SPI Flash Memory ドライバー関数を更新。 ● QSPI Flash Memory ドライバー関数を更新。
2025/01/17	2.2J	● GPIO 周辺機器ドライバーの割り込み初期化の問題を修正。 ● ソフトウェア・リファレンス・デザインを更新。
2025/08/08	2.3J	● GMD ソフトウェアのバージョン情報を更新。 ● GMD の 2025.01 バージョンに適合するようにソフトウェア・リファレンス・デザインを更新。 ● QSPI Flash Memory のドライバー関数を更新。 ● FAT32、32GB の SD-Card をサポートする SD_Host_v2 と SD_SPI_IF_v1 ソリューションを追加。

目次

目次	v
図一覧.....	ix
表一覧.....	x
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 組込み RTOS ソフトウェアプログラミング	3
1.3 プロトコルスタック・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	9
4.1 特徴.....	9
4.2 レジスタの定義.....	10
4.3 初期化の定義	10
4.4 ドライバーの使用方法	11
5 Timer	12
5.1 特徴.....	12
5.2 レジスタの定義.....	12
5.3 初期化の定義	13
5.4 ドライバーの使用方法	13
6 Watch Dog	14
6.1 特徴.....	14
6.2 レジスタの定義.....	14
6.3 初期化の定義	15
6.4 ドライバーの使用方法	16
7 GPIO	17
7.1 特徴.....	17

7.2 レジスタの定義.....	17
7.3 初期化の定義	20
7.4 ドライバーの使用方法.....	21
8 I²C Master.....	22
8.1 特徴.....	22
8.2 レジスタの定義.....	22
8.3 ドライバーの使用方法.....	23
9 SPI Master.....	25
9.1 特徴.....	25
9.2 レジスタの定義.....	25
9.3 初期化の定義	26
9.4 ドライバーの使用方法.....	26
10 RTC.....	28
10.1 特徴.....	28
10.2 レジスタの定義.....	29
10.3 ドライバーの使用方法.....	30
11 TRNG.....	31
11.1 特徴	31
11.2 レジスタの定義.....	31
11.3 ドライバーの使用方法	34
12 DualTimer.....	35
12.1 特徴.....	35
12.2 レジスタの定義.....	35
12.3 ドライバーの使用方法.....	37
13 SD-Card.....	39
13.1 特徴.....	39
13.2 レジスタの定義.....	39
13.2.1 SD-Card 周辺機器	39
13.2.2 SD_Host_v2.....	42
13.2.3 SD_SPI_IF_v1	42
13.3 ドライバーの使用方法.....	43
13.3.1 SD-Card 周辺機器	43
13.3.2 SD_Host_v2.....	43
13.3.3 SD_SPI_IF_v1	43
14 CAN	45
14.1 特徴.....	45
14.2 レジスタの定義.....	45

14.3 ドライバーの使用方法	50
15 Ethernet	53
15.1 特徴	53
15.2 レジスタの定義	53
15.3 ドライバーの使用方法	55
16 DDR3 Memory.....	57
16.1 特徴	57
16.2 レジスタの定義	57
16.3 ドライバーの使用方法	58
17 SPI-Flash Memory	59
17.1 特徴	59
17.2 レジスタの定義	59
17.3 ドライバーの使用方法	64
17.3.1 QSPI-Flash ドライバー	64
17.3.2 SPI-Flash ドライバー	66
18 PSRAM Memory	67
18.1 特徴	67
18.2 レジスタの定義	67
18.3 ドライバーの使用方法	68
19 RTOS	70
19.1 uC/OS-III	70
19.1.1 特徴	70
19.1.2 バージョン	70
19.1.3 構成	70
19.2 FreeRTOS.....	70
19.2.1 特徴	70
19.2.2 バージョン	71
19.2.3 構成	71
19.3 RT-Thread Nano	71
19.3.1 特徴	71
19.3.2 バージョン	71
19.3.3 構成	71
20 プロトコルスタック・ソフトウェアプログラミング	73
20.1 TCP/IP プロトコルスタック	73
20.1.1 特徴	73
20.1.2 バージョン	74

21 アプリケーションプログラム	75
21.1 UART	75
21.2 Timer	75
21.3 Watch Dog	76
21.4 GPIO	76
21.5 I ² C Master	76
21.6 SPI Master	77
21.7 RTC.....	77
21.8 TRNG.....	77
21.9 DualTimer	77
21.10 SD-Card.....	77
21.11 CAN.....	78
21.12 Ethernet	78
21.13 DDR3 Memory	78
21.14 SPI-Flash Memory.....	78
21.15 QSPI-Flash Memory	78
21.16 PSRAM Memory	79
21.17 Interrupt	79
21.18 DMM	79
21.19 AHB Master.....	79
21.20 APB Master.....	80
21.21 uC/OS-III.....	80
21.22 FreeRTOS.....	80
21.23 RT-Thread Nano	80
21.24 Embedded Memory	80
21.25 External Memory.....	80
21.26 Make Hex.....	80
21.27 Merge Bit	81
21.28 LwIP プロトコルスタック	81
21.29 uIP プロトコルスタック	81

図一覧

図 4-1 UART の構造	9
図 5-1 Timer の構造	12
図 6-1 Watch Dog の動作フロー	14
図 7-1 GPIO の構造	17
図 10-1 RTC の構造	29

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 割り込みベクターテーブル	7
表 4-1 UART レジスタの定義	10
表 4-2 UART 初期化の定義	11
表 4-3 UART ドライバーの使用方法	11
表 5-1 Timer レジスタの定義	12
表 5-2 Timer 初期化の定義	13
表 5-3 Timer ドライバーの使用方法	13
表 6-1 Watch Dog レジスタの定義	15
表 6-2 Watch Dog 初期化の定義	15
表 6-3 Watch Dog ドライバーの使用方法	16
表 7-1 GPIO レジスタの定義	17
表 7-2 GPIO 初期化の定義	20
表 7-3 GPIO ドライバーの使用方法	21
表 8-1 I ² C Master レジスタの定義	22
表 8-2 I ² C Master ドライバーの使用	23
表 9-1 SPI Master レジスタの定義	25
表 9-2 SPI Master 初期化の定義	26
表 9-3 SPI Master ドライバーの使用方法	27
表 10-1 RTC レジスタの定義	29
表 10-2 RTC ドライバーの使用方法	30
表 11-1 TRNG レジスタの定義	31
表 11-2 TRNG ドライバーの使用方法	34
表 12-1 DualTimer レジスタの定義	35
表 12-2 DualTimer ドライバーの使用方法	37
表 13-1 SD-Card レジスタの定義	40
表 13-2 SD-Card ドライバーの使用方法	43

表 14-1 CAN レジスタの定義.....	45
表 14-2 CAN ドライバーの使用方法.....	50
表 15-1 Ethernet レジスタの定義.....	53
表 15-2 Ethernet ドライバーの使用方法.....	55
表 16-1 DDR3 Memory レジスタの定義	57
表 16-2 DDR3 Memory ドライバーの使用方法	58
表 17-1 SPI-Flash Memory レジスタの定義	59
表 17-2 QSPI-Flash Memory ドライバーの使用方法.....	64
表 17-3 SPI-Flash Memory ドライバーの使用方法	66
表 18-1 PSRAM Memory レジスタの定義	67
表 18-2 PSRAM Memory ドライバーの使用方法	68

1 ソフトウェアプログラミング・ライブラリ

Gowin_EMPU_M1 のソフトウェアプログラミング・ライブラリ : ...
¥library。

ダウンロード URL :

cdn.gowinsemi.com.cn/Gowin_EMPU_M1.zip

Gowin_EMPU_M1 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
- プロトコルスタック・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_M1 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

関数ライブラリファイル	説明
システムの定義	
startup_GOWIN_M1.s	Cortex-M1コアのスタートアッププログラム
core_cm1.h	Cortex-M1コアレジスタの定義
GOWIN_M1.h	割り込みベクターテーブル、外部レジスタ、およびアドレスマッピングの定義
system_GOWIN_M1.c system_GOWIN_M1.h	Cortex-M1コアシステム初期化とシステムクロックの定義
Linkerの定義	
GOWIN_M1_flash_burn.ld GOWIN_M1_flash_xip.ld	GMD Flashリンカー・スクリプト・ファイル : burn : Flashによる起動、ITCMで実行。 xip : Flashによる起動および実行。
周辺機器ドライバーの定義	
GOWIN_M1_gpio.c	GPIO ドライバー関数の定義

関数ライブラリファイル	説明
GOWIN_M1_gpio.h	
GOWIN_M1_can.c GOWIN_M1_can.h	CANドライバー関数の定義
GOWIN_M1_ethernet.c GOWIN_M1_ethernet.h	Ethernetドライバー関数の定義
GOWIN_M1_ddr3.c GOWIN_M1_ddr3.h	DDR3 Memoryドライバー関数の定義
GOWIN_M1_psram.c GOWIN_M1_psram.h	PSRAM Memoryドライバー関数の定義
GOWIN_M1_spi_flash.c GOWIN_M1_spi_flash.h	SPI-Flash Memoryの読み出し、書き込み、および消去のドライバー関数の定義
GOWIN_M1_qspi_flash.c GOWIN_M1_qspi_flash.h	QSPI-Flash Memoryドライバー関数の定義
GOWIN_M1_timer.c GOWIN_M1_timer.h	Timerドライバー関数の定義
GOWIN_M1_wdog.c GOWIN_M1_wdog.h	Watch Dogドライバー関数の定義
GOWIN_M1_uart.c GOWIN_M1_uart.h	UARTドライバー関数の定義
GOWIN_M1_rtc.c GOWIN_M1_rtc.h	RTCドライバー関数の定義
GOWIN_M1_trng.c GOWIN_M1_trng.h	TRNGドライバー関数の定義
GOWIN_M1_dualtimer.c GOWIN_M1_dualtimer.h	DualTimerドライバー関数の定義
GOWIN_M1_i2c.c GOWIN_M1_i2c.h	I2C Masterドライバー関数の定義
GOWIN_M1_spi.c GOWIN_M1_spi.h	SPI Masterドライバー関数の定義
GOWIN_M1_sdcard.c GOWIN_M1_sdcard.h	SD-Cardドライバー関数の定義
GOWIN_M1_misc.c GOWIN_M1_misc.h	割り込み優先度管理およびSysTickの定義
GOWIN_M1_it.c GOWIN_M1_it.h	割り込み処理関数の定義
ミドルウェアの定義	
retarget.c uart.c uart.h	UART printf関数の再定義 UART初期化の定義
malloc.c malloc.h	動的メモリ管理
gpio.c gpio.h	GPIO初期化の定義

関数ライブラリファイル	説明
delay.c delay.h	遅延関数の定義
ソリューション	
apb_sd_host.h apb_sd_host_driver.h apb_sd_host_driver.c	ソリューション「SD_Host_v2」 APB SD Hostレジスタ、ドライバー関数の定義
spi_sdcard.h spi_sdcard.c	ソリューション「SD_SPI_IF_v1」 SPI Masterインターフェース経由でSD-Cardにアクセスするためのドライバー関数の定義

1.2 組込み RTOS ソフトウェアプログラミング

Gowin_EMPU_M1 は、次の組込み RTOS ソフトウェアプログラミングをサポートします。

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

1.3 プロトコルスタック・ソフトウェアプログラミング

Gowin_EMPU_M1 は、イーサネット TCP/IP プロトコルスタック・ソフトウェアプログラミングをサポートしています。オープンソースの LwIP プロトコルスタックと uIP プロトコルスタックが提供されています。

LwIP プロトコルスタックは、オペレーティングシステムの有無にかかわらず実行できる小規模なオープンソース TCP/IP プロトコルスタックで、TCP プロトコルの主な機能を維持しながらメモリ使用量を削減できます。

uIP プロトコルスタックは、オペレーティングシステムのサポートを必要としないシンプルな組み込みネットワークプロトコルスタックであり、イベント駆動型でプログラムされ、ARP、IP、ICMP、TCP の 4 つの基本プロトコルを実装します。

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_M1 標準周辺機器のメモリマッピングの定義を表 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
ITCM	-	0x00000000	1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB, 512KB
DTCM	-	0x20000000	1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB, 512KB
External Instruction Memory	-	0x00000000	外部命令メモリ
External Data Memory	-	0x20100000	外部データメモリ
TIMER0	TIMER_TypeDef	0x50000000	タイマー0 (Timer0)
TIMER1	TIMER_TypeDef	0x50001000	タイマー1 (Timer1)
UART0	UART_TypeDef	0x50004000	汎用非同期送受信回路0(UART0)
UART1	UART_TypeDef	0x50005000	汎用非同期送受信回路1(UART1)
Watch Dog	WDOG_TypeDef	0x50008000	ウォッチドッグ (Watch Dog)
RTC	RTC_RegDef	0x50006000	リアルタイムクロック (RTC)
TRNG	TRNG_RegDef	0x5000F000	真性乱数生成器 (TRNG)
DualTimer	DUALTIMER_RegDef	0x50002000	デュアルタイマー (DualTimer)
SPI_FLASH	SPI_FLASH_RegDef	0x50003000	SPI Flashメモリ (SPI-FlashとQSPI-Flash Memory)
I2C	I2C_TypeDef	0x5000A000	I ² Cマスター (I ² C Master)
SPI	SPI_TypeDef	0x5000B000	SPIマスター (SPI Master)
SD-Card	SDCard_TypeDef	0x50009000	SDカード (SD-Card)
GPIO0	GPIO_TypeDef	0x40000000	汎用入出力ポート (GPIO)
CAN	CAN_RegDef	0x45000000	コントローラ・エリア・ネットワ

標準周辺機器	タイプ	アドレスマッピング	説明
			ーク (CAN)
Ethernet	ETH_RegDef	0x46000000	イーサネット(Ethernet)
DDR3	DDR3_RegDef	0x88000000	DDR3メモリ (DDR3 Memory)
PSRAM	PSRAM_TypeDef	0x82000000	擬似SRAM (PSRAM Memory)
APB Master [1]	-	0x60000000	拡張APB Master [1]
APB Master [2]	-	0x60100000	拡張APB Master [2]
APB Master [3]	-	0x60200000	拡張APB Master [3]
APB Master [4]	-	0x60300000	拡張APB Master [4]
APB Master [5]	-	0x60400000	拡張APB Master [5]
APB Master [6]	-	0x60500000	拡張APB Master [6]
APB Master [7]	-	0x60600000	拡張APB Master [7]
APB Master [8]	-	0x60700000	拡張APB Master [8]
APB Master [9]	-	0x60800000	拡張APB Master [9]
APB Master [10]	-	0x60900000	拡張APB Master [10]
APB Master [11]	-	0x60A00000	拡張APB Master [11]
APB Master [12]	-	0x60B00000	拡張APB Master [12]
APB Master [13]	-	0x60C00000	拡張APB Master [13]
APB Master [14]	-	0x60D00000	拡張APB Master [14]
APB Master [15]	-	0x60E00000	拡張APB Master [15]
APB Master [16]	-	0x60F00000	拡張APB Master [16]
AHB Master [1]	-	0x80000000	拡張AHB Master [1]
AHB Master [2]	-	0x81000000	拡張AHB Master [2]
AHB Master [3]	-	0x86000000	拡張AHB Master [3]
AHB Master [4]	-	0x89000000	拡張AHB Master [4]
AHB Master [5]	-	0x8A000000	拡張AHB Master [5]
AHB Master [6]	-	0x8B000000	拡張AHB Master [6]

2.2 コアシステムのメモリマッピング

Cortex-M1 コアシステムの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
SysTick	SysTick_Type	0xE000E010	SysTick configuration struct
NVIC	NVIC_BASE	0xE000E100	NVIC configuration struct
SCnSCB	SCnSCB_Type	0xE000E000	System control Register not in SCB

システム制御	タイプ	アドレスマッピング	説明
SCB	SCB_Type	0xE000ED00	SCB configuration struct

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(1、8、16、32 を利用可能)の割り込み処理信号を提供。
- 4 つの拡張外部割り込み信号を提供。
- 0～3 のプログラマブル割り込み優先度をサポート。

Cortex-M1 割り込みベクターテーブルを表 3-1 に示します。

表 3-1 割り込みベクターテーブル

割り込みベクターアドレス	割り込みタイプ	割り込み番号	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	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

割り込みベクターアドレス	割り込みタイプ	割り込み番号	説明
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	EXTINT_0_Handler	10	16+10: External 0 Handler
0x0000006C	DTimer_Handler	11	16+11: DualTimer Handler
0x00000070	TRNG_Handler	12	16+12: TRNG Handler
0x00000074	EXTINT_1_Handler	13	16+13: External 1 Handler
0x00000078	EXTINT_2_Handler	14	16+14: External 2 Handler
0x0000007C	EXTINT_3_Handler	15	16+15: External 3 Handler
0x00000080	GPIO0_0_Handler	16	16+16: GPIO0_0 Handler
0x00000084	GPIO0_1_Handler	17	16+17: GPIO0_1 Handler
0x00000088	GPIO0_2_Handler	18	16+18: GPIO0_2 Handler
0x0000008C	GPIO0_3_Handler	19	16+19: GPIO0_3 Handler
0x00000090	GPIO0_4_Handler	20	16+20: GPIO0_4 Handler
0x00000094	GPIO0_5_Handler	21	16+21: GPIO0_5 Handler
0x00000098	GPIO0_6_Handler	22	16+22: GPIO0_6 Handler
0x0000009C	GPIO0_7_Handler	23	16+23: GPIO0_7 Handler
0x000000A0	GPIO0_8_Handler	24	16+24: GPIO0_8 Handler
0x000000A4	GPIO0_9_Handler	25	16+25: GPIO0_9 Handler
0x000000A8	GPIO0_10_Handler	26	16+26: GPIO0_10 Handler
0x000000AC	GPIO0_11_Handler	27	16+27: GPIO0_11 Handler
0x000000B0	GPIO0_12_Handler	28	16+28: GPIO0_12 Handler
0x000000B4	GPIO0_13_Handler	29	16+29: GPIO0_13 Handler
0x000000B8	GPIO0_14_Handler	30	16+30: GPIO0_14 Handler
0x000000BC	GPIO0_15_Handler	31	16+31: GPIO0_15 Handler

4UART

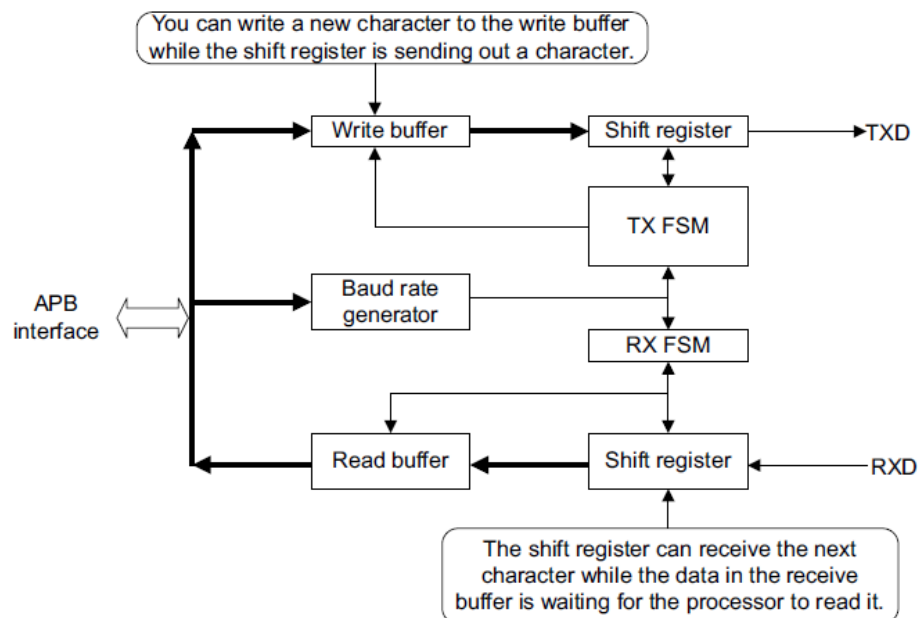
4.1 特徴

Gowin_EMPU_M1 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。

- 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¥cm1¥device_support¥GOWIN_M1.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 [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	0x0000 0	[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¥GOWIN_M1_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 ドライバーの使用方法

UART ドライバーの使用を表 4-3 に示します。UART ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_uart.c にあります。

表 4-3 UART ドライバーの使用方法

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

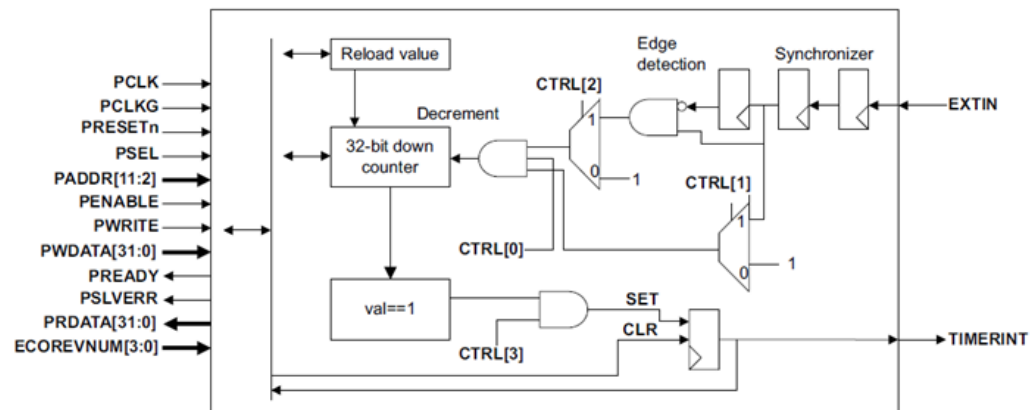
5.1 特徴

Gowin_EMPU_M1 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

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¥cm1¥device_support¥GOWIN_M1.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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					clock [1] Select external input as enable [0] Enable
VALUE	0x004	RW	32	0x0000 0000	[31:0] Current value
RELOAD	0x008	RW	32	0x0000 0000	[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¥GOWIN_M1_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 ドライバーの使用を表 5-3 に示します。Timer ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_timer.c にあります。

表 5-3 Timer ドライバーの使用方法

名称	説明
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

6 Watch Dog

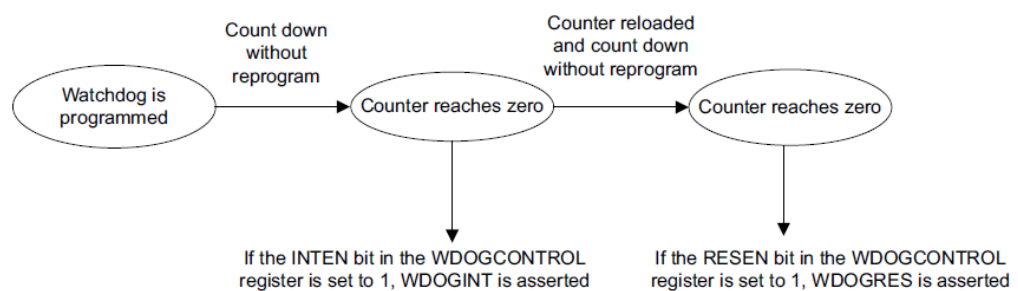
6.1 特徴

Gowin_EMPU_M1 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。
- 割り込みを監視し、カウンタが 0 までデクリメントされると、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¥cm1¥device_support¥GOWIN_M1.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	0xFFFF FFFF	The value from which the counter is to decrement
VALUE	0x04	RO	32	0xFFFF FFFF	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	0x0000 0000	[32:1] Enable register writes [0] Register write enable status
RESERVED	0xF00- 0xC00	–	–	–	Reserved
ITCR	0xF00	RW	1	0x0	Integration test mode enable
ITOP	0xF04	WO	2	0x0	[1] Integration test WDOGRES value [0] Integration test WDOGIN T 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¥GOWIN_M1_wdog.h にあります。

表 6-2 Watch Dog 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	WDOGIN_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¥src¥GOWIN_M1_wdog.c にあります。

表 6-3 Watch Dog ドライバーの使用方法

名称	説明
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

7 GPIO

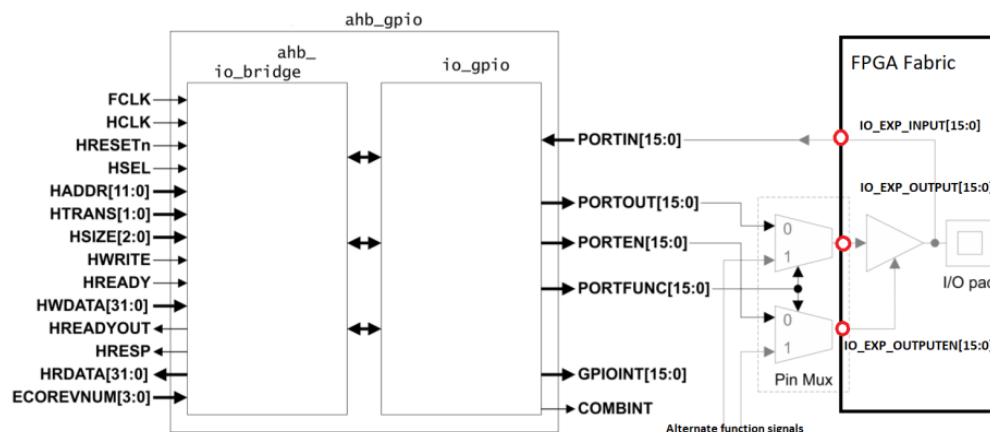
7.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの 16 ビット入出力インターフェースの GPIO 周辺機器が含まれています：

- FPGA Fabric に接続
- 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¥cm1¥device_support¥GOWIN_M1.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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					Write to data output register Read back value goes through double flip-flop Synchronize logic with delay 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 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 Write 1 to set the enable bit Write 0 no effect Read back 0 indicates interrupt disabled 1 indicates interrupt enabled.

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
MASKLOW BYTE	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
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¥GOWIN_M1_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 GPIO_Pin_15	16 bits GPIO Pins
GPIO_Mode	GPIO_Mode_TypeDef	GPIO_Mode_IN GPIO_Mode_OUT GPIO_Mode_AF	16 bits GPIO Pins mode

名称	タイプ	値	説明
GPIO_Int	GPIOInt_TypeDef	GPIO_Int_Disable GPIO_Int_Low_Level GPIO_Int_High_Level GPIO_Int_Falling_Edge GPIO_Int_Rising_Edge	16 bits GPIO Pins interrupt

7.4 ドライバーの使用方法

GPIO ドライバーの使用を表 7-3 に示します。GPIO ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_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 I²C Master

8.1 特徴

Gowin_EMPU_M1 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
- バスアービトラージョンおよびアービトラージョンロストを検出
- 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

8.2 レジスタの定義

I²C Master レジスタの定義を表 8-1 に示します。I²C Master レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 8-1 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	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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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 [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 ドライバーの使用方法

I²C Master ドライバーの使用方法を表 8-2 に示します。I²C Master ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_i2c.c にあります。

表 8-2 I²C Master ドライバーの使用

名称	説明
I2C_Init	I2C Initialization
I2C_SendByte	Send a byte to I2C bus
I2C_SendBytes	Send multiple bytes to I2C bus
I2C_SendData	Send multiple bytes to I2C bus once time
I2C_ReceiveByte	Read a byte from I2C bus
I2C_ReadBytes	Read multiple bytes from I2C bus
I2C_ReceiveData	Read multiple bytes from I2C bus once time
I2C_Rate_Set	Set I2C traffic rate
I2C_Enable	Enable I2C bus

名称	説明
I2C_UnEnable	Disable I2C bus
I2C_InterruptOpen	Open I2C interrupt
I2C_InterruptClose	Close I2C interrupt

9 SPI Master

9.1 特徴

Gowin_EMPU_M1 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¥cm1¥device_support¥GOWIN_M1.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	32	0x00000000	Read data register [31:8] Reserved [7:0] Read data
WDATA	0x04	WO	32	0x00000000	Write data register [31:8] Reserved [7:0] Write data
STATUS	0x08	RW	32	0x00000000	[31:8] Reserved [7] Overflow error status [6] Receive ready status [5] Transmit ready status [4] Be transmitting

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[3] Transmit overrun error status [2] Receive overrun error status [1:0] Reserved
SSMASK	0x0C	RW	32	0x00000000	[31:1] Reserved [0] Select and enable slave
CTRL	0x10	RW	32	0x00000000	[31:5] Reserved [4:3] Clock selected, CLK_I / 2/4/6/8 [2] Clock polarity [1] Clock phase [0] Direction, 1 is MSB first

9.3 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¥GOWIN_M1_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

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¥src¥GOWIN_M1_spi.c にあります。

表 9-3 SPI Master ドライバーの使用方法

名称	説明
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_ReadWriteByte	Full duplex read and write a byte
SPI_WriteData	Writes data
SPI_ReadData	Reads data
SPI_Select_Slave	Select slave

10_{RTC}

10.1 特徴

Gowin_EMPU_M1 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。

- 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 RTCSRCCLK 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 0xFFFFFFFF 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。

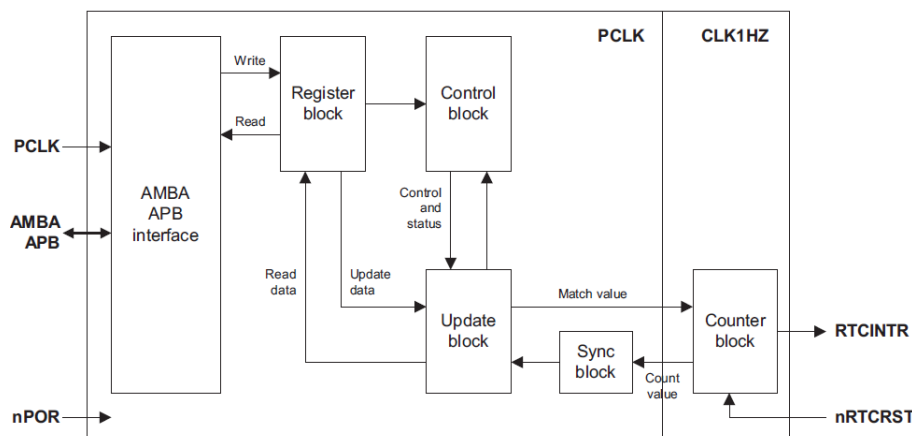
Load レジスタ RTC_LOAD_VALUE に書き込むことで RTC をロード・更新します。

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 10-1 に示します。

図 10-1 RTC の構造



10.2 レジスタの定義

RTC レジスタの定義を表 10-1 に示します。RTC レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 10-1 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_COUNTER_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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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

10.3 ドライバーの使用方法

RTC ドライバーの使用方法を表 10-2 に示します。RTC ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_rtc.c にあります。

表 10-2 RTC ドライバーの使用方法

名称	説明
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

11 TRNG

11.1 特徴

Gowin_EMPU_M1 には、APB バスを介してアクセスされる 1 つの 32 ビット TRNG 周辺機器が含まれています。

- デジタルロジックは、真の乱数ビットストリームを生成および取得。
- デジタルインバータチェーンに基づく内部エントロピーソースを含む。
- MCU コアが 200MHz で動作する場合、10K bits/s のエントロピーを生成。
- APB バスインターフェース。

11.2 レジスタの定義

TRNG レジスタの定義を表 11-1 に示します。TRNG レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 11-1 TRNG 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
RESERVE1	0x000-0x0FC	-	-	-	Reserved
RNG_IMR	0x100	RW	32	0x0000000F	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	0x00000000	Interrupt status register [31:4] Reserved

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[3] A Von Neumann error [2] A Continuous Random Number Generation Testing (CRNGT) error [1] The Autocorrelation test failed four times 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	0x0000 0000	Interrupt clear register [31:4] Reserved [3] Clear a Von Neumann error [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	0x0000 0000	Configuration register [31:2] Reserved [1:0] Selects the number of inverters: 00 = Selects the shortest inverter chain length 01 = Selects the short inverter chain length 10 = Selects the long inverter chain length 11 = Selects the longest inverter chain length
TRNG_VALID	0x110	RO	32	0x0000 0000	Valid register [31:1] Reserved [0] TRNG is complete, data can be read from EHR_DATA[0-5]
EHR_DATA0	0x114— x128	RO	32	0x0000 0000	Entropy holding register data register Return 32 bits from the 192-bit EHR DATA0 returns bit[31:0] DATA1 returns bit[63:32] DATA2 returns bit[95:64] DATA3 returns bit[127:96]
EHR_DATA1					
EHR_DATA2					
EHR_DATA3					
EHR_DATA4					
EHR_DATA5					

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					DATA4 returns bit[159:128] DATA5 returns bit[191:160]
RND_SOURCE_ENABLE	0x12C	RW	32	0x0000 0000	Random source enable register [31:1] Reserved [0] 1 = enable entropy source; 0 = disable entropy source
SAMPLE_CNT1	0x130	RW	32	0x0000 FFFF	Sample count register [31:0] Sets the number of rng_clk cycles
AUTOCORR_STATISTIC	0x134	RW	32	0x0000 0000	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	0x0000 0000	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	0x0000 0000	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	0x0000 0000	Busy register [31:1] Reserved [0] Reflects the status of rng_busy signal
RST_BIT_COUNTER	0x1BC	WO	32	0x0000 0000	Reset bit counter register [31:1] Reserved [0] Write any value to this bit resets the bits counter and TRNG valid registers
RESERVE4	0x1C0-	-	-	-	Reserved

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
	0x1DC				
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 ドライバーの使用方法

TRNG ドライバーの使用を表 11-2 に示します。TRNG ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_trng.c にあります。

表 11-2 TRNG ドライバーの使用方法

名称	説明
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

12DualTimer

12.1 特徴

Gowin_EMPU_M1 には、APB バスを介してアクセスされる 1 つの 32 ビットと 16 ビット DualTimer 周辺機器が含まれています：

- APB バスインターフェース
- 2 つのプログラム可能な 32 ビットまたは 16 ビットのダウンカウンタを含む
- ダウンカウンタが 0 にカウントダウンする時に割り込みを生成

12.2 レジスタの定義

DualTimer レジスタの定義を表 12-1 に示します。DualTimer レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 12-1 DualTimer 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
TIMER1LOAD	0x00	RW	32	0x0000 0000	Timer1 load register [31:0] Timer1 load value
TIMER1VALUE	0x04	RO	32	0xFFFF FFFF	Timer1 current value register [31:0] Timer1 current value
TIMER1CONTROL	0x08	RW	32	0x0000 0020	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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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
TIMER1INTCLR	0x0C	WO	-	-	Timer1 interrupt clear register Any write clears the interrupt output of the counter
TIMER1RIS	0x10	RO	32	0x0000 0000	Timer1 raw interrupt status register [31:1] Reserved [0] Raw interrupt status from the counter
TIMER1MIS	0x14	RO	32	0x0000 0000	Timer1 interrupt status register [31:1] Reserved [0] Enable interrupt status from the counter
TIMER1BGL OAD	0x18	RW	32	0x0000 0000	Timer1 background load register [31:0] The value used to reload the counter
RESERVE1	-	-	-	-	Reserved
TIMER2LOAD	0x00	RW	32	0x0000 0000	Timer2 load register [31:0] Timer2 load value
TIMER2VALUE	0x04	RO	32	0xFFFF FFFF	Timer2 current value register [31:0] Timer2 current value
TIMER2CONTROL	0x08	RW	32	0x0000 0020	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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[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	0x0000 0000	Timer2 raw interrupt status register [31:1] Reserved [0] Raw interrupt status from the counter
TIMER2MIS	0x14	RO	32	0x0000 0000	Timer2 interrupt status register [31:1] Reserved [0] Enable interrupt status from the counter
TIMER2BGLOAD	0x18	RW	32	0x0000 0000	Timer2 background load register [31:0] The value used to reload the counter

12.3 ドライバーの使用方法

DualTimer ドライバーの使用を表 12-2 に示します。DualTimer ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_dualtimer.c にあります。

表 12-2 DualTimer ドライバーの使用方法

名称	説明
DUALTIMER1_Init	Initialized DualTimer1
DUALTIMER2_Init	Intialized DualTimer2
Clear_DUALTIMER_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

名称	説明
Get_DULATIMER_interrupt_num	Get timer ID of DualTimer1 or DualTimer2

13_{SD-Card}

13.1 特徴

Gowin_EMPU_M1 には、APB バスを介してアクセスされる 1 つの SD-Card 周辺機器が含まれています。

- SD/MMC フォーマットのメモリカードをサポート
- メモリカードの初期化をサポート
- シンプルな SPI バスアクセス
- ブロック読み出しとブロック書き込みをサポート
- 512 バイトの受信および送信バッファを提供
- APB バスインターフェース
- APB バスインターフェースと SPI コアロジックの独立したクロック
- SD/MMC メモリカードの最大レートに近いデータ転送速度
- FAT16 ファイルシステムを採用する 4GB 以内の SD-Card のみをサポート

Gowin_EMPU_M1 は、FAT32、32GB の SD-Card をサポートする SD_Host_v2 と SD_SPI_IF_v1 ソリューションを提供しています。

すべての SD-Card は、初めて使用する前にフォーマットする必要があります。ことに注意してください。

13.2 レジスタの定義

13.2.1 SD-Card 周辺機器

Gowin_EMPU_M1 システムには、FAT16、4GB SD-Card コントローラ周辺機器が組み込まれています。

SD-Card レジスタの定義を表 13-1 に示します。SD-Card レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 13-1 SD-Card 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
SPI_MASTER_VERSION	0x000	RW	8	0x00	SPI master version register [7:4] Major revision number [3:0] Minor revision number
SPI_MASTER_CONTROL	0x001	WO	8	0x00	SPI master control register [7:1] Reserved [0] Reset core logic and register 1 = Reset core logic and register, self clearing
TRANS_TYPE	0x002	RW	8	0x00	Transaction type register [7:2] Reserved [1:0] Set the transaction type 00 = Direct access 01 = Initialized SD 10 = Read SD block 11 = Write SD block
TRANS_CTRL	0x003	WO	8	0x00	Transaction control register [7:1] Reserved [0] Start transaction 1 = Start transaction, self clearing
TRANS_STS	0x004	RO	8	-	Transaction status register [7:1] Reserved [0] Transaction busy 1 = Transaction busy
TRANS_ERROR	0x005	RO	8	-	Transaction error register [7:6] Reserved [5:4] SD write error 00 = Write no error 01 = Write command error 10 = Write data error 11 = Write busy error [3:2] SD read error 00 = Read no error 01 = Read command error 10 = Read token error [1:0] SD initialize error 00 = Initialize no error 01 = Initialize command 0 error 10 = Initialize command 1 error
DIRECT_ACCES	0x006	RW	8	0x00 /	Data direct access register

レジスタ名	アドレス オフセット	タイ プ	幅	初期値	説明
S_DATA				-	[7:0] Transmit data Set TX_DATA prior to starting a DIRECT_ACCESS transaction [7:0] Receive data Read RX_DATA after completing a DIRECT_ACCESS transaction
SD_ADDR_7_0	0x007	RW	8	0x00	SD address [7:0] bit register [7:0] SD_ADDR[7:0]
SD_ADDR_15_8	0x008	RW	8	0x00	SD address [15:8] bit register [7:0] SD_ADDR[15:8]
SD_ADDR_23_16	0x009	RW	8	0x00	SD address [23:16] bit register [7:0] SD_ADDR[23:16]
SD_ADDR_31_24	0x00A	RW	8	0x00	SD address [31:24] bit register [7:0] SD_ADDR[31:24]
SPI_CLK_DEL	0x00B	RW	8	0x00	SPI clock control register [7:0] Control the frequency of the SPI_CLK after SD initialization is completed
RESERVED0	0x00C- 0x00F	-	-	-	Reserved
RX_FIFO_DATA	0x010	RW	8	-	SPI block reading data register [7:0] SD/MMC block read data, fifo size matches the SD/MMC block size of 512 bytes
RESERVED1	0x011	-	-	-	Reserved
RX_FIFO_DATA_COUNT_MSB	0x012	RO	8	-	MSB byte of reading data count register [7:0] MSByte of FIFO_DATA_COUNT, indicates the number of data entries within the fifo
RX_FIFO_DATA_COUNT_LSB	0x013	RO	8	-	LSB byte of reading data count register [7:0] LSByte of FIFO_DATA_COUNT, indicates the number of data entries within the fifo

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
RX_FIFO_CONT ROL	0x014	WO	8	0x00	SD block reading data control register [7:1] Reserved [0] Force fifo empty 1 = Force fifo empty, delete all the data samples within the fifo, self clearing
RESERVED2	0x015- 0x019	-	-	-	Reserved
TX_FIFO_DATA	0x020	WO	8	-	SD block writing data register [7:0] SD/MMC block write data, fifo size matches the SD/MMC block size of 512 bytes
RESERVED3	0x021- 0x023	-	-	-	Reserved
TX_FIFO_CONT ROL	0x024	WO	8	0x00	SD block writing data control register [7:1] Reserved [0] Force fifo empty 1 = Force fifo empty, delete all the data samples within the fifo, self clearing

13.2.2 SD_Host_v2

ソリューション「…¥solution¥SD_Host_v2」は、FAT32 ファイルシステムを採用する 32GB SD-Card をサポートし、APB SD Host Controller を介して SD-Card を制御します。

レジスタの定義は、
solution¥SD_Host_v2¥ref_design¥MCU_RefDesign¥apb_sd_host¥project
¥apb_sd_host¥inc¥apb_sd_host.h にあります。

詳細については、『Gowin APB SD Host IP ユーザーガイド
([IPUG1215](#))』を参照してください。

13.2.3 SD_SPI_IF_v1

ソリューション「…¥solution¥SD_SPI_IF_v1」は、FAT32 ファイルシステムを採用する 32GB SD-Card をサポートし、SPI Master 周辺機器を介して SD-Card を制御します。

SPI Master レジスタの定義は、
library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

詳細については、[9SPI Master](#) を参照してください。

13.3 ドライバーの使用方法

13.3.1 SD-Card 周辺機器

SD-Card ドライバーの使用を表 13-に示します。SD-Card ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_sdcard.c にあります。

表 13-2 SD-Card ドライバーの使用方法

名称	説明
SD_Init	SD hardware initialization
SD_BlockWrite	SD block write data, block size is 512 bytes.
SD_BlockRead	SD block read data, block size is 512 bytes.

13.3.2 SD_Host_v2

ソリューション「…¥solution¥SD_Host_v2」は、FAT32 ファイルシステムを採用する 32GB SD-Card をサポートし、APB SD Host Controller を介して SD-Card を制御します。

ドライバの定義は、
solution¥SD_Host_v2¥ref_design¥MCU_RefDesign¥apb_sd_host¥project¥apb_sd_host¥src¥apb_sd_host_driver.c にあります。

詳細については、『Gowin APB SD Host IP ユーザーガイド ([IPUG1215](#))』を参照してください。

13.3.3 SD_SPI_IF_v1

ソリューション「solution¥SD_SPI_IF_v1」は、FAT32 ファイルシステムを採用する 32GB SD-Card をサポートし、SPI Master 周辺機器を介して SD-Card を制御します。

ドライバの使用方法を表 13-に示します。ドライバの定義は、
solution¥SD_SPI_IF_v1¥ref_design¥MCU_RefDesign¥cm1_fatfs¥project¥fatfs¥src¥spi_sdcard.c にあります。

表 13- SD_SPI_IF_v1 ドライバーの使用方法

名称	説明
SD_Init	Initialize the SD communication.
SD_Detect	Detect whether SD card is plugged in the memory slot correctly.
SD_GetCardInfo	Get information about specific card.
SD_WriteBlock	Write a block of data on SD card.
SD_WriteMultiBlocks	Write multiple blocks of data on SD card.
SD_ReadBlock	Read a block of data from SD card.
SD_ReadMultiBlocks	Read multiple blocks of data from SD card.
SD_GetCSDRegister	Get the CSD card register.
SD_GetCIDRegister	Get the CID card register.

SD_SendCmd	Send 5 bytes command to SD card.
SD_GetDataResponse	Get SD data response.
SD_GetResponse	Get SD response.
SD_GetStatus	Get SD status.
SD_GetCardType	Get SD card revision, SDSC or SDHC.
SD_GoldleState	Put SD to idle state.
SD_WriteByte	Write a byte on SD card.

14_{CAN}

14.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの CAN 周辺機器が含まれています。

- AHB インターフェース
- CAN2.0A と CAN2.0B プロトコル、および ISO 11898-1 標準に準拠
- CAN-FD をサポート：非 ISO CAN FD で、Bosch プロトコルに準拠、ISO/DIS 11898-1 に準拠。
- 独立したシステムクロックと CAN バスクロック
- 特定のアプリケーションで送受信メッセージを保存するための最適なバッファサイズを提供する柔軟な共有バッファスキーム。総バッファ深さは、IP コアの生成時に構成できます。送信バッファ、受信バッファ、および高優先度送信バッファをサポートします。CPU は、送信バッファ、受信バッファ、および高優先度送信バッファの深さを個別に構成できます。
- アクセプタンス・フィルタの数を 1~16 個に構成可能
- プログラム可能なボーレートプリスケイラー (BRP)：CAN バスクロックに基づいて TQ CLK を生成。8 ビットの BRP レジスタは 4~255 分周をサポート。

14.2 レジスタの定義

CAN レジスタの定義を表 14-1 に示します。CAN レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 14-1 CAN 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
SRST	0x0000	RW	32	0x00000000	Software reset register

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[31:1] Reserved [0] Control reset 1 = Hardware reset 0 = Software reset
CMD	0x0004	RW	32	0x00000000	Command register [31:1] Reserved [0] Command settings 1 = Working mode 0 = Command mode
BRP	0x0008	RW	32	0x00000000	Baud rate prescaler register [31:8] Reserved [7:0] Baud rate prescaler
BTN	0x000C	RW	32	0x00000000	Bit timing (nominal) register [28:24] sjw_nom [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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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] Enable TX high-priority message retry [21] Enable TX high-priority message failed [8] Enable Error status [5] Enable 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[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 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
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 ドライバーの使用方法

CAN ドライバーの使用方法を表 14-2 に示します。CAN ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_can.c にあります。

表 14-2 CAN ドライバーの使用方法

名称	説明
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

名称	説明
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

名称	説明
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_ie	Set AFM ID Extension mask
can_read_afm_ie	Get AFM ID Extension mask
can_set_afm_bid	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_txhbspace	Get txhbspace
can_read_txhbsts_txhbwerr	Get txhbwerr
can_read_rxbsts	Get rxbdepth

15 Ethernet

15.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの Ethernet 周辺機器が含まれています。

- AHB インターフェース
- IEEE802.3 プロトコルでイーサネット MAC 層の機能を説明
- RGMII/GMII/MII インターフェースをサポート
- 10/100/1000M のレートをサポート
- 全二重および半二重モードをサポート、半二重モードでの衝突検出をサポート
- CRC を自動的に追加および検証するかどうかはオプション
- padding(充填)機能をサポート
- イーサネットフレームの分類統計をサポート
- イーサネットフレーム・エラーの統計をサポート
- IFG を構成可能
- Jumbo モードをサポート
- 全二重モードでの Flow Control をサポート
- 管理インターフェース(MDC/MDIO)をサポート

15.2 レジスタの定義

Ethernet レジスタの定義を表 15-1 に示します。Ethernet レジスタの定義は、library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 15-1 Ethernet 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
ETH_TX_DATA	0x000-0x5FF	WO	32	0x00000000	Transmit data registers

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

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
					[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
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 ドライバーの使用方法

Ethernet ドライバーの使用を表 15-2 に示します。Ethernet ドライバーの定義は、library¥libraries¥drivers¥src¥GOWIN_M1_ethernet.c にあります。

表 15-2 Ethernet ドライバーの使用方法

名称	説明
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

名称	説明
miim_write	MIIM interface transmits data
miim_receive	MIIM interface receives data

16 DDR3 Memory

16.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの DDR3 Memory 周辺機器が含まれています。

- AHB インターフェース
- 業界標準の DDR3 SDRAM デバイスおよび JESD79-3F 仕様互換のモジュールをサポート
- 16 ビットのデータ幅をサポート
- UDIMM メモリモジュールをサポート
- x8/x16 データ幅のメモリをサポート
- プログラム可能なバースト長(4)
- クロック比 1 : 4 をサポート

16.2 レジスタの定義

DDR3 Memory レジスタの定義を表 16-1 に示します。DDR3 Memory レジスタの定義は、

library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 16-1 DDR3 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
RESERVED	0x0000	-	-	-	Reserved
WR_ADDR	0x0004	RW	32	0x0	Write address register
WR_DATA	0x0008-0x0014	WO	128	0x0	Write data register
RD_ADDR	0x0018	RW	32	0x0	Read address register
RD_EN	0x001c	RW	32	0x0	Read enable register [31:1] Reserved [0] Read enable 1 = Enable 0 = Disable

レジスタ名	アドレスオフセット	タイプ	幅	初期値	説明
RD_DATA	0x0020-0x002c	RO	128	0x0	Read data register
INIT	0x0030	RW	32	0x0	Initialized completely flag register [31:1] Reserved [0] Initialized completely flag
WR_EN	0x0034	RW	32	0x0	Write enable and ending flag register [31:1] Reserved [0] Write enable and ending flag 1 = Enable 0 = Ending

16.3 ドライバーの使用方法

DDR3 Memory ドライバーの使用方法を表 16-2 に示します。DDR3 Memory ドライバーの定義は、
library¥libraries¥drivers¥src¥GOWIN_M1_ddr3.c にあります。

表 16-2 DDR3 Memory ドライバーの使用方法

名称	説明
DDR3_Init	Initialize DDR3 Memory
DDR3_Read	Read data from DDR3 Memory
DDR3_Write	Write data into DDR3 Memory

17 SPI-Flash Memory

17.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの SPI-Flash Memory 周辺機器が含まれています。

- SPI-Flash のダウンロードは AHB バスインターフェースで実装
- SPI-Flash の読み出し、書き込み、および消去は APB バスインターフェースで実装
- Quad SPI-Flash Memory の読み出し、書き込み、および消去をサポート

17.2 レジスタの定義

SPI-Flash Memory レジスタの定義を表 17-1 に示します。SPI-Flash Memory レジスタの定義は、
library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 17-1 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 00 = 1 byte 01 = 2 bytes 10 = 3 bytes 11 = 4 bytes [15:13] Reserved [12:8] Data length

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[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 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

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					[4] Status of SPI-Flash write protect signal [3] Status of SPI MISO signal [2] Status of SPI MOSI signal [1] Status of SPI SCLK signal [0] Status of SPI CS signal
RESERVED1[2]	0x18-0x1C	-	-	-	Reserved
TRANSCTRL	0x20	RW	32	0x0	SPI transfer control register [31] Reserved [30] SPI command phase enable 0 = Disable the command phase 1 = Enable the command phase (Master mode only) [29] SPI address phase enable 0 = Disable the address phase 1 = Enable the address phase (Master mode only) [28] SPI address phase format 0 = Address phase is single mode 1 = The format of the address phase is the same as the DualQuad data phase (Master mode only) [27:24] Transfer mode 0000 = Write and read at the same time 0001 = Write only 0010 = Read only 0011 = Write, Read 0100 = Read, Write 0101 = Write, Dummy, Read 0110 = Read, Dummy, Write 0111 = None data 1000 = Dummy, Write 1001 = Dummy, Read 1010~1111 = Reserved [23:22] SPI data phase format 00 = Single mode 01 = Dual I/O mode 10 = Quad I/O mode 11 = Reserved [21] Append and one-byte special token following the address phase for SPI read transfers [20:12] Transfer count for write data [11] The value of the one-byte special token following the address phase for

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					SPI read transfers 0 = token value is 0x00 1 = token value is 0x69 [10:9] Dummy data count [8:0] Transfer count for read data
CMD	0x24	RW	32	0x0	SPI command register [31:8] Reserved [7:0] SPI command
ADDR	0x28	RW	32	0x0	SPI address register [31:0] SPI address (Master mode only)
DATA	0x2C	RW	32	0x0	SPI data register [31:0] Data to transmit or the received data
CTRL	0x30	RW	32	0x0	SPI controller register [31:21] Reserved [20:16] Transmit FIFO threshold [15:13] Reserved [12:8] Receive FIFO threshold [7:5] Reserved [4] TX DMA enable [3] RX DMA enable [2] Transmit FIFO reset [1] Receive FIFO reset [0] SPI reset
STATUS	0x34	RO	32	0x0	SPI status register [31:24] Reserved [23] Transmit FIFO full flag [22] Transmit FIFO empty flag [21] Reserved [20:16] Number of valid entries 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 [3] Enable the SPI transmit FIFO

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					threshold interrupt [2] Enable the SPI receive FIFO threshold interrupt [1] Enable SPI transmit FIFO underrun interrupt (Slave mode only) [0] Enable SPI receive FIFO overrun interrupt (Slave mode only)
INTRST	0x3C	WO	32	0x0	SPI interrupt status register [31:6] Reserved [5] Slave command interrupt (Slave mode only) [4] End of SPI transfer interrupt [3] TX FIFO threshold interrupt [2] RX FIFO threshold interrupt [1] TX FIFO underrun interrupt (Slave mode only) [0] RX FIFO overrun interrupt (Slave mode only)
TIMING	0x40	RW	32	0x0	SPI interface timing register [31:14] Reserved [13:12] The minimum time between the edges of SPI CS and the edges of SCLK [11:8] The minimum time the SPI CS should stay HIGH [7:0] The clock frequency ratio between the clock source and SPI interface SCLK
RESERVED2[3]	0x44-0x4C	-	-	-	Reserved
MEMCTRL	0x50	RW	32	0x0	SPI memory access control register [31:9] Reserved [8] This bit is set when “MEMCTRL” / “TIMING” is written [7:4] Reserved [3:0] Selects the SPI command
RESERVED3[3]	0x54-0x5C	-	-	-	Reserved
SLVST	0x60	RW	32	0x0	SPI slave status register [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

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

17.3 ドライバーの使用方法

17.3.1 QSPI-Flash ドライバー

QSPI-Flash Memory ドライバーの使用方法を表 17-2 に示します。
QSPI-Flash Memory ドライバーの定義は、
library¥libraries¥drivers¥src¥GOWIN_M1_qspi_flash.c にあります。

表 17-2 QSPI-Flash Memory ドライバーの使用方法

名称	説明
qspi_flash_init	Initialize QSPI-Flash Memory
change_mode_qspi_flash	Switch QSPI-Flash Memory mode between download and read, write, erase memory
qspi_flash_io_fast_read	Read data from QSPI-Flash Memory fastest, a single command and multi data
qspi_flash_fast_read	Read data from QSPI-Flash Memory fast, a single command and a single data

qspi_flash_write	Write data into QSPI-Flash Memory
qspi_flash_4ksector_erase	Erase 4KB sectors of QSPI-Flash Memory
qspi_flash_64ksector_erase	Erase 64KB sectors of QSPI-Flash Memory
qspi_flash_page_program	Write data into QSPI-Flash Memory with pages
qspi_flash_chip_erase	Erase full chip of QSPI-Flash Memory
qspi_flash_write_sr	Write status register of QSPI-Flash Memory
qspi_flash_read_sr	Read status register of QSPI-Flash Memory
qspi_flash_Enable	Enable QSPI-Flash Memory

17.3.2 SPI-Flash ドライバー

SPI-Flash Memory ドライバーの使用方法を表 17-3 に示します。SPI-Flash Memory ドライバーの定義は、`library¥libraries¥drivers¥src¥GOWIN_M1_spi_flash.c` にあります。

表 17-3 SPI-Flash Memory ドライバーの使用方法

名称	説明
<code>spi_flash_init</code>	Initialize SPI-Flash Memory
<code>change_mode_spi_flash</code>	Switch SPI-Flash Memory mode between download and read, write, erase memory
<code>spi_flash_read</code>	Read data from SPI-Flash Memory
<code>spi_flash_write</code>	Write data into SPI-Flash Memory
<code>spi_flash_4ksector_erase</code>	Erase 4KB sectors of SPI-Flash Memory
<code>spi_flash_page_program</code>	Write data into SPI-Flash Memory with pages
<code>spi_flash_64ksector_erase</code>	Erase 64KB sectors of SPI-Flash Memory

18 PSRAM Memory

18.1 特徴

Gowin_EMPU_M1 には、AHB バスを介してアクセスされる 1 つの PSRAM Memory 周辺機器が含まれています。

- AHB インターフェース
- 標準の PSRAM インターフェースと互換性あり
- 8 ビットのデータ幅をサポート
- x8 データ幅のメモリをサポート
- プログラム可能なバースト長(32)
- クロック比 1 : 2 をサポート
- 初期レイテンシ(6)をサポート
- 固定レイテンシ・モードをサポート
- パワーダウン・オプションをサポート
- ドライブ強度は 50
- セルフリフレッシュ領域は full
- リフレッシュレートは normal

18.2 レジスタの定義

PSRAM Memory レジスタの定義を表 18-1 に示します。PSRAM Memory レジスタの定義は、
library¥libraries¥cmsis¥cm1¥device_support¥GOWIN_M1.h にあります。

表 18-1 PSRAM 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
CMD	0x00	RW	1	0x0	Command register [0] Operation type

レジスタ名	アドレス オフセット	タイプ	幅	初期値	説明
					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

18.3 ドライバーの使用方法

PSRAM Memory ドライバーの使用方法を表 18-2 に示します。
PSRAM Memory ドライバーの定義は、
library¥libraries¥drivers¥src¥GOWIN_M1_psram.c にあります。

表 18-2 PSRAM Memory ドライバーの使用方法

名称	説明
PSRAM_Check_Init_Status	Check the status of PSRAM Memory

名称	説明
	initialization
PSRAM_Mode_Set	Set the mode fo 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

19_{RTOS}

Gowin_EMPU_M1 は、uC/OS-III、FreeRTOS、および RT-Thread Nano RTOS をサポートします。

19.1 uC/OS-III

19.1.1 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_M1 は uC/OS-III をサポートします。
- uC/OS-III のソースコードは Micrium の[ホームページ](#)からダウンロード可能。

19.1.2 バージョン

Gowin_EMPU_M1 は、バージョン V3.03.00 の uC/OS-III をサポートします。

19.1.3 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、使用されている開発ボードをサポートするようにすることが可能。

19.2 FreeRTOS

19.2.1 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_M1 は FreeRTOS をサポートします。
- FreeRTOS のソースコードは FreeRTOS のホームページ <http://www.FreeRTOS.org> からダウンロードしてください。

19.2.2 バージョン

Gowin_EMPU_M1 は、バージョン V10.2.1 の FreeRTOS をサポートします。

19.2.3 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。

19.3 RT-Thread Nano

19.3.1 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 Nano 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_M1 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。

19.3.2 バージョン

Gowin_EMPU_M1 は、バージョン V3.1.5 の RT-Thread Nano をサポートします。

19.3.3 構成

- `bsp¥cm1¥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¥cm1¥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。

20

プロトコルスタック・ソフトウェアプログラミング

Gowin_EMPU_M1 は、次のプロトコルスタック・ソフトウェアプログラミングをサポートしています：TCP/IP プロトコルスタック。

20.1 TCP/IP プロトコルスタック

Gowin_EMPU_M1 は、オープンソースの TCP/IP プロトコルスタック LwIP をサポートします。

LwIP プロトコルスタックは、組み込みシステムおよび小型デバイス向けに設計された軽量のオープンソース TCP/IP プロトコルスタックです。LwIP プロトコルスタックが提供する TCP/IP ネットワークプロトコルのサポートにより、組み込みシステムはネットワークを介して他のデバイスと通信できるようになります。

20.1.1 特徴

- 軽量：メモリとプロセッサリソースの消費量が少なく、組み込みシステムで効率的に実行できます。
- カスタマイズ可能：特定のニーズに合わせて必要なプロトコルと機能のみを含めることができます。これにより、メモリとプロセッサリソースの消費量が削減されます。
- 高性能：ゼロコピーやイベント駆動メカニズムなどの性能最適化手法を使用して、ネットワーク通信の効率とスループットを向上させます。
- 移植性：移植性に優れており、RTOS FreeRTOS や uC/OS-III などのさまざまなオペレーティングシステムおよびハードウェアプラットフォーム上で実行できます。
- 汎用性：複数のプロトコルをサポートします。TCP/IP プロトコルスタックに加えて、HTTP、SNMP、MQTT などのいくつかの一般的なアプリケーション層プロトコル実装も提供します。

- Gowin_EMPU_M1 は、MCU、RTOS FreeRTOS、および uC/OS-III モードの LwIP プロトコルスタックをサポートします。

20.1.2 バージョン

Gowin_EMPU_M1 は、バージョン V2.1.2 の LwIP プロトコルスタックをサポートします。

21 アプリケーションプログラム

Gowin_EMPU_M1 は、ARM Keil MDK(テスト済みバージョン V5.26) および GMD (GOWIN MCU Designer (テスト済みバージョン 2025.01) ソフトウェア環境用のアプリケーションプログラムを提供します。

次のセクションでは、Gowin_EMPU_M1 の個々のアプリケーションプログラムについて説明します。

21.1 UART

UART アプリケーションプログラム :

...

```
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥printf
```

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥printf
```

...

```
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥uart_rx
```

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥uart_rx
```

...

```
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥uart_rx_intr
```

...

```
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥uart_rx_intr
```

21.2 Timer

Timer アプリケーションプログラム :

...

```

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥time
r
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
timer

```

21.3 Watch Dog

Watch Dog アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥watc
hdog
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
watchdog

```

21.4 GPIO

GPIO アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥led
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
led
...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥gpio
_in_intr
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
gpio_in_intr
...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥keys
can
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
keyscan

```

21.5 I²C Master

I²C Master アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥i2c_
master
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥

```

i2c_master

21.6 SPI Master

SPI Master アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥spi_master

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥spi_master

21.7 RTC

RTC アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥rtc

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥rtc

21.8 TRNG

TRNG アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥trng

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥trng

21.9 DualTimer

DualTimer アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥dualtimer

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥dualtimer

21.10 SD-Card

SD-Card アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_fatfs

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_fatfs

...¥solution¥SD_Host_v2¥ref_design¥MCU_RefDesign¥apb_sd_host

…¥solution¥SD_SPI_IF_v1¥ref_design¥MCU_RefDesign¥cm1_fatfs

すべての SD-Card は、初めて使用する前にフォーマットする必要があることに注意してください。

21.11 CAN

CAN アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥can

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥can

21.12 Ethernet

Ethernet アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥ethernet

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥ethernet

21.13 DDR3 Memory

DDR3 Memory アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥ddr3

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥ddr3

21.14 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¥cm1_demo¥project¥spi_flash

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥spi_flash

21.15 QSPI-Flash Memory

QSPI-Flash Memory アプリケーションプログラム :

...

```

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥qspi
_flash
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
qspi_flash

```

21.16 PSRAM Memory

PSRAM Memory アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥psra
m
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
psram

```

21.17 Interrupt

Interrupt アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥inter
rupt
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
interrupt

```

21.18 DMM

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

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥dm
m
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
dmm

```

21.19 AHB Master

AHB Master アプリケーションプログラム :

```

...
¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥ahb
_master
...
¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥
ahb_master

```

21.20 APB Master

APB Master アプリケーションプログラム :

...

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_demo¥project¥apb_master

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_demo¥src¥project¥apb_master

21.21 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¥cm1_ucos_iii

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_ucos_iii

21.22 FreeRTOS

RTOS FreeRTOS アプリケーションプログラム :

...¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_freertos

...¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_freertos

21.23 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¥cm1_rtthread_nano

...

¥ref_design¥MCU_RefDesign¥GMD_RefDesign¥cm1_rtthread_nano

21.24 Embedded Memory

Embedded Memory ソリューションのアプリケーションプログラム :

...

¥solution¥Embedded_Memory¥ref_design¥MCU_RefDesign¥cm1_demo

21.25 External Memory

External Memory ソリューションのアプリケーションプログラム :

...

¥solution¥External_Memory¥ref_design¥MCU_RefDesign¥cm1_demo

21.26 Make Hex

Make Hex ツールのアプリケーションプログラム :

...¥tool¥make_hex¥ref_design¥MCU_RefDesign¥cm1_demo

21.27 Merge Bit

Merge Bit ツールのアプリケーションプログラム：

…¥tool¥merge_bit¥ref_design¥MCU_RefDesign¥cm1_demo

21.28 LwIP プロトコルスタック

LwIP プロトコルスタック・アプリケーションプログラム：

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
freertos_ping

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
freertos_tcp_netcon_client

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
freertos_tcp_socket_client

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
freertos_tcpserver_socket

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
raw_ping

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
raw_udp

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
uc3_client_ok

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
uc3_ping_ok

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥lwip_
uc3_server_ok

21.29 uIP プロトコルスタック

uIP プロトコルスタック・アプリケーションプログラム：

…

¥ref_design¥MCU_RefDesign¥MDK_RefDesign¥cm1_tcpip¥project¥ui_h
elloworld

