RE01 1500KB、256KB グループ

SMIP ドライバ 詳細仕様書

要旨

本書では、RE01 1500KB、256KB グループ CMSIS software package の SMIP ドライバ（以下、SMIP ドライバ）の詳細仕様を説明します。

動作確認デバイス

RE01 1500KB グループ

RE01 256KB グループ

目次

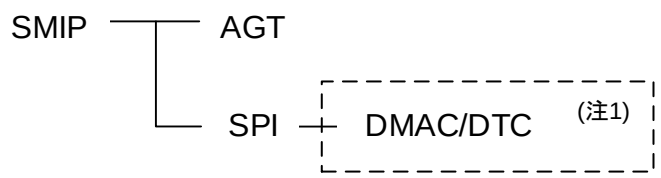
1. 概要

SMIP ドライバは、RE01 1500KB および 256KB グループにて SPI を使用し MIP-LCD を制御するためのドライバです。本ドライバは、以下の LCD にて動作確認をしています。

型番	メーカ	解像度	表示色
LPM013M126A	JDI	176(H)×176(V)	カラー（8 色）
TN0181ANVNANN-*N*03	京セラ	256(H)×256(V)	モノクロ（白黒 2 値）
LS013B7DH03	SHARP	128(H)×128(V)	モノクロ（白黒 2 値）

以下、各メーカに対応する型番のディスプレイは、メーカ名で記載します。コンフィグレーションで定義されている各 LCD 設定のデフォルト値は、上記型番の LCD に対応しています。

本ドライバでは、図 1-1 に示す周辺機能を使用します。



注1. SPIドライバにてDMAドライバを使用する設定にした場合のみ

図 1-1 SMIP ドライバ周辺機能ブロック図

本ドライバで使用する周辺機能の一覧を以下に示します。

周辺機能	内容	動作モード
AGT	VCOM(EXTCOMIN)出力	タイマモード/パルス出力モード
SPI(注 1)	データ出力	マスタ送信モード、MSB/LSB ファースト(注 2)、8bit 長
DMAC/DTC	SPI のデータ書き込み (SPI ドライバにて DMA ドライバを使用する設定をした場合のみ)	ノーマル転送

注1. SPI ドライバの設定が必要です。SPI ドライバの詳細については「RE01 1500KB、256KB グループ CMSIS ドライバ SPI 仕様書(R01AN4728)」を参照してください。

注2. MSB/LSB は r_smip_cfg.h の"SMIP_CFG_SPI_BIT_ORDER"で設定します。

1.1 SMIP で使用する SPI ドライバの設定

1.1.1 DMA 使用設定

SPI ドライバにて DMA ドライバを使用するかを r_spi_cfg.h の SPIn_TRANSMIT_CONTROL(n = 0,1)定義で設定します。DTC ドライバを使用する場合の例を図 1-2 に示します。

```
#define SPI0_TRANSMIT_CONTROL   SPI_USED_DTC     ///< SPI0 transmit control
```

図 1-2 r_spi_cfg.h での DTC ドライバ使用設定例(SPI0 使用時)

1.1.2 SPI 通信制御割り込み(NVIC)設定

SMIP ドライバで、SPI をマスタ送信モードで動作させるには、r_system_cfg.h にて使用する割り込みを登録する必要があります。割り込み(NVIC)の詳細は「RE01 1500KB、256KB グループ CMSIS パッケージを用いた開発スタートアップガイド（r01an4660）」の「割り込み（NVIC）の設定」を参照してください。SPI ドライバで使用する割り込み定義を表 1-1 に示します。

表 1-1 NVIC の登録定義(n = 0,1、m = 0～3)

モード	使用用途	NVIC 登録定義
マスタ	送信のみで使用	[送信制御に割り込み、DTC を使用時] SYSTEM_CFG_EVENT_NUMBER_SPIn_SPTI [送信制御に DMAC を使用時] SYSTEM_CFG_EVENT_NUMBER_DMACm_INT
		SYSTEM_CFG_EVENT_NUMBER_SPIn_SPII

```
#define SYSTEM_CFG_EVENT_NUMBER_SCI0_TXI
          (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)    /*!< Numbers 1/5/9/13/17/21/25/29 only */
#define SYSTEM_CFG_EVENT_NUMBER_SPI0_SPTI
          (SYSTEM_IRQ_EVENT_NUMBER1)              /*!< Numbers 1/5/9/13/17/21/25/29 only */
#define SYSTEM_CFG_EVENT_NUMBER_SOL_DL
          (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)    /*!< Numbers 1/9/17/25 only */
・・・
#define SYSTEM_CFG_EVENT_NUMBER_SCI0_TEI
          (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)    /*!< Numbers 2/6/10/14/18/22/26/30 only */
#define SYSTEM_CFG_EVENT_NUMBER_SPI0_SPII
          (SYSTEM_IRQ_EVENT_NUMBER2)              /*!< Numbers 2/6/10/14/18/22/26/30 only */
#define SYSTEM_CFG_EVENT_NUMBER_SPI0_SPTEND
          (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)    /*!< Numbers 2/6/10/14/18/22/26/30 only */
・・・
```

図 1-3 r_sysytem_cfg.h での NVIC への割り込み登録例(SPI0 使用時)

1.1.3 SPI 端子設定

SPI ドライバで使用する端子は、pin.c の R_RSPI_Pinset_CHn(n=0,1)関数で設定、R_RSPI_Pinclr_CHn 関数で解放されます。

使用する端子は、pin.c の R_RSPI_Pinset_CHn、R_RSPI_Pinclr_CHn (n = 0,1)関数を編集して選択してください。SPI 機能で使用される各端子名には_A、_B、_C および_D という接尾語が付加されています。SPI 機能を割り当てる場合、同じ接尾語の機能端子を選択してください。(注 1)(注 2)

SPI0 の端子設定例を図 1-4 に示します。

注1. 接尾語が同じ信号は、タイミング調整されているグループを表しています。違うグループの信号を同時に使用することはできません。例外として、SPI の「RSPCKA_C」と「MOSIA_C」は「_B」のグループと同時に使用します。また、「SSLB0_D」は「_B」のグループと同時に使用します。

注2. 本ドライバで使用する RST および SCS 信号はポート出力によって制御されます。r_smip_cfg.h にて指定した RST および SCS 制御端子は、SPI を含めた各周辺機能に割り当てないでください。同様に、VCOM 信号に AGTOn を使用しない場合（SMIP_CFG_AGTO_EN の設定値が 1 の場合）、VCOM 制御端子を周辺機能に割り当てないでください。

```
/*****************************************************************************//**
* @brief This function sets Pin of RSPI0.
* @note  In the case of SPI function, the same signal names are present with @n
* the suffixes "_A", "_B" "_C" and "_D" attached. These indicate groups @n
* in terms of timing adjustment, and signals from different @n
* groups cannot be used at the same time. The exceptions are the RSPCKA_C and MOSIA_C signals @n
* for the SPI and the SSLB0_D signal, which can be used at the same time as signals from group B. @n
*****************************************************************************/
/* Function Name : R_RSPI_Pinset_CH0 */
void R_RSPI_Pinset_CH0(void)  // @suppress("API function naming") @suppress("Function length")
{
    /* Disable protection for PFS function (Set to PWPR register) */
    R_SYS_RegisterProtectDisable(SYSTEM_REG_PROTECT_MPC);

//    /* MISOA_A : P105 */
//    PFS->P105PFS_b.PMR  = 0U;
//    PFS->P105PFS_b.ASEL = 0U;
//    PFS->P105PFS_b.ISEL = 0U;
//    PFS->P105PFS_b.PSEL = R_PIN_PRV_RSPI_PSEL;
//    PFS->P105PFS_b.PMR  = 1U;

    /* P500 を MISOA 用端子に設定 */
    /* MISOA_B : P500 */
    PFS->P500PFS_b.ASEL = 0U;
    PFS->P500PFS_b.ISEL = 0U;
    PFS->P500PFS_b.PSEL = R_PIN_PRV_RSPI_PSEL;
    PFS->P500PFS_b.PMR  = 1U;

/* SCS(RST)信号に P103 を使用する場合、P103 を SSLA0A 用端子に設定しないでください */
//    /* SSLA0_A : P103 */
//    PFS->P103PFS_b.ASEL = 0U;  /* 0: Do not use as an analog pin, 1: Use as an analog pin. */
//    PFS->P103PFS_b.ISEL = 0U;  /* 0: Do not use as an IRQn input pin,  1: Use as an IRQn input pin. */
//    PFS->P103PFS_b.PSEL = R_PIN_PRV_RSPI_PSEL;
//    PFS->P103PFS_b.PMR  = 1U;  /* 0: Use the pin as a general I/O port,
                                                                  1: Use the pin as a peripheral module. */
 . . .
}/* End of function R_RSPI_Pinset_CH0() */
/*****************************************************************************//**
* @brief This function clears the pin setting of RSPI0.
*****************************************************************************/
/* Function Name : R_RSPI_Pinclr_CH0 */
void R_RSPI_Pinclr_CH0(void)  // @suppress("API function naming")
{
    /* Disable protection for PFS function (Set to PWPR register) */
    R_SYS_RegisterProtectDisable(SYSTEM_REG_PROTECT_MPC);
//    /* MISOA_A : P105 */
//    PFS->P105PFS &= R_PIN_PRV_CLR_MASK;

    /* MISOA 端子を解放 */
    /* MISOA_B : P500 */
    PFS->P500PFS &= R_PIN_PRV_CLR_MASK;

//    /* SSLA0_A : P103 */
//    PFS->P103PFS &= R_PIN_PRV_CLR_MASK;
 . . .
}
```

図 1-4 端子設定例

1.2 SMIP で使用する AGT の設定

1.2.1 AGT タイマ割り込み(NVIC)設定

AGT タイマ割り込みで VCOM(EXTCOMIN)出力を行う場合、AGT タイマ割り込み（AGTn_AGTI）設定が必要です。割り込み(NVIC)の詳細は「RE01 1500KB、256KB グループ CMSIS パッケージを用いた開発スタートアップガイド（r01an4660）」の「割り込み（NVIC）の設定」を参照してください。

AGT タイマ割り込み設定が必要な組み合わせを表 1-2 に示します。

表 1-2 AGTI 割り込み設定組み合わせ

SMIP_CFG_EXTMODE(注 1)	送信処理	AGTI 割り込み設定
0	京セラ(注 2)	必要
0	JDI および SHARP(注 3)	不要
1	京セラ(注 2)	必要
1	JDI および SHARP(注 3)	必要

注1. EXTMODE 端子設定定義です。EXTMODE 端子入力レベルを設定します。
詳細は「4.6.7 VCOM(EXTCOMIN)出力端子設定定義」を参照してください。

注2. st_smip_cfg_t 構造体 type メンバの b4 が 1 のとき、送信制御は VCOM 出力に同期します

注3. st_smip_cfg_t 構造体 type メンバの b4 が 0 のとき、送信制御は VCOM 出力に同期しません

AGT タイマ割り込み設定が必要な場合、r_system_cfg.h にて NVIC 登録にしてください。AGT タイマ割り込みの登録例を図 1-5 に示します。

```
・・・
#define SYSTEM_CFG_EVENT_NUMBER_QSPI_INTR
    (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)  /*!< Numbers 3/7/11/15/19/23/27/31 only */
#define SYSTEM_CFG_EVENT_NUMBER_AGT0_AGTI
    (SYSTEM_IRQ_EVENT_NUMBER11)         /*!< Numbers 3/11/19/27 only */
#define SYSTEM_CFG_EVENT_NUMBER_USBFS_USBR
    (SYSTEM_IRQ_EVENT_NUMBER_NOT_USED)  /*!< Numbers 3/11/19/27 only */
・・・
```

図 1-5 r_sysytem_cfg.h での NVIC への割り込み登録例(AGT0 使用時)

1.2.2 AGTO 端子設定

VCOM(EXTCOMIN)制御の AGTO 出力を有効にした場合(注)、AGT 端子設定関数および端子解放関数が実行されます。AGT で使用する端子は、R_AGT_Pinset_CHn(n=0,1)関数で設定、R_AGT_Pinclr_CHn 関数で解放されます。

使用する端子は、pin.c の R_AGT_Pinset_CHn、R_AGT_Pinclr_CHn 関数内を修正して設定してください。AGTO0 の端子設定例を図 1-6 に示します。

注3. SMIP_CFG_AGTO_EN の設定値が 1 の場合に、VCOM(EXTCOMIN)制御の AGTO 出力が有効となります。詳細は「4.6.5 EXTMODE 端子設定定義」を参照してください。

```
/*****************************************************************************//**
* @brief This function sets Pin of AGT0.
*******************************************************************************/
/* Function Name : R_AGT_Pinset_CH0 */
void R_AGT_Pinset_CH0(void)  // @suppress("API function naming")
{
・・・

    /* AGTO0 : P808 */
//    PFS->P808PFS_b.ASEL = 0U;  /* 0: Do not use as an analog pin, 1: Use as an analog pin. */
//    PFS->P808PFS_b.ISEL = 0U;  /* 0: Do not use as an IRQn input pin,  1: Use as an IRQn input pin. */
//    PFS->P808PFS_b.PSEL = R_PIN_PRV_AGT_PSEL;
//    PFS->P808PFS_b.PMR  = 1U;
                        /* 0: Use the pin as a general I/O port, 1: Use the pin as a peripheral module. */

/* P111 を AGTO0 用端子に設定 */
    /* AGTO0 : P111 */
    PFS->P111PFS_b.ASEL = 0U;  /* 0: Do not use as an analog pin, 1: Use as an analog pin. */
    PFS->P111PFS_b.ISEL = 0U;  /* 0: Do not use as an IRQn input pin,  1: Use as an IRQn input pin. */
    PFS->P111PFS_b.PSEL = R_PIN_PRV_AGT_PSEL;
    PFS->P111PFS_b.PMR  = 1U;
                       /* 0: Use the pin as a general I/O port, 1: Use the pin as a peripheral module. */

    /* AGTOA0 : P807 */
//    PFS->P807PFS_b.ASEL = 0U;  /* 0: Do not use as an analog pin, 1: Use as an analog pin. */
//    PFS->P807PFS_b.ISEL = 0U;  /* 0: Do not use as an IRQn input pin,  1: Use as an IRQn input pin. */
//    PFS->P807PFS_b.PSEL = R_PIN_PRV_AGT_PSEL;
//    PFS->P807PFS_b.PMR  = 1U;
                    /* 0: Use the pin as a general I/O port, 1: Use the pin as a peripheral module. */
・・・
}/* End of function R_AGT_Pinset_CH0() */

/*****************************************************************************//**
* @brief This function clears the pin setting of AGT0.
*******************************************************************************/
/* Function Name : R_AGT_Pinclr_CH0 */
void R_AGT_Pinclr_CH0(void)  // @suppress("API function naming")
{
  ・・・
    /* AGTO0 : P808 */
//    PFS->P808PFS &= R_PIN_PRV_CLR_MASK;

/* AGTO0 端子を解放 */
    /* AGTO0 : P111 */
    PFS->P111PFS &= R_PIN_PRV_CLR_MASK;

    /* AGTOA0 : P807 */
//    PFS->P807PFS &= R_PIN_PRV_CLR_MASK;
・・・
}/* End of function R_AGT_Pinclr_CH0() */
```

図 1-6 端子設定例

2. ドライバ構成

2.1 ファイル構成

SMIP ドライバは CMSIS Driver Package の HAL_Driver に該当し、ベンダ独自ファイル格納ディレクトリ内の” r_smip_api.c”、” r_smip_api.h”、” r_smip_cfg.h”の 3 個のファイルで構成されます。各ファイルの役割を表 2-1 に、ファイル構成を図 2-1 に示します。

表 2-1 SMIP ドライバ 各ファイルの役割

ファイル名	内容
r_smip_api.c	ドライバソースファイルです ドライバ関数の実態を用意します SMIP ドライバを使用する場合は、本ファイルをビルドする必要があります
r_smip_api.h	ドライバヘッダファイルです ドライバ内で使用するマクロ／型／プロトタイプ宣言が定義されています SMIP ドライバを使用する場合は、本ファイルをインクルードする必要があります
r_smip_cfg.h	コンフィグレーション定義ファイルです ユーザが設定可能なコンフィグレーション定義を用意します

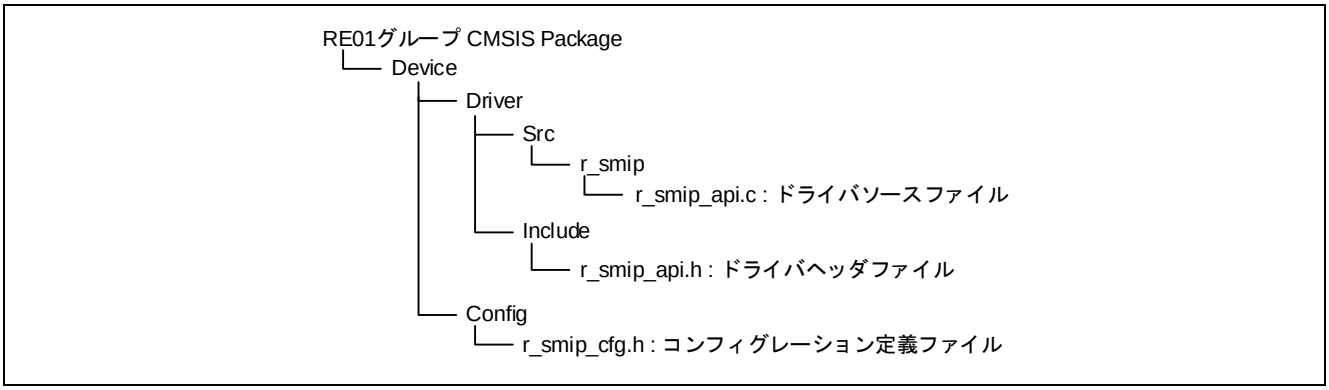


図 2-1 SMIP ドライバファイル構成

2.2 ドライバ API

SMIP ドライバの API を表 2-2 に示します。

表 2-2 SMIP ドライバ API

API	内容
R_SMIP_GetVersion	SMIP ドライバのバージョンを取得します
R_SMIP_Open	SMIP ドライバのオープン（RAM の初期化、SPI ドライバの初期化）を行います AGTI 割り込みを使用する場合(注 1)、割り込みの設定も行います
R_SMIP_Close	SMIP ドライバを解放（SPI ドライバの解放）します AGTI 割り込みを使用する場合(注 1)、割り込みを禁止にします
R_SMIP_PowerOn	MIP-LCD パワーオンシーケンスを実行します
R_SMIP_PowerOff	MIP-LCD パワーオフシーケンスを実行します
R_SMIP_AllZero	0 データを画面全体に送信します
R_SMIP_AllOne	1 データを画面全体に送信します
R_SMIP_Send	モノクロデータの送信を実行します
R_SMIP_SendColor	カラーデータの送信を実行します
R_SMIP_SendCommand(注 2)	コマンドの送信を実行します
R_SMIP_Suspend	SPI 通信設定をクリアし、SPI で使用する端子を解放します
R_SMIP_Resume	SPI 通信設定を再度行い、端子を SPI に割り当てます
R_SMIP_ReconfigSpiSpeed	SPI 通信速度を変更します

注1. SMIP_CFG_EXTMODE が 1、もしくは送信制御が VCOM 出力に同期する場合を指します
詳細は「1.2.1 AGT タイマ割り込み(NVIC)設定」を参照してください

注2. Open 関数にて京セラを指定した場合、使用できません

2.3 SMIP の設定

本ドライバでは、Open 関数にて使用する LCD の形式を指定します。JDI、京セラ、SHARP を使用する場合、第 3 引数に g_smip_tbl_lcd_info[XXX]のポインタを指定してください。XXX には、LCD 形式定義を指定します。SMIP LCD 形式定義の詳細は「2.4.1 SMIP LCD 形式定義」を参照してください。

Open 関数の第 1 引数には使用する SPI チャネルを、第 2 引数には SPI 送信クロック周波数(Hz)を指定してください。コールバック関数を使用する場合、第 4 引数にコールバック関数を指定してください。

例）JDI 製 LCD を使用する場合（SPI0、1Mhz、コールバックなしの場合）

```
R_SMIP_Open(0, 1000000, &g_smip_tbl_lcd_info[SMIP_TYPE_JDI] ,NULL);
```

2.4 マクロ／型定義

SMIP ドライバで、ユーザが参照可能なマクロ／型定義を r_smip_api.h ファイルで定義しています。

2.4.1 SMIP LCD 形式定義

SMIP LCD タイプ定義は、Open 関数の第 3 引数に指定する構造体テーブルのインデックスに使用します。

表 2-3 SMIPLCD 形式定義一覧

定義	値	内容
SMIP_TYPE_KYOCERA	(0x00)	京セラ製 LCD を使用
SMIP_TYPE_JDI	(0x01)	JDI 製 LCD を使用
SMIP_TYPE_SHARP	(0x02)	SHARP 製 LCD を使用

2.4.2 SMIP エラーコード定義

SMIP のエラーコード定義です。

表 2-4 SMIP エラーコード定義一覧

定義	値	内容
SMIP_OK	0	SMIP ドライバの処理が正常に完了しました
SMIP_ERROR	1	SMIP ドライバ処理が異常終了しました
SMIP_ERROR_BUSY	2	SMIP ドライバがビジー状態です
SMIP_ERROR_PARAMETER	3	パラメータが不正です
SMIP_ERROR_MODE	4	モード設定が不正です
SMIP_ERROR_LOCKED	5	AGT のリソースがロックされています
SMIP_ERROR_TIMEOUT	6	タイムアウトが発生しました
SMIP_ERROR_SYSTEM_SETTING	7	システム設定が不正です
SMIP_ERROR_POWER_SEQUENCE	8	パワーオンもしくはパワーオフシーケンスで異常が発生しました

2.4.3 SMIP イベント定義

SMIP のイベント定義です。

表 2-5 SMIP イベント定義一覧

定義	値	内容
SMIP_EVENT_SEND_COMPLETE	(1UL << 0)	データ出力が正常に完了しました
SMIP_EVENT_SEND_ERROR	(1UL << 1)	データ出力で異常が発生しました

2.5 構造体定義

SMIP ドライバでは、ユーザが参照可能な構造体定義を r_smip_api.h ファイルで定義しています。

2.5.1 st_smip_cfg_t 構造体

Open 関数で使用する LCD 設定を指定するときに使用する構造体です。京セラ、JDI、SHARP の設定は、r_smip_api.c に定義されています。設定を変更する場合、r_smip_cfg.h にて、st_smip_cfg_t 構造体メンバに対応するコンフィグレーションを変更してください。コンフィグレーションの設定については「4.6 コンフィグレーション」を参照してください。

st_smip_cfg_t 構造体で定義するパワーオン/パワーオフシーケンスのタイミング設定および送信処理時間設定を図 2-2～図 2-4 に、st_smip_cfg_t 構造体メンバ一覧を表 2-6 に示します。

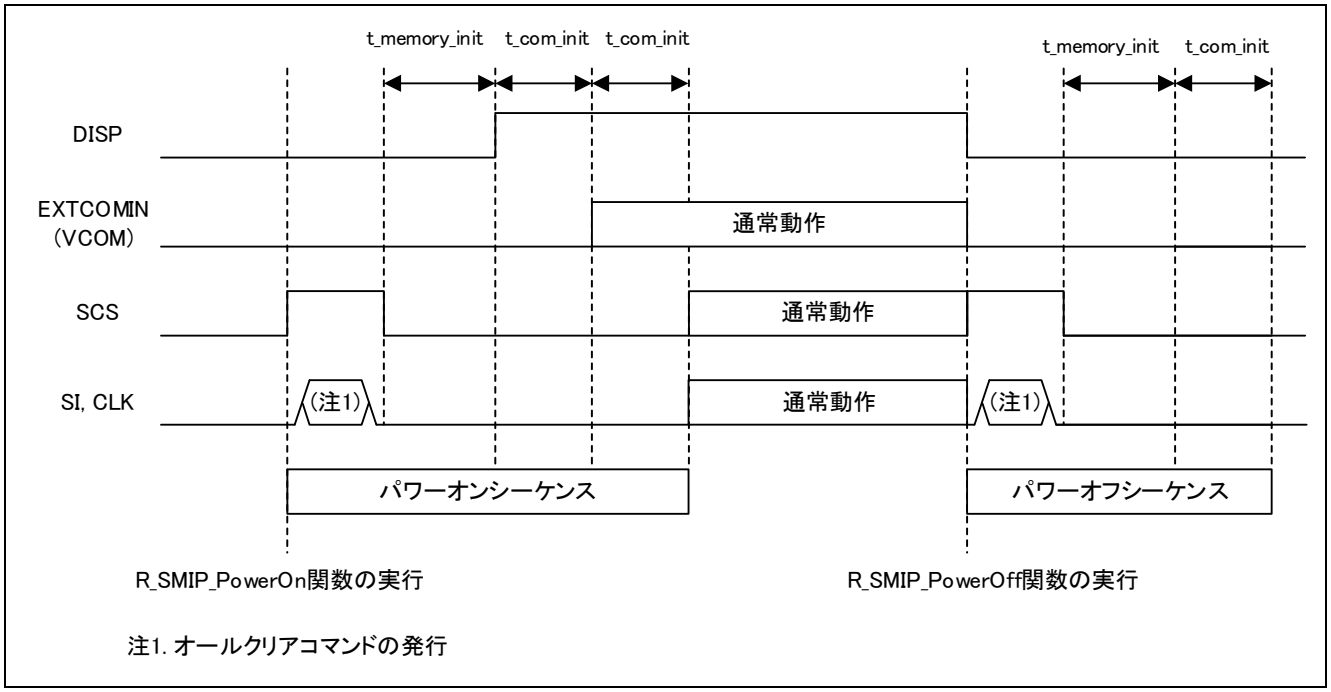


図 2-2 JDI および SHARP のパワーオン/パワーオフシーケンスのタイミング設定

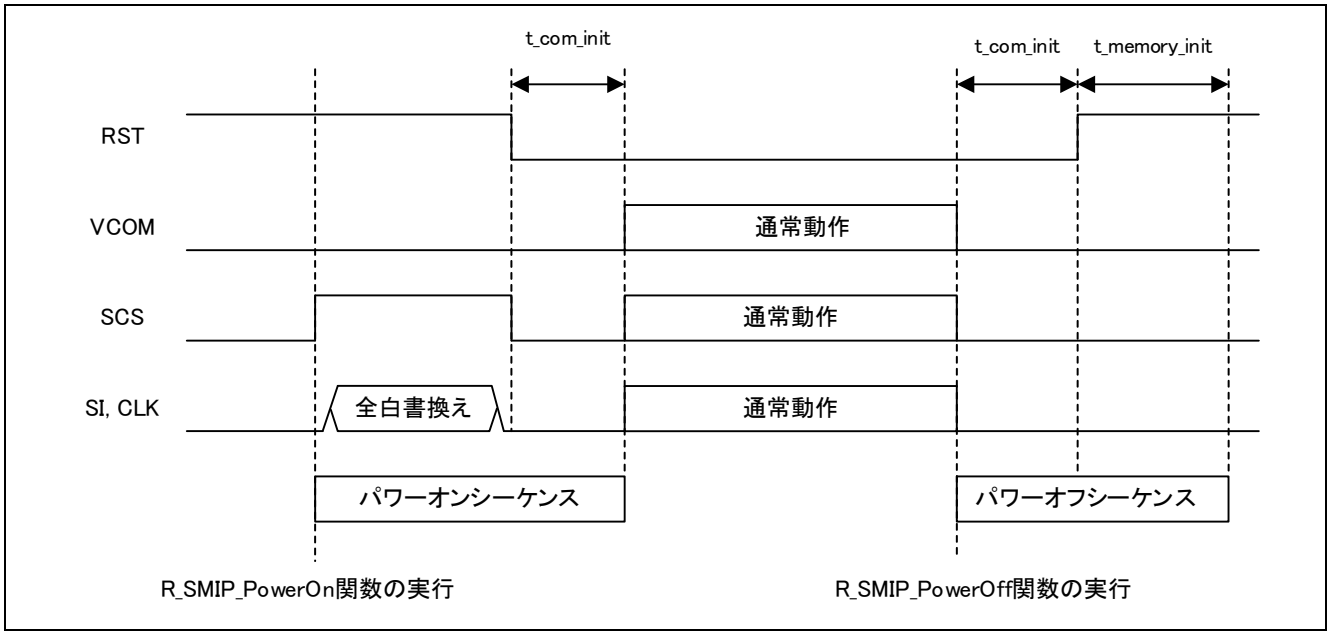


図 2-3 京セラのパワーオン/パワーオフシーケンスのタイミング設定

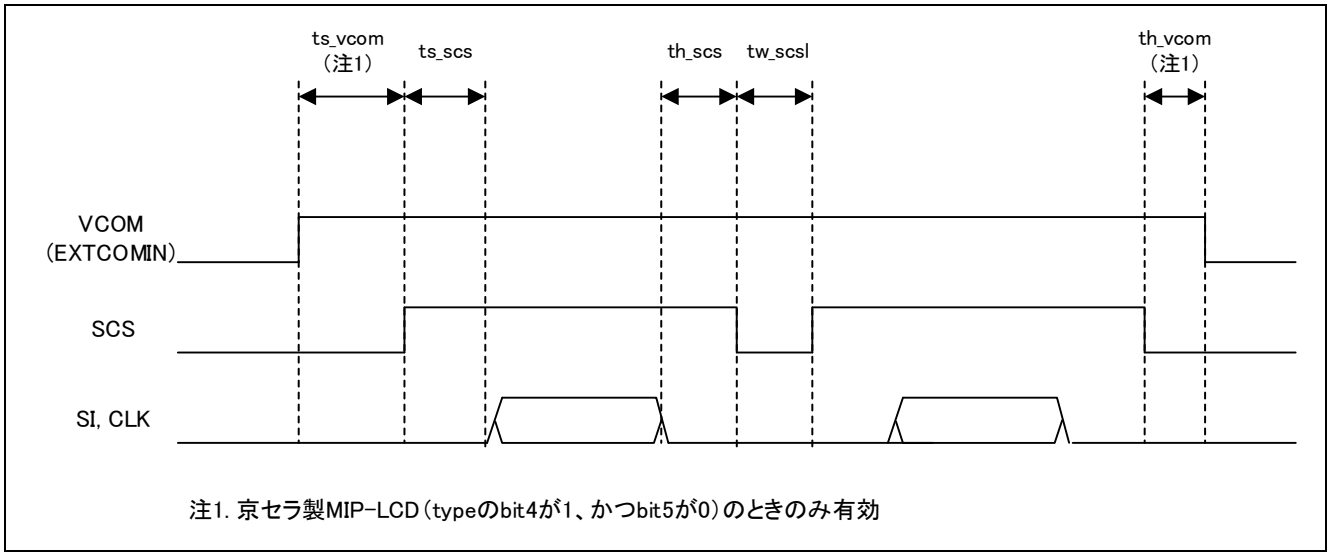


図 2-4 送信処理時間設定

表 2-6 st_smip_cfg_t 構造体

要素名	型	内容
type	uint8_t	b0: カラーモード 0: 3 ビットデータモード 1: 4 ビットデータモード b1: 電源 ON-OFF シーケンス 0:JDI、SHARP 形式 1:京セラ形式 b2: AG ビットエンディアン 0: LSB 1: MSB b3:マルチライン間のダミー出力 0: ダミー出力しない 1: ダミー出力する b4: Send タイミング 0: Send 関数実行時 1: VCOM 出力に同期して送信
mode_size	uint8_t	モードビット数
mode_dmy_size	uint8_t	モードダミービット数
address_size	uint8_t	アドレスビット数
end_dummy_size	uint8_t	ダミービット数
first_addr	uint8_t	初期ラインアドレス
disp_width	uint16_t	MIP-LCD の水平方向ビット数
disp_line	uint16_t	MIP-LCD のライン数
vcom_clk	uint16_t	VCOM 周波数
t_memory_init	uint16_t	JDI,SHARP：メモリ初期化待ち時間 京セラ：終了時の RESET 入力時間
t_com_init	uint16_t	JDI,SHARP：VCOM 待ち時間 京セラ：VCOM 待ち時間入力時間
ts_scs	uint8_t	SCS セットアップ時間
th_scs	uint8_t	SCS ホールド時間
tw_scsl	uint8_t	SCS"L"幅(注 1)
ts_vcom	uint8_t	VCOM セットアップ時間(注 2)
th_vcom	uint8_t	VCOM ホールド時間(注 2)
*cmd_tbl	uint8_t	コマンドテーブル

注1. SMIP_CFG_SCS_WAIT_EN=0 の場合、本設定は無効です
注2. type の b4 が 1 の場合のみ有効です

3. 状態遷移図

SMIP ドライバの状態遷移図を、図 3-1 に示します。

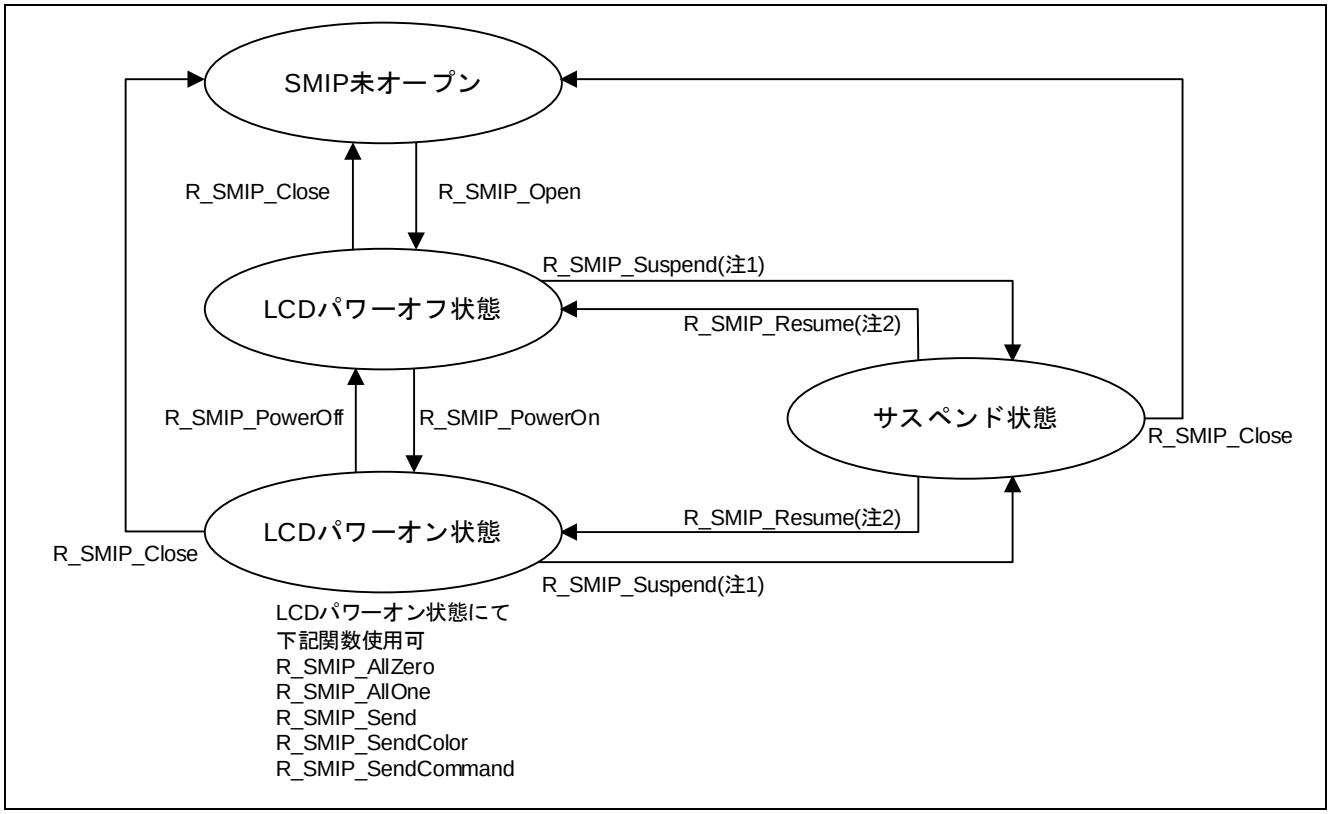


図 3-1 状態遷移図(注 3)(注 4)

注1. SPI 通信による LCD 表示処理中に実行すると、R_SMIP_ERROR_BUSY が返ります

注2. サスペンド状態で R_SMIP_Resume 関数を実行した場合の状態遷移は以下のようになります
LCD パワーオフ状態からサスペンド状態に遷移した場合：LCD パワーオフ状態に復帰
LCD パワーオン状態からサスペンド状態に遷移した場合：LCD パワーオン状態に復帰

注3. R_SMIP_GetVersion 関数はいずれの状態からでも呼び出せます

注4. R_SMIP_ReconfigSpiSpeed 関数は、Open 関数実行後に呼び出すことができます（LCD パワーオン/パワーオフ状態共に実行可能）
ただし、SPI 通信による LCD 表示処理中に実行すると、R_SMIP_ERROR_BUSY が返ります

4. ドライバ動作説明

SMIP ドライバは、SPI ドライバを使用し、任意の画像データを MIP-LCD に出力します。本章では、「1 概要」に記載している LCD をもとに説明します。

LCD の設定は、st_smip_cfg_t 構造体テーブルとして r_smip_api.c に定義されています。st_smip_cfg_t 構造体については「2.5.1 st_smip_cfg_t 構造体」を参照してください。各 LCD には、st_smip_cfg_t 構造体のメンバに対応したコンフィグレーションが定義されています。コンフィグレーション定義の詳細は「4.6 コンフィグレーション」を確認してください。

4.1 JDI 製 MIP-LCD 使用例

4.1.1 JDI 製 MIP-LCD の画像表示例

JDI 製 LCD の設定は、r_smip_api.c の g_smip_tbl_lcd_info[SMIP_TYPE_JDI]に定義されています。JDI の設定定義については「4.6.14 JDI 製 LCD 設定定義」を参照してください。

JDI を使用する場合の設定例を図 4-1 に示します。

```
#include "r_smip_api.h"

static void callback(uint32_t event);
extern const unsigned char image_jdi[11970];    /* JDI 形式の画像データ */
main()
{
    (void)R_SMIP_Open(1, 1000000, &g_smip_tbl_lcd_info[SMIP_TYPE_JDI] ,callback);
                            /* 使用する SPI チャネル：1、JDI 製 LCD、コールバック関数を指定 */
    (void)R_SMIP_PowerOn();    /* SMIP パワーオンシーケンスの実行 */
    while(1);
    {
        R_SYS_SoftwareDelay(1000, SYSTEM_DELAY_UNITS_MILLISECONDS);
        (void)R_SMIP_SendColor(0, 176, image_jdi);
                            /* 開始アドレス：0、データラインサイズ：176、画像データテーブルを指定*/
        R_SYS_SoftwareDelay(1000, SYSTEM_DELAY_UNITS_MILLISECONDS);
        (void)R_SMIP_SendCommand(0x20);    /* オールクリアコマンド */
    }
}

/***********************************************************************************************************
* callback function
***********************************************************************************************************/
static void callback(uint32_t event)
{
    /* すべての転送完了した際の処理を記述 */
    switch(event)
    {
        case SMIP_EVENT_SEND_COMPLETE:
        {
            /* 送信が正常に完了した場合の処理を記載 */
        }
        break;
        case SMIP_EVENT_SEND_ERROR:
        {
            /* 送信でエラーが発生した場合の処理を記載 */
        }
        break;
    }
}
```

図 4-1 SMIP LCD 設定および送信例（JDI 製）

4.1.2 JDI 製 MIP-LCD の画像データ

JDI のアドレスマップを図 4-2 に、SMIP_CFG_MAKE_LINE_INFO が 0 (送信時に SMIP ドライバがヘッダ、フッタを付与しない(注 1))の場合のデータ配置例を図 4-3 に、SMIP_CFG_MAKE_LINE_INFO が 1 (送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例を図 4-4 に示します。(注 2)

注1. SMIP_CFG_MAKE_LINE_INFO を 0 に設定した場合、ヘッダおよびフッタの自動生成機能が無効になります。ヘッダ、フッタを含めた画像データを用意する必要があります。

注2. ここでは SMIP_CFG_SPI_BIT_ORDER が 0(SPI のビットオーダーに MSB ファーストを選択)での配置例を示します。SMIP_CFG_SPI_BIT_ORDER を 1(SPI のビットオーダーに LSB ファーストを選択)の場合は、配置するデータのビット列を反転してください。

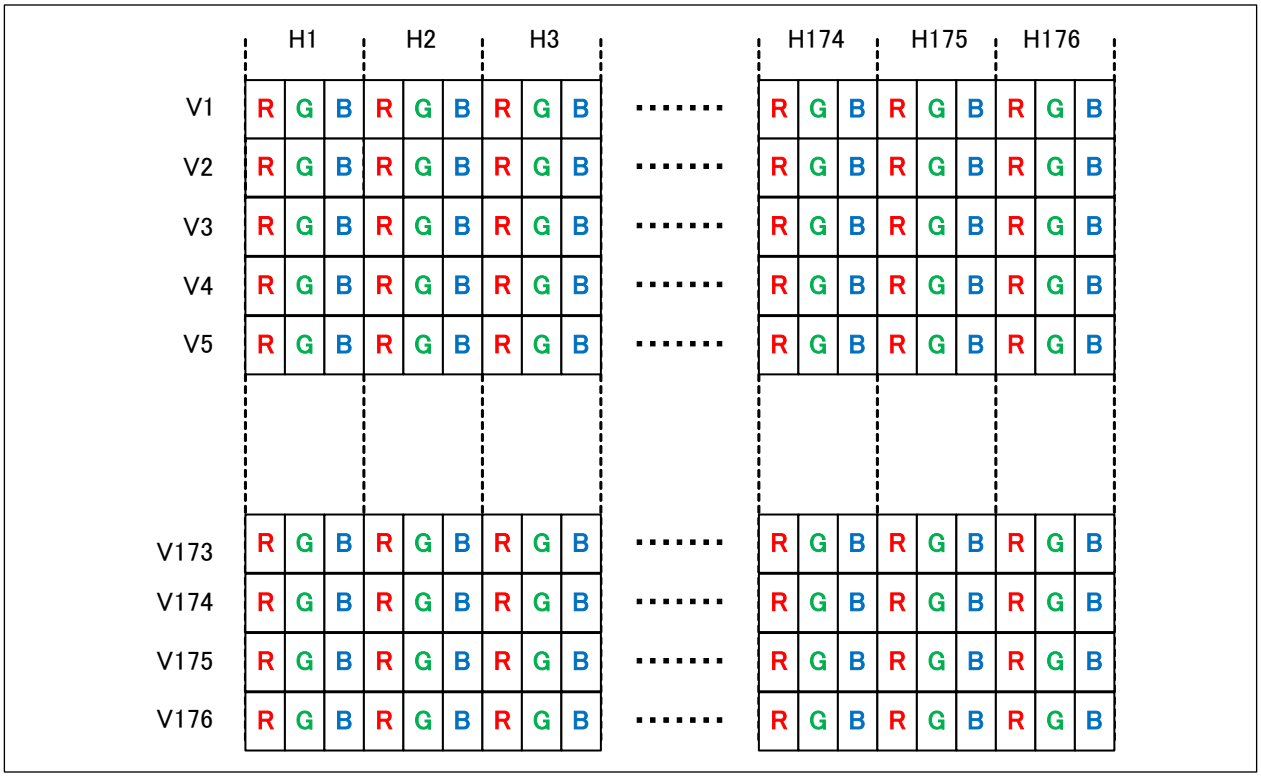


図 4-2 JDI 製 MIP-LCD のアドレスマップ（3 ビットカラー、176 x 176）

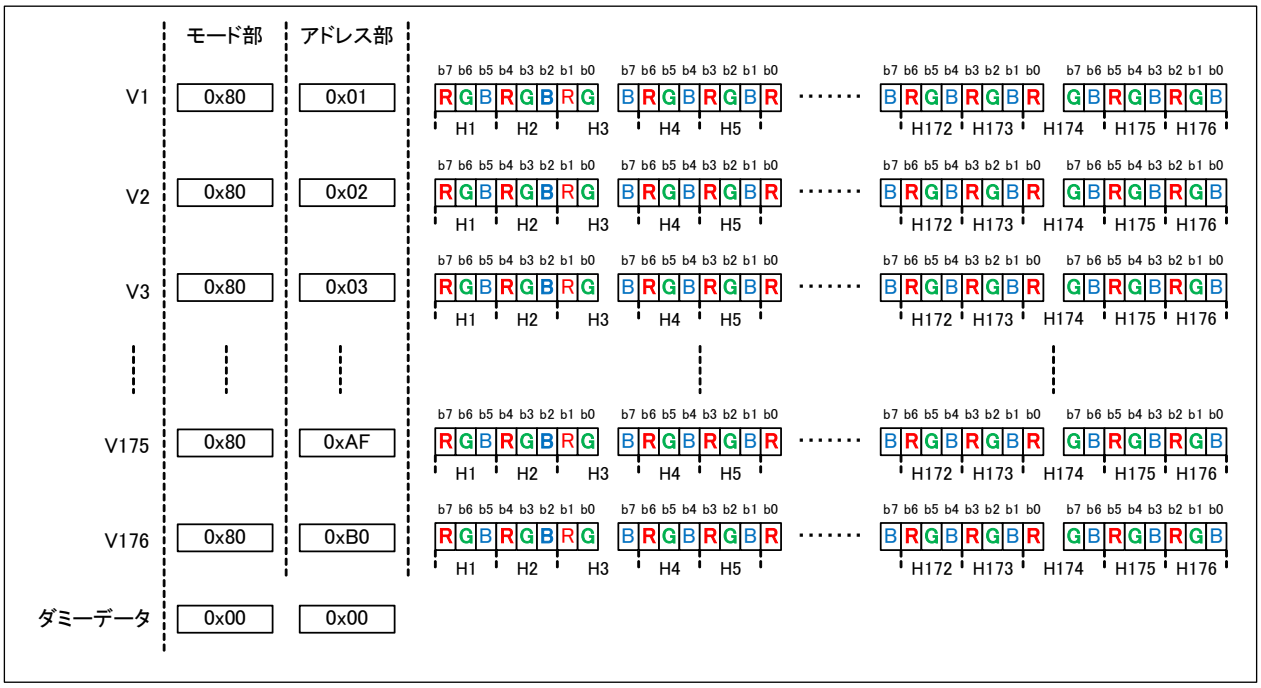


図 4-3　SMIP_CFG_MAKE_LINE_INFO が 0
(送信時に SMIP ドライバがヘッダ、フッタを付与しない)の場合のデータ配置例

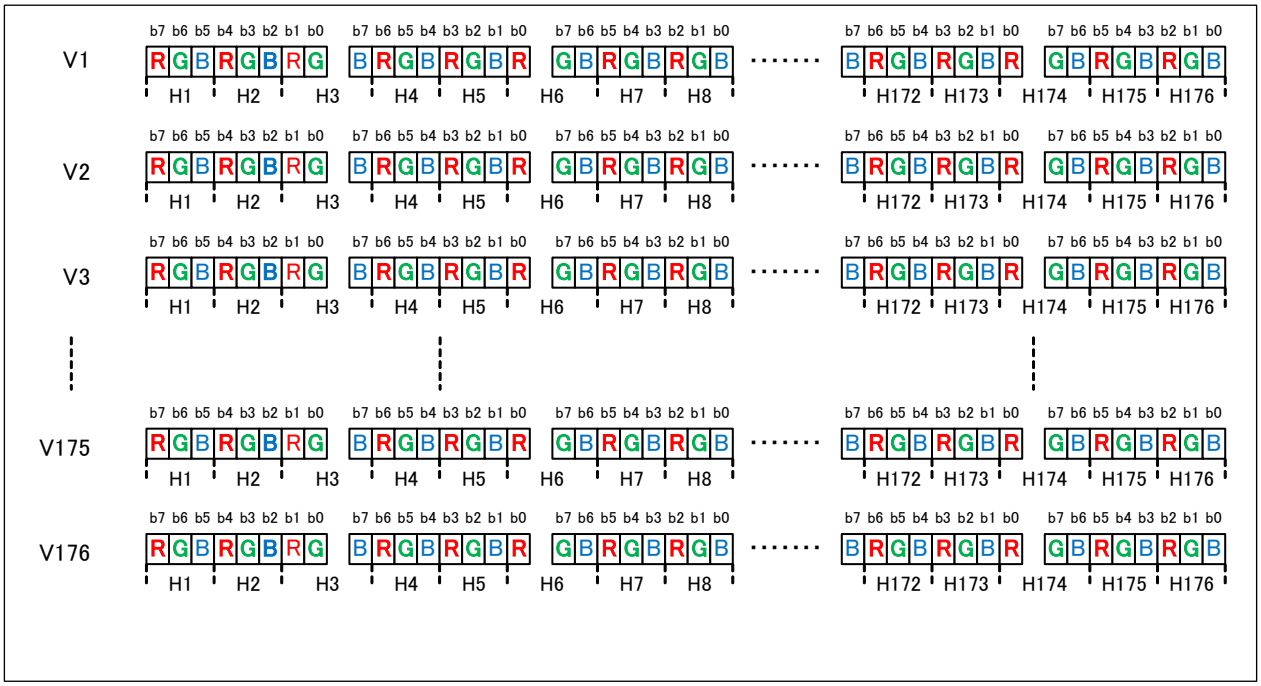


図 4-4　SMIP_CFG_MAKE_LINE_INFO が 1
(送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例

4.2 SHARP 製 MIP-LCD 使用例

4.2.1 SHARP 製 MIP-LCD の画像表示例

SHARP の設定は、r_smip_api.c の g_smip_tbl_lcd_info[SMIP_TYPE_SHARP]に定義されています。SHARP の設定定義については「4.6.16 SHARP 形式 LCD 設定定義」を参照してください。

SHARP を使用する場合の設定例を図 4-5 に示します。

```
#include "r_smip_api.h"

static void callback(uint32_t event);
extern const unsigned char image_sharp[2306];    /* SHARP 形式の画像データ */
main()
{
    (void)R_SMIP_Open(1, 1000000, &g_smip_tbl_lcd_info[SMIP_TYPE_SHARP] ,callback);
                              /* 使用する SPI チャネル：1、SHARP 製 LCD、コールバック関数を指定 */
    (void)R_SMIP_PowerOn();    /* SMIP パワーオンシーケンスの実行 */
    (void)R_SMIP_Send(0, 128, image_sharp);
                 /* 開始アドレス：0、データラインサイズ：128、画像データテーブルを指定*/
    while(1);
    {
        R_SYS_SoftwareDelay(1000, SYSTEM_DELAY_UNITS_MILLISECONDS);
        (void)R_SMIP_Send(0, 128, image_sharp);
                                   /* 開始アドレス：0、データラインサイズ：128、画像データテーブルを指定*/
        R_SYS_SoftwareDelay(1000, SYSTEM_DELAY_UNITS_MILLISECONDS);
        (void)R_SMIP_SendCommand(0x20);    /* オールクリアコマンド */
    }

    while(1);
}

/***********************************************************************************************
* callback function
***********************************************************************************************/
static void callback(uint32_t event)
{
    /* すべての転送完了した際の処理を記述 */
    switch(event)
    {
        case SMIP_EVENT_SEND_COMPLETE:
        {
            /* 送信が正常に完了した場合の処理を記載 */
        }
        break;
        case SMIP_EVENT_SEND_ERROR:
        {
            /* 送信でエラーが発生した場合の処理を記載 */
        }
        break;
    }
}
```

図 4-5　SMIP LCD 設定および送信例（SHARP 製）

4.2.2 SHARP 製 MIP-LCD の画像データ

SHARP のアドレスマップを図 4-6 に、SMIP_CFG_MAKE_LINE_INFO が 0 (送信時に SMIP ドライバがヘッダ、フッタを付与しない(注 1))の場合のデータ配置例を図 4-7 に、SMIP_CFG_MAKE_LINE_INFO が 1 (送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例を図 4-8 に示します。(注 2)

注1. SMIP_CFG_MAKE_LINE_INFO を 0 に設定した場合、ヘッダおよびフッタの自動生成機能が無効になります。ヘッダ、フッタを含めた画像データを用意する必要があります。

注2. ここでは SMIP_CFG_SPI_BIT_ORDER が 0(SPI のビットオーダーに MSB ファーストを選択)での配置例を示します。SMIP_CFG_SPI_BIT_ORDER を 1(SPI のビットオーダーに LSB ファーストを選択)の場合は、配置するデータのビット列を反転してください。

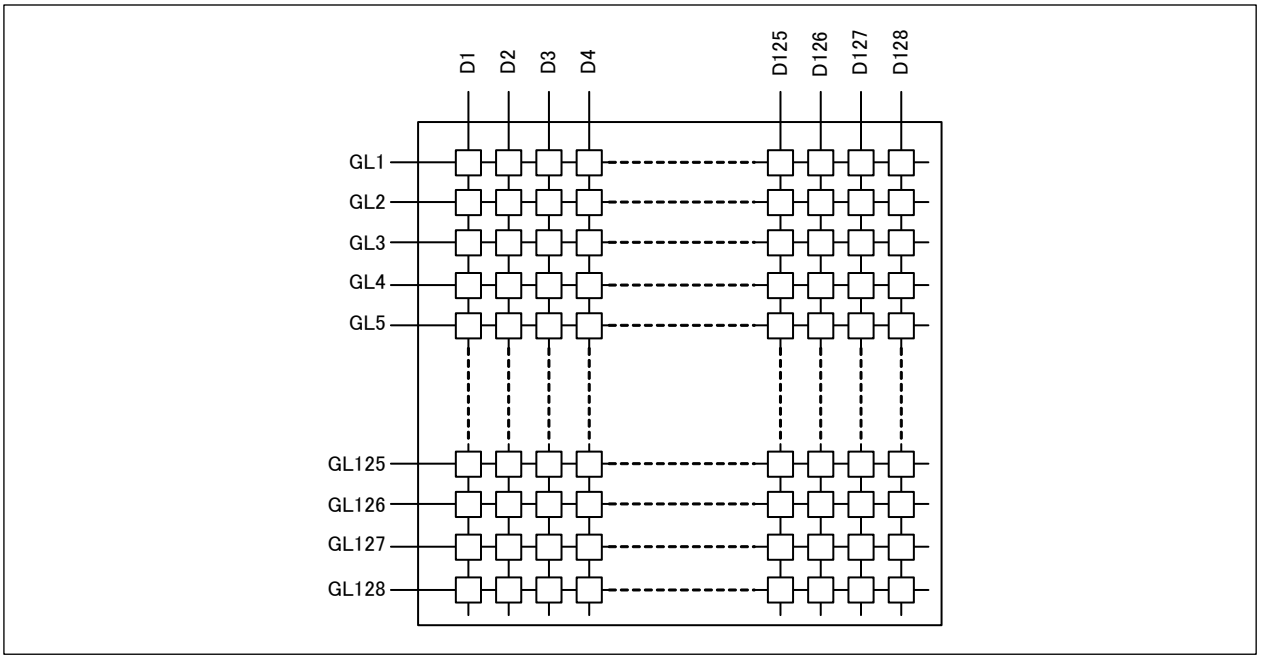


図 4-6 SHARP 製 MIP-LCD のアドレスマップ（モノクロ、128 x 128）

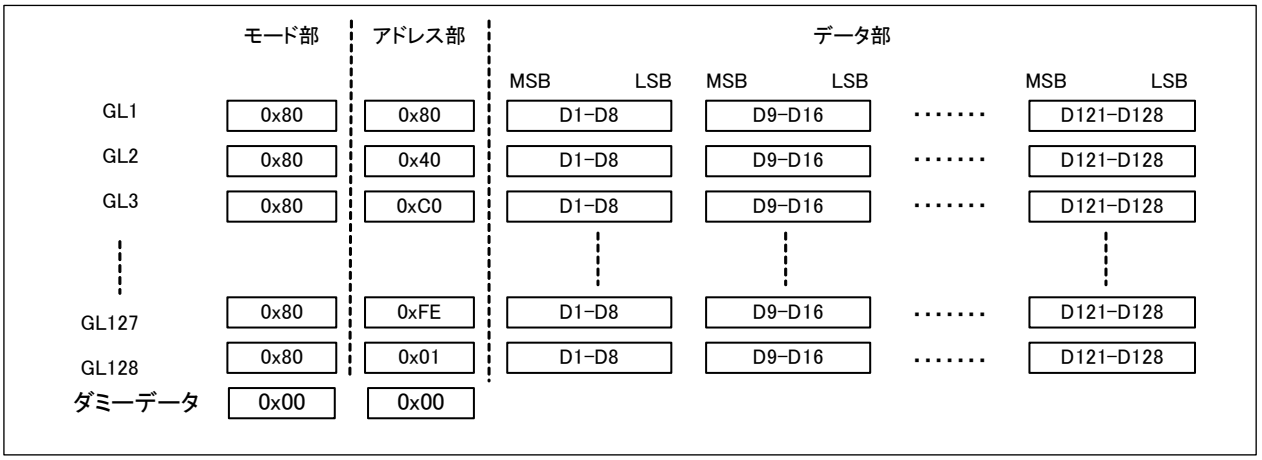


図 4-7 SMIP_CFG_MAKE_LINE_INFO が 0
(送信時に SMIP ドライバがヘッダ、フッタを付与しない)の場合のデータ配置例

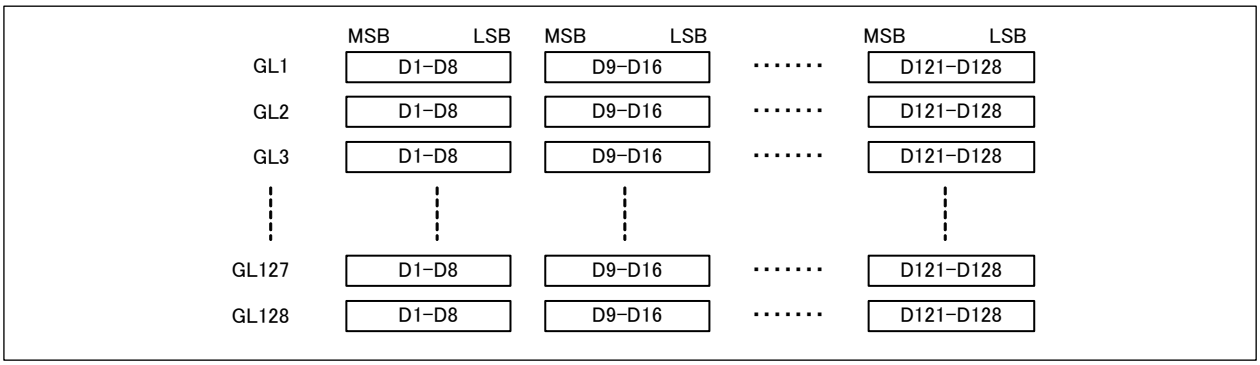


図 4-8　SMIP_CFG_MAKE_LINE_INFO が 1
(送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例

4.3　京セラ製 MIP-LCD 使用例

4.3.1　京セラ製 MIP-LCD の画像表示例

京セラの設定は、r_smip_api.c の g_smip_tbl_lcd_info[SMIP_TYPE_KYOCERA]に定義されています。京セラの設定定義については「4.6.15 京セラ形式 LCD 設定定義」を参照してください。

京セラ製 LCD を使用する場合の設定例を図 4-9 に示します。

```
#include "r_smip_api.h"

static void callback(uint32_t event);
extern const unsigned char image_kyocera[9472];    /* 京セラ形式の画像データ */
main()
{
    (void)R_SMIP_Open(1, 1000000, &g_smip_tbl_lcd_info[SMIP_TYPE_KYOCERA] ,callback);
                              /* 使用する SPI チャネル：1、京セラ製 LCD、コールバック関数を指定 */
    (void)R_SMIP_PowerOn();    /* SMIP パワーオンシーケンスの実行 */
    (void)R_SMIP_Send(0, 256, image_kyocera);
                              /* 開始アドレス：0、データラインサイズ：256、画像データテーブルを指定*/
    while(1);
}

/***********************************************************************************************************
* callback function
***********************************************************************************************************/
static void callback(uint32_t event)
{
    /* すべての転送完了した際の処理を記述 */
    switch(event)
    {
        case SMIP_EVENT_SEND_COMPLETE:
        {
            /* 送信が正常に完了した場合の処理を記載 */
        }
        break;
        case SMIP_EVENT_SEND_ERROR:
        {
            /* 送信でエラーが発生した場合の処理を記載 */
        }
        break;
    }
}
```

図 4-9　SMIP LCD 設定および送信例（京セラ製）

4.3.2 京セラ製 MIP-LCD の画像データ

京セラのアドレスマップを図 4-10 に、SMIP_CFG_MAKE_LINE_INFO が 0 (送信時に SMIP ドライバがヘッダ、フッタを付与しない(注 1))の場合のデータ配置例を図 4-11 に、SMIP_CFG_MAKE_LINE_INFO が 1 (送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例を図 4-12 に示します。

注1. SMIP_CFG_MAKE_LINE_INFO を 0 に設定した場合、ヘッダおよびフッタの自動生成機能が無効になります。ヘッダ、フッタを含めた画像データを用意する必要があります。(注 2)

注2. ここでは SMIP_CFG_SPI_BIT_ORDER が 0(SPI のビットオーダーに MSB ファーストを選択)での配置例を示します。SMIP_CFG_SPI_BIT_ORDER を 1(SPI のビットオーダーに LSB ファーストを選択)の場合は、配置するデータのビット列を反転してください。

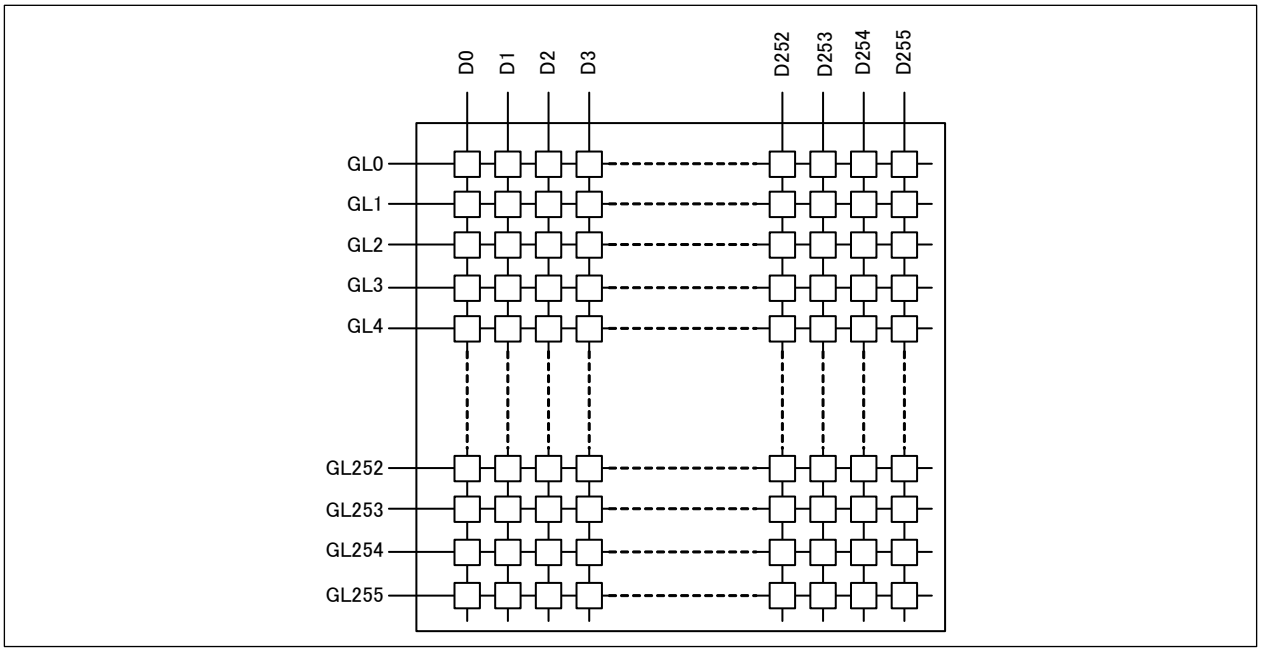


図 4-10 京セラ製 MIP-LCD のアドレスマップ（モノクロ、256 x 256）

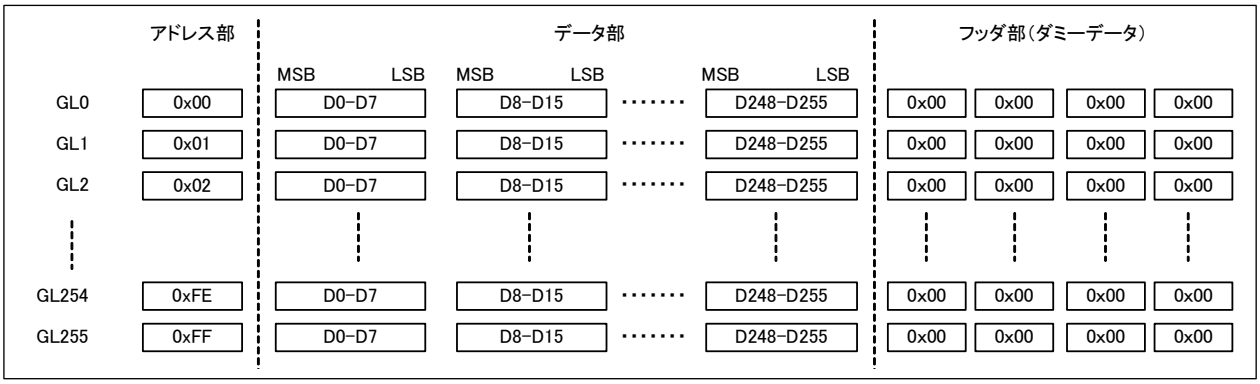


図 4-11 SMIP_CFG_MAKE_LINE_INFO が 0
(送信時に SMIP ドライバがヘッダ、フッタを付与しない)の場合のデータ配置例

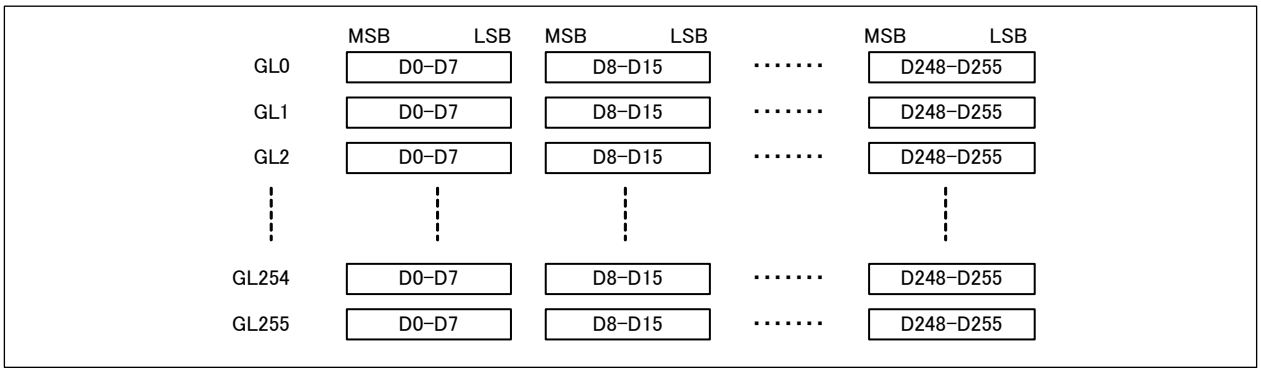


図 4-12　SMIP_CFG_MAKE_LINE_INFO が 1
(送信時に SMIP ドライバがヘッダ、フッタを付与する)の場合のデータ配置例

4.4　サスペンドモードへの遷移

消費電力低減機能を使用して SPI 機能への電源供給を遮断する場合、事前にサスペンドモードに遷移させることで LCD の表示を維持します。サスペンドモードへ遷移させる場合、R_SMIP_Suspend 関数を使用します(注 1)。サスペンドモード遷移時、SPI の設定および SPI 使用端子は解放されます(注 2)。サスペンドモードから SPI 再設定を行う場合、R_SMIP_Resume 関数を使用します。

サスペンドモードにおける各端子の状態を表 4-1 に、R_SMIP_Suspend 関数によるサスペンドモードへの遷移例を図 4-13 に示します。

注1. SPI 通信による LCD 表示処理中に実行すると、R_SMIP_ERROR_BUSY が返ります

注1. サスペンドモード時、以下の関数は使用できません
画像データ出力を行う関数（R_SMIP_AllZero、R_SMIP_AllOne、R_SMIP_Send、R_SMIP_SendColor）、R_SMIP_SendCommand 関数、R_SMIP_PowerOn 関数、R_SMIP_PowerOff 関数

表 4-1 サスペンドモードにおける各端子状態一覧

端子	状態
DISP(RST)	移行前の状態を保持
EXTCOMIN(VCOM)	VCOM 出力反転(注)
SCS	L 出力
SI,CLK	解放状態

注　以下のいずれかの条件の場合、VCOM 出力は移行前の状態を保持します（反転出力は行いません）
・消費電力低減機能にて ISO1 ドメイン（AGTn 機能）への電源供給を遮断した場合
・AGTOn 出力機能が無効（r_smip_cfg.h のマクロ"SMIP_CFG_AGTO_EN" が 0）かつ、CPU が停止(スタンバイモード中)している場合

```
#include "r_smip_api.h"

static void callback(uint32_t event);
extern const unsigned char image_sharp[2306];    /* SHARP 形式の画像データ */
main()
{
    /* サスペンドモードへの遷移は、Open による初期化後に実施可能 */
    (void)R_SMIP_Open(1, 5000000, &g_smip_tbl_lcd_info[SMIP_TYPE_SHARP] ,callback);
    /* 使用する SPI チャネル：1、SHARP 製 LCD、コールバック関数を指定 */
    (void)R_SMIP_PowerOn();    /* SMIP パワーオンシーケンスの実行 */
    (void)R_SMIP_Send(0, 128, image_sharp);
                               /* 開始アドレス：0、データラインサイズ：128、画像データテーブルを指定*/
    while(1);
    {
        ・・・
        /* SPI 機能への電源供給遮断前に、サスペンドモードへ遷移 */
        while (SMIP_OK != R_SMIP_Suspend())
        {
            /* SPI による LCD 表示データ送信処理中の場合、
               SMIP_ERROR_BUSY となりサスペンドモードへの遷移不可 */
            R_SYS_SoftwareDelay(1, SYSTEM_DELAY_UNITS_MILLISECONDS);
        }
        /* サスペンドモードに遷移後、LCD の表示は維持*/
        ・・・
        /* SPI 機能への電源供給復帰後、LCD の表示を更新させる場合は
           Resume 関数にて SPI 再設定および SPI への端子割り当てを実施 */
        (void)R_SMIP_Resume(100000);  /* 引数には SPI の通信速度を指定(R_SMIP_Open 関数での指定値より変更可)
                                         SPI 通信速度設定が不正な場合は SMIP_ERROR_PARAMETER が返る */
        ・・・
    }
}
```

図 4-13　R_SMIP_Suspend 関数によるサスペンドモードへの遷移例

4.5　SPI 通信速度の変更

SPI の通信速度を変更する場合、もしくは SPI のカウントソースが変更された場合、R_SMIP_ReconfigSpiSpeed 関数にて通信速度を再設定します(注)。R_SMIP_ReconfigSpiSpeed 関数による SPI 通信速度再設定例を図 4-14 に示します。

注　SPI 通信による LCD 表示処理中に実行すると、R_SMIP_ERROR_BUSY が返ります

```
#include "r_smip_api.h"

static void callback(uint32_t event);
extern const unsigned char image_sharp[2306];    /* SHARP 形式の画像データ */
main()
{
    /* SPI 通信速度の変更は、Open による初期化後に実施可能 */
    (void)R_SMIP_Open(1, 1000000, &g_smip_tbl_lcd_info[SMIP_TYPE_SHARP] ,callback);
    ・・・
    while(1);
    {
        ・・・
        /* SPI 通信速度を 500kHz に変更 */
        /* SPI 通信速度設定が不正な場合は、SMIP_ERROR_PARAMETER が返る */
        while (SMIP_OK != R_SMIP_ReconfigSpiSpeed(500000))
        {
            /* SPI による LCD 表示データ送信処理中の場合、SPI の通信速度は変更不可 */
            R_SYS_SoftwareDelay(1, SYSTEM_DELAY_UNITS_MILLISECONDS);
        }
        ・・・
    }
`
```

図 4-14　R_SMIP_ReconfigSpiSpeed 関数による SPI 通信速度再設定例

4.6 コンフィグレーション

SMIP ドライバは、ユーザが設定可能なコンフィグレーションを r_smip_cfg.h ファイルに用意します。

4.6.1 パラメータチェック

SMIP ドライバにおけるパラメータチェックの有効/無効を設定します。

名称：SMIP_CFG_PARAM_CHECKING_ENABLE

表 4-2 SMIP_CFG_PARAM_CHECKING_ENABLE の設定

設定値	内容
0	パラメータチェックを無効にします 引数の妥当性判定に関するエラーの検出を行いません
1（初期値）	パラメータチェックを有効にします 引数の妥当性判定に関するエラーの検出を行います

4.6.2 SCS"L"期間(tw_scsl)待ち処理設定定義

SCS"L"期間(tw_scsl)待ち処理の方法を設定します。

名称： SMIP_CFG_SCS_WAIT_EN

表 4-3 SMIP_CFG_SCS_WAIT_EN の設定

設定値	内容
0	SCS"L"期間(tw_scsl)待ち処理を SMIP ドライバ内で実施しない(注)
1（初期値）	SCS"L"期間(tw_scsl)待ち処理を SMIP ドライバ内で実施する

注　コールバック関数などで連続してデータを送信する場合は、ユーザープログラムにて SCS"L"幅(twSCSL)を満たす待ち時間を挿入してください。

4.6.3 コマンド最大バイト数定義

コマンド送信時の最大バイト数を設定します。

名称： SMIP_CFG_COMMAND_MAX_BYTE

表 4-4 SMIP_CFG_COMMAND_MAX_BYTE の設定

設定値	内容
2（初期値）	コマンド最大バイト数

4.6.4 1 ライン送信用最大バッファサイズ定義

1 ライン送信時の最大バッファサイズを設定します。

名称： SMIP_CFG_ALL_WRITE_BUFF

表 4-5 SMIP_CFG_ALL_WRITE_BUFF の設定

設定値	内容
37（初期値）	1 ライン送信用の最大バッファサイズ

4.6.5 EXTMODE 端子設定定義

EXTMODE 端子の入力レベルを設定します。京セラでは、1 を設定してください。

名称： SMIP_CFG_EXTMODE

表 4-6 SMIP_CFG_EXTMODE の設定

設定値	内容
0	EXTMODE 端子入力レベルが"L"(EXTCOMIN 出力を行わない)(注)
1（初期値）	EXTMODE 端子入力レベルが"H"(EXTCOMIN 出力を行う)

注　COM 入力は SendCommand 関数を使用して入力してください

4.6.6 VCOM(EXTCOMIN)出力制御用 AGTO 出力設定定義

VCOM(EXTCOMIN)制御の AGTO 出力を有効もしくは無効に設定します。

名称： SMIP_CFG_AGTO_EN

表 4-7 SMIP_CFG_AGTO_EN の設定

設定値	内容
0（初期値）	VCOM(EXTCOMIN)制御にポートを使用
1	VCOM(EXTCOMIN)制御に AGTOn 出力を使用

4.6.7 VCOM(EXTCOMIN)出力端子設定定義

VCOM(EXTCOMIN)出力に使用する端子を定義します。

表 4-8 VCOM 出力端子設定定義一覧

定義名	初期値	内容
SMIP_CFG_VCOM_PORT	PORT1	VCOM(EXTCOMIN)端子に PORT1 を選択
SMIP_CFG_VCOM_PIN	6	VCOM(EXTCOMIN)端子に PORTi06 を選択 (i は SMIP_CFG_VCOM_PORT で指定したポート)

4.6.8 SCS 出力端子設定定義

SCS 出力に使用する端子を定義します。

表 4-9 SCS 設定定義一覧

定義名	初期値	内容
SMIP_CFG_SCS_PORT	PORT1	SCS 端子に PORT1 を選択
SMIP_CFG_SCS_PIN	3	SCS 端子に PORTi03 を選択 (i は SMIP_CFG_SCS_PORT で指定したポート)

4.6.9 RST、DISP 出力端子設定定義

RST 出力に使用する端子を定義します。

表 4-10 RST、DISP 設定定義一覧

定義名	初期値	内容
SMIP_CFG_RST_DISP_PORT	PORT1	RST 端子に PORT1 を選択
SMIP_CFG_RST_DISP_PIN	5	RST 端子に PORTi05 を選択 (i は SMIP_CFG_RST_PORT で指定したポート)

4.6.10 送信データ情報生成定義

送信データ情報のヘッダおよびフッタ自動生成機能を有効もしくは無効に設定します。ヘッダおよびフッタの自動生成機能による送信データ情報の違いについては「4 ドライバ動作説明」を参照してください。

名称： SMIP_CFG_MAKE_LINE_INFO

表 4-11 SMIP_CFG_MAKE_LINE_INFO の設定

設定値	内容
0	送信時にヘッダ、フッタを付与しない（SMIP ドライバでヘッダ、フッタを自動生成しない）
1（初期値）	送信時にヘッダ、フッタを付与する（SMIP ドライバでヘッダ、フッタを自動生成する）

4.6.11 VCOM(EXTCOMIN)タイマ設定用定義

VCOM(EXTCOMIN)制御に AGT タイマを使用する場合の、AGT タイマ設定を定義します。

表 4-12 VCOM(EXTCOMIN)タイマ設定定義一覧

定義名	初期値	内容
SMIP_CFG_VCOM_TIMER_CH	0	VCOM(EXTCOMIN)用タイマの AGT チャネル （0 または 1）
SMIP_CFG_VCOM_TIMER_INT_PRIORITY	3	VCOM(EXTCOMIN)用タイマの割り込み （AGTn_AGTI）優先レベル 設定可能範囲：0～3 （最高優先度：0、最低優先度：3）
SMIP_CFG_VCOM_TIMER_CLOCK	0	AGT タイマのベースクロック 0：サブクロック(SOCO) 1：低速オンチップオシレータ(LOCO)
SMIP_CFG_VCOM_TIMER_LPM	1	AGT タイマの低消費電力モード設定 0：通常モード 1：低消費電力モード

4.6.12 タイムアウト値設定定義

パワーオンおよびパワーオフシーケンスのタイムアウト時間を指定します

名称： SMIP_CFG_TIMEOUT

表 4-13 SMIP_CFG_TIMEOUT の設定

設定値	内容
2000	パワーオンおよびパワーオフシーケンスのタイムアウト時間

4.6.13 SPI 通信のビットオーダー設定定義

SPI 通信で使用するビットオーダーを指定します

名称： SMIP_CFG_SPI_BIT_ORDER

表 4-14 SMIP_CFG_SPI_BIT_ORDER の設定

設定値	内容
0(初期値)	MSB ファースト
1	LSB ファースト

4.6.14 JDI 製 LCD 設定定義

Open 関数にて JDI を指定した場合、JDI 製 LCD 設定定義の設定で動作します。JDI 製 LCD の設定は、st_smip_cfg_t 構造体テーブルの g_smip_tbl_lcd_info[SMIP_TYPE_JDI]に定義されています。以下に JDI 製 LCD 設定定義一覧を示します。

表 4-15 JDI 製 LCD 設定定義一覧

定義名	初期値	内容	st_smip_cfg_t メンバ
SMIP_CFG_JDI_COLOR_MODE	0	カラーモード設定 0：3 ビットカラーモード 1：4 ビットカラーモード	type:b0
SMIP_CFG_JDI_MODE_SIZE	6	モードビットサイズ	mode_size
SMIP_CFG_JDI_MODE_DMY_SIZE	0	モードダミービットサイズ	mode_dmy_size
SMIP_CFG_JDI_ADDR_SIZE	10	アドレスビットサイズ	address_size
SMIP_CFG_JDI_DUMMY_SIZE	16	ダミービットサイズ	end_dummy_size
SMIP_CFG_JDI_DISP_WIDTH	176	LCD 幅	disp_width
SMIP_CFG_JDI_DISP_LINE	176	LCD ラインサイズ	disp_line
SMIP_CFG_JDI_VCOM_CLK	120 * 10	VCOM(EXTCOMIN)周期(×0.1Hz)	vcom_clk
SMIP_CFG_JDI_T_MEMORY_INI	1000	ピクセルメモリ初期化時間(us)	t_memory_init
SMIP_CFG_JDI_T_COM_INI	30	COM 初期化時間(us)	t_com_init
SMIP_CFG_JDI_TS_SCS	6	SCS のセットアップ時間(us)	ts_scs
SMIP_CFG_JDI_TH_SCS	2	SCS のホールド時間(us)	th_scs
SMIP_CFG_JDI_TW_SCSL	6	SCS の Low 時間(us)	tw_scsl
SMIP_CFG_JDI_FIRST_ADDR	1	初期ラインアドレス	first_addr

4.6.15 京セラ形式 LCD 設定定義

Open 関数にて京セラ形式の LCD を指定した場合、京セラ形式 LCD 設定定義の設定で動作します。京セラ製 LCD の設定は、st_smip_cfg_t 構造体テーブルの g_smip_tbl_lcd_info[SMIP_TYPE_KYOCERA]に定義されています。以下に京セラ形式 LCD 設定定義一覧を示します。

表 4-16 京セラ形式 LCD 設定定義一覧

定義名	初期値	内容	st_smip_cfg_t メンバ
SMIP_CFG_KYOCERA_COLOR_MODE	0	カラーモード 0: 3 ビットカラーモード 1: 4 ビットカラーモード	type:b0
SMIP_CFG_KYOCERA_MODE_SIZE	0	モードビットサイズ	mode_size
SMIP_CFG_KYOCERA_MODE_DMY_SIZE	0	モードダミービットサイズ	mode_dmy_size
SMIP_CFG_KYOCERA_ADDR_SIZE	8	アドレスビットサイズ	address_size
SMIP_CFG_KYOCERA_DUMMY_SIZE	32	ダミービットサイズ	end_dummy_size
SMIP_CFG_KYOCERA_DISP_WIDTH	256	LCD 幅	disp_width
SMIP_CFG_KYOCERA_DISP_LINE	256	LCD ラインサイズ	disp_line
SMIP_CFG_KYOCERA_VCOM_CLK	0.5 * 10	VCOM 周期(×0.1Hz)	vcom_clk
SMIP_CFG_KYOCERA_T_MEMORY_INI	1000	ピクセルメモリ初期化時間(us)	t_memory_init
SMIP_CFG_KYOCERA_T_COM_INI	1000	COM 初期化時間(us)	t_com_init
SMIP_CFG_KYOCERA_TS_SCS	4	SCS のセットアップ時間(us)	ts_scs
SMIP_CFG_KYOCERA_TH_SCS	4	SCS のホールド時間(us)	th_scs
SMIP_CFG_KYOCERA_TW_SCSL	10	SCS の Low 時間(us)	tw_scsl
SMIP_CFG_KYOCERA_TS_VCOM	4	VCOM のセットアップ時間(us)	ts_vcom
SMIP_CFG_KYOCERA_TH_VCOM	1	VCOM のホールド時間(us)	th_vcom
SMIP_CFG_KYOCERA_FIRST_ADDR	0	初期ラインアドレス	first_addr

4.6.16 SHARP 形式 LCD 設定定義

Open 関数にて SHARP 形式の LCD を指定した場合、SHARP 形式 LCD 設定定義の設定で動作します。SHARP 製 LCD の設定は、st_smip_cfg_t 構造体テーブルの g_smip_tbl_lcd_info[SMIP_TYPE_SHARP]に定義されています。以下に SHARP 形式 LCD 設定定義一覧を示します。

表 4-17 SHARP 式 LCD 設定定義一覧

定義名	初期値	内容	st_smip_cfg_t メンバ
SMIP_CFG_SHARP_COLOR_MODE	0	カラーモード 0: 3 ビットカラーモード 1: 4 ビットカラーモード	type:b0
SMIP_CFG_SHARP_MODE_SIZE	3	モードビットサイズ	mode_size
SMIP_CFG_SHARP_MODE_DMY_SIZE	5	ヘッダダミービットサイズ	mode_dmy_size
SMIP_CFG_SHARP_ADDR_SIZE	8	アドレスビットサイズ	address_size
SMIP_CFG_SHARP_DUMMY_SIZE	16	ダミービットサイズ	end_dummy_size
SMIP_CFG_SHARP_DISP_WIDTH	128	LCD 幅	disp_width
SMIP_CFG_SHARP_DISP_LINE	128	LCD ラインサイズ	disp_line
SMIP_CFG_SHARP_VCOM_CLK	60*10	VCOM(EXTCOMIN)周期 (×0.1Hz)	vcom_clk
SMIP_CFG_SHARP_T_MEMORY_INI	0	ピクセルメモリ初期化時間 (us)	t_memory_init
SMIP_CFG_SHARP_T_COM_INI	30	COM 初期化時間(us)	t_com_init
SMIP_CFG_SHARP_TS_SCS	6	SCS のセットアップ時間 (us)	ts_scs
SMIP_CFG_SHARP_TH_SCS	2	SCS のホールド時間(us)	th_scs
SMIP_CFG_SHARP_TW_SCSL	6	SCS の Low 時間(us)	tw_scsl
SMIP_CFG_SHARP_FIRST_ADDR	1	初期ラインアドレス	first_addr

4.6.17 関数の RAM 配置

SMIP ドライバの特定関数を RAM で実行するための設定を行います。

関数の RAM 配置を設定するコンフィグレーションは、関数ごとに定義を持ちます。

名称：SMIP_CFG_SECTION_xxx

xxx には関数名をすべて大文字で記載

例） R_SMIP_Open 関数 → SMIP_CFG_R_SMIP_OPEN

表 4-18 SMIP_CFG_SECTION_xxx の設定

設定値	内容
SYSTEM_SECTION_CODE	関数を RAM に配置しません
SYSTEM_SECTION_RAM_FUNC	関数を RAM に配置します

表 4-19 各関数の RAM 配置初期状態

番号	関数名	RAM 配置
1	R_SMIP_GetVersion	
2	R_SMIP_Open	
3	R_SMIP_Close	
4	R_SMIP_PowerOn	
5	R_SMIP_PowerOff	
6	R_SMIP_AllZero	
7	R_SMIP_AllOne	
8	R_SMIP_Send	
9	R_SMIP_SendColor	
10	R_SMIP_SendCommand	
11	R_SMIP_Suspend	
12	R_SMIP_Resume	
13	R_SMIP_ReconfigSpiSpeed	
14	r_smip_cb_spi_event	✔
15	r_smip_cb_vcom_timer_event	✔

5. 使用上の注意

5.1 NVIC への AGTI 割り込み登録

SMIP_CFG_EXTMODE が 1 もしくは、送信制御が VCOM 出力に同期する場合、AGT タイマ割り込み（AGTn_AGTI）設定が必要です。詳細は「1.2 SMIP で使用する AGT の設定」を参照してください。

5.2 SPI ドライバの設定について

本ドライバでは SPI ドライバを使用しています。SPI の設定については「1 概要」を参照してください。

5.3 関数の実行制限

本ドライバで画像データ出力を行う関数（R_SMIP_AllZero、R_SMIP_AllOne、R_SMIP_Send、R_SMIP_SendColor）、R_SMIP_SendCommand 関数は R_SMIP_PowerOn 関数実行後にのみ使用できます。

5.4 RST、SCS および VCOM 制御端子設定について

本ドライバで使用する RST および SCS 信号は、ポート出力によって制御されます。r_smip_cfg.h にて指定した RST および SCS 制御端子は、周辺機能に割り当てないでください。端子の周辺機能設定は、pin.c にて変更できます。同様に、VCOM 信号に AGTOn を使用しない場合、VCOM 制御端子を周辺機能に割り当てないでください。

5.5 R_SMIP_ReconfigSpiSpeed 関数実行中に Suspend 関数を実行した場合の動作について

R_SMIP_ReconfigSpiSpeed 関数による SPI 通信速度変更時に、割り込みで R_SMIP_Suspend 関数を実行した場合、不正な動作となる恐れがあります。R_SMIP_ReconfigSpiSpeed 関数実行中は、R_SMIP_Suspend 関数を実行しないでください。

6. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RE01 1500KB グループ ユーザーズマニュアル　ハードウェア編 R01UH0796
RE01 256KB グループ ユーザーズマニュアル　ハードウェア編 R01UH0894

（最新版をルネサス エレクトロニクスホームページから入手してください。）

RE01 グループ CMSIS Package スタートアップガイド

RE01 1500KB、256KB グループ CMSIS パッケージを用いた開発スタートアップガイド R01AN4660

（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース

（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：開発環境

（最新版をルネサス エレクトロニクスホームページから入手してください。）

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Oct.10.2019	ー	初版
1.01	Oct.30.2019	プログラム	以下の関数を追加 ・R_SMIP_Suspend ・R_SMIP_Resume ・R_SMIP_ReconfigSpiSpeed
		ー	誤記修正
		6, 7	SCS(RST)信号の注意事項および端子設定例を修正
		6~9	pin.c のデフォルト端子設定コメントアウト化にともなう修正
		9	ドライバ API の追加 ・R_SMIP_Suspend ・R_SMIP_Resume ・R_SMIP_ReconfigSpiSpeed
		14	「サスペンド状態」の追加
		22	「サスペンドモードへの遷移」を追加
		23	「SPI 通信速度の変更」を追加
		32	「R_SMIP_ReconfigSpiSpeed 関数実行中に Suspend 関数を実行した場合の動作について」を追加
1.03	Mar.4.2020	4, 18, 21, 23, 30	SPI 通信のビットオーダー設定を追加
		21~24	データ配列の誤記修正
		プログラム	SPI 通信のビットオーダー設定(SMIP_CFG_SPI_BIT_ORDER)を追加 内部関数 r_smip_tm_lpm_on()の AGT モジュールストップ処理を削除
1.04	Mar.5.2020	ー	256KB グループに対応
		プログラム (256KB)	
1.05	Apr.17.2020	プログラム (1500KB)	ToolNews R20TS0574 の修正を実施
		プログラム (256KB)	京セラ LCD 使用時に AGT レジスタ値が書き換えられない問題を修正
		プログラム	R_SMIP_Resume 関数実行時、SPI 通信のビットオーダーに SMIP_CFG_SPI_BIT_ORDER の設定が反映されるよう修正
		-	誤記修正

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替および外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。