

Traveo™ Family クロックシステムの設定方法

著者: Hachisu, Kotaro

関連製品ファミリ: Traveo Family

S6J3110/3120/3200/3310/3320/3330/3340/3350/3360/3370/3400/3510 シリーズ

関連ドキュメント: [関連ドキュメント](#)

本アプリケーションノートでは、Traveo™ family S6J3110/3120/3200/3310/3320/3330/3340/3350/3360/3370/3400/3510 シリーズのクロックシステムの設定方法を説明します。

目次

1 はじめに	1	3.2 クロックシステム設定手順 (PLL 動作)	12
2 クロックシステム概要	1	3.3 クロック設定手順例	15
2.1 クロック制御部	2	4 関連ドキュメント	19
2.2 クロック生成部	2	4.1 Datasheets	19
2.3 クロック分周/分配部	5	4.2 Hardware Manuals	20
2.4 周辺機能とソースクロックの組み合わせ確認方法 ..	7	改訂履歴	21
3 クロックシステムの設定方法	10	セールス、ソリューションおよび法律情報	22
3.1 クロックシステム設定方法 (RUN 動作)	10		

1 はじめに

本アプリケーションノートでは、S6J3110/3120/3200/3310/3320/3330/3340/3350/3360/3370/3400/3510 シリーズのクロックシステム設定方法について記載します。

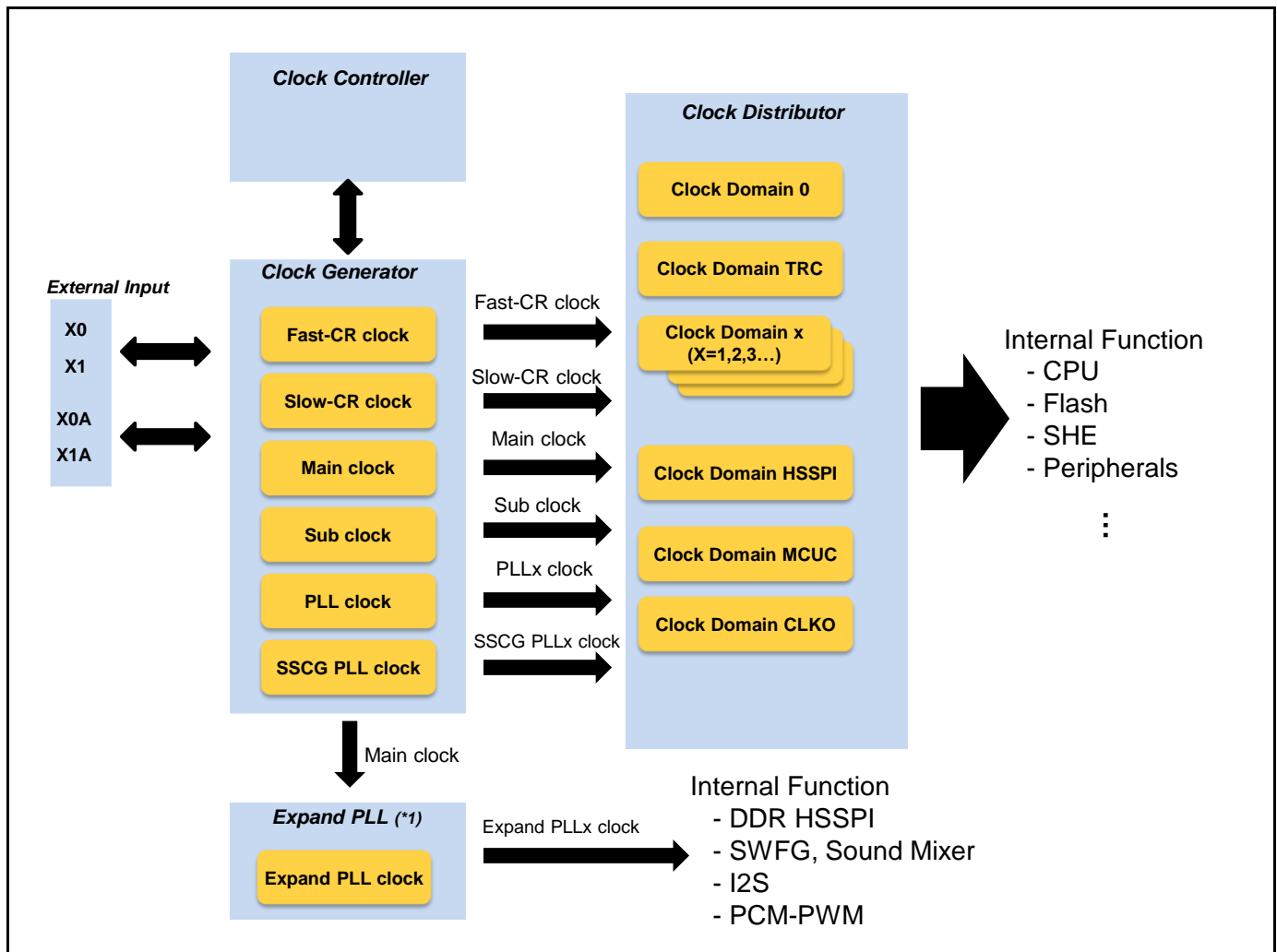
2 クロックシステム概要

本章ではクロックシステムの概要について説明します。図 1 のようにクロックシステムは 4 つのモジュールで構成されています。

- クロック制御部
- クロック生成部
- クロック分周/分配部
- 拡張 PLL¹

¹ 拡張 PLL は S6J3360/ S6J3370/ S6J3510 シリーズのみ搭載しています。

図 1. クロックシステム概要図



2.1 クロック制御部

クロック制御部はレジスタ設定値に基づいて、クロックシステムに制御信号を生成します。

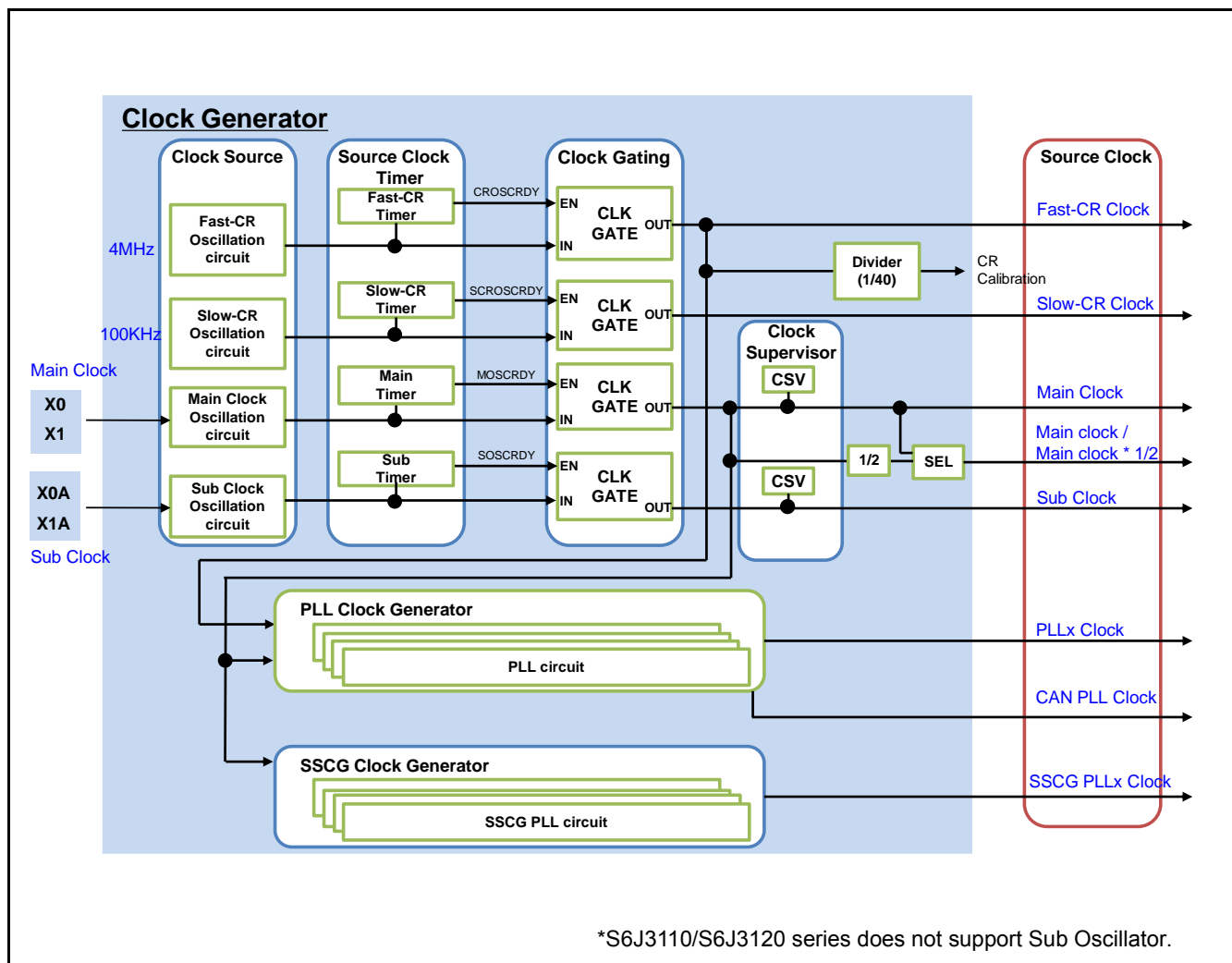
2.2 クロック生成部

図 2 にクロック生成部のブロックダイアグラムを示します。クロック生成部はマイコンの発振クロックからソースクロックを生成します。Traveo Family マイコンは複数のクロックソースをサポートしています。

- メインクロック
- 高速 CR クロック
- 低速 CR クロック
- サブクロック

各クロックの周波数については、対象製品のデータシートを参照してください。すべてのクロックソースには、クロックの発振安定待ち時間の間、クロック出力をゲーティングするタイマ（ソースクロックタイマ）が用意されています。低消費電力で動作させるため、クロックゲーティング機能を使用して、内部回路へのクロック供給を止めることも可能です。

図 2. クロック生成部 ブロックダイアグラム



クロック生成部には複数の PLL 発振回路を内蔵しており、メインクロックもしくは高速 CR クロックから PLL クロックを生成します。PLL 使用時、PLL クロックの通倍/分周設定は、レジスタ設定(PLLDIVL/PLLDIVN/PLLDIVM)により行います。図 3 に S6J3110/3120 シリーズの PLL クロック生成部の構成、表 1 に PLL クロック設定例を示します。

図 3. PLL クロック生成部

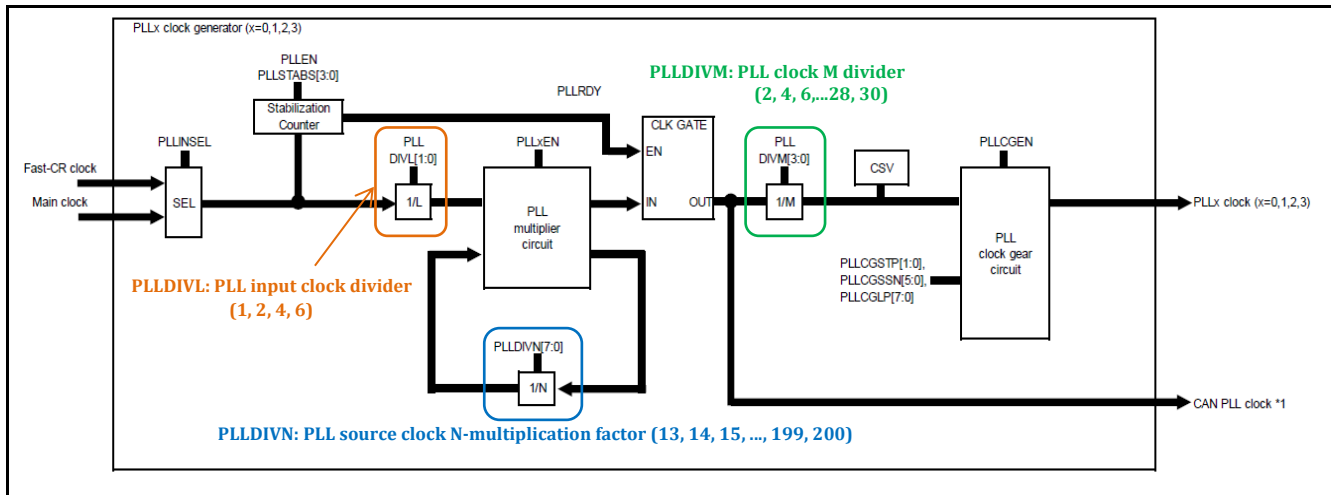


表 1. PLL クロック通倍/分周設定例

Input Clock	Input Clock Divider Configuration (Set by PLLDIVL)	PLLIn	PLL Multiplier Configuration (Set by PLLDIVN)	PLLout	Output Divider Configuration (Set by PLLDIVM)	PLL Clock
4 MHz	1	4 MHz	120	480 MHz	2	240 MHz
4 MHz	1	4 MHz	144	576 MHz	4	144 MHz
4 MHz	1	4 MHz	120	480 MHz	4	120 MHz
4 MHz	1	4 MHz	144	576 MHz	6	96 MHz
4 MHz	1	4 MHz	120	480 MHz	6	80 MHz

PLL クロックの周波数は以下の式で算出できます。

$$\text{PLLIn} = (\text{Input clock}) / (\text{divider setting by PLLDIVL})$$

$$\text{PLLout} = \text{PLLIn} \times (\text{multiplier setting by PLLDIVN})$$

$$\text{PLL clock} = \text{PLLout} / (\text{divider setting by PLLDIVM})$$

PLL の出力クロック (PLLout) の周波数許容範囲は製品型格ごとに異なります。下記の範囲内になるようクロック通倍設定を行ってください。

- S6J3118, S6J3119, S6J311A, S6J311B, S6J311C, S6J311D, S6J311E: 400 MHz ~ 576 MHz
- S6J3128, S6J3129, S6J312A: 400 MHz ~ 512 MHz
- S6J3200 シリーズ: 400 MHz ~ 型格オプション仕様²
- S6J3300, S6J3350 シリーズ: 200 MHz ~ 型格オプション仕様²
- S6J3360, S6J3370, S6J3510 シリーズ: 200 MHz ~ 型格オプション仕様²
- S6J3400 シリーズ: 200 MHz ~ 320 MHz

CPU 動作クロック周波数の最大値は以下です。詳細はデータシートを参照してください。

- S6J311B, S6J311C, S6J311D, S6J311E: 144 MHz

² データシートを参照してください。最大動作周波数は型格および PLL/SSCG クロックの番号により異なります。

- S6J3118, S6J3119, S6J311A: 96 MHz
- S6J3128, S6J3129, S6J312A: 128 MHz
- S6J32xC: 240 MHz (x = 3, 4, 5, 6, 7, 8),
- S6J32xA: 160 MHz (x = A, B, C, D)
- S6J32xE: 240 MHz (x = E, F, G, K, L, M, N)
- S6J3300, S6J3350 シリーズ: 240 MHz
- S6J3360, S6J3370, S6J3510 シリーズ: 132 MHz
- S6J3400 シリーズ: 132 MHz

詳細は各製品のデータシートおよびハードウェアマニュアルを参照してください。

2.3 クロック分周/分配部

クロック分周/分配部は、クロックドメインごとにソースクロック選択および分周回路が存在します。Traveo Family マイコンには以下のクロックドメインが存在し、各ドメインでは 1 つのソースクロックが選択されます。分周されたソースクロックが内部クロックとして分配されます。

- クロックドメイン 0 (CD0)
- クロックドメイン 1/2/3/4/5 (CD1/CD2/CD3/CD4/CD5)
- TRC クロックドメイン (CD_TRC)
- HSSPI クロックドメイン (CD_HSSPI)
- MCUC クロックドメイン (CD_MCUC)
- CLKO クロックドメイン (CD_CLKO)

クロックドメインの構成や内部動作クロックの最大動作周波数の詳細は、各製品のデータシートおよびハードウェアマニュアルを参照してください。図 4 と図 5 に S6J3110 シリーズにおけるクロックドメイン 0 および MCUC クロックドメインのクロック分周/分配部のブロックダイアグラムを示します。ソースクロックが選択された後、ソースクロックの分周クロックが内蔵機能への動作クロックとして分配されます。

図 4. クロック分周/分配部ブロックダイアグラム (クロックドメイン 0)

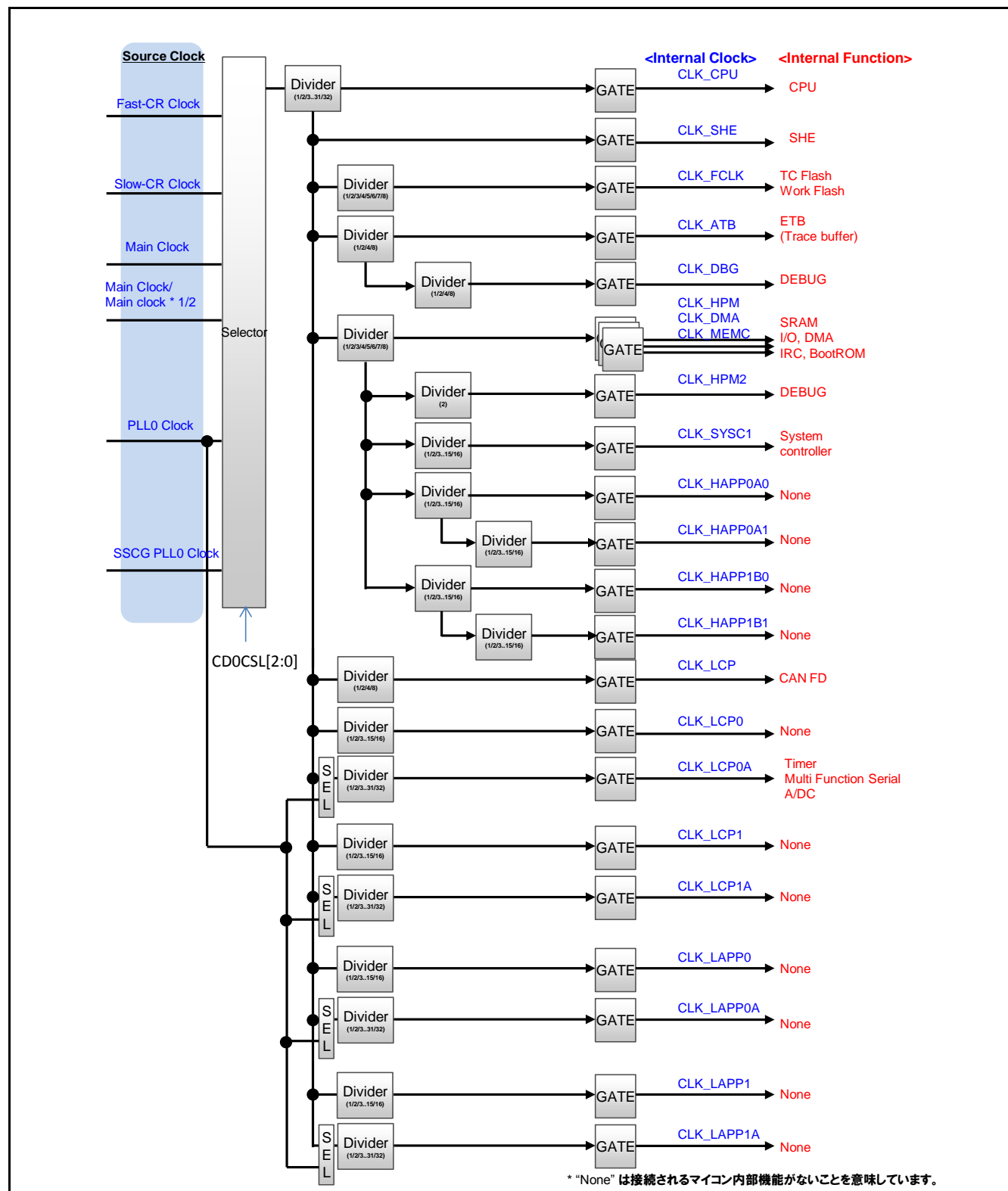
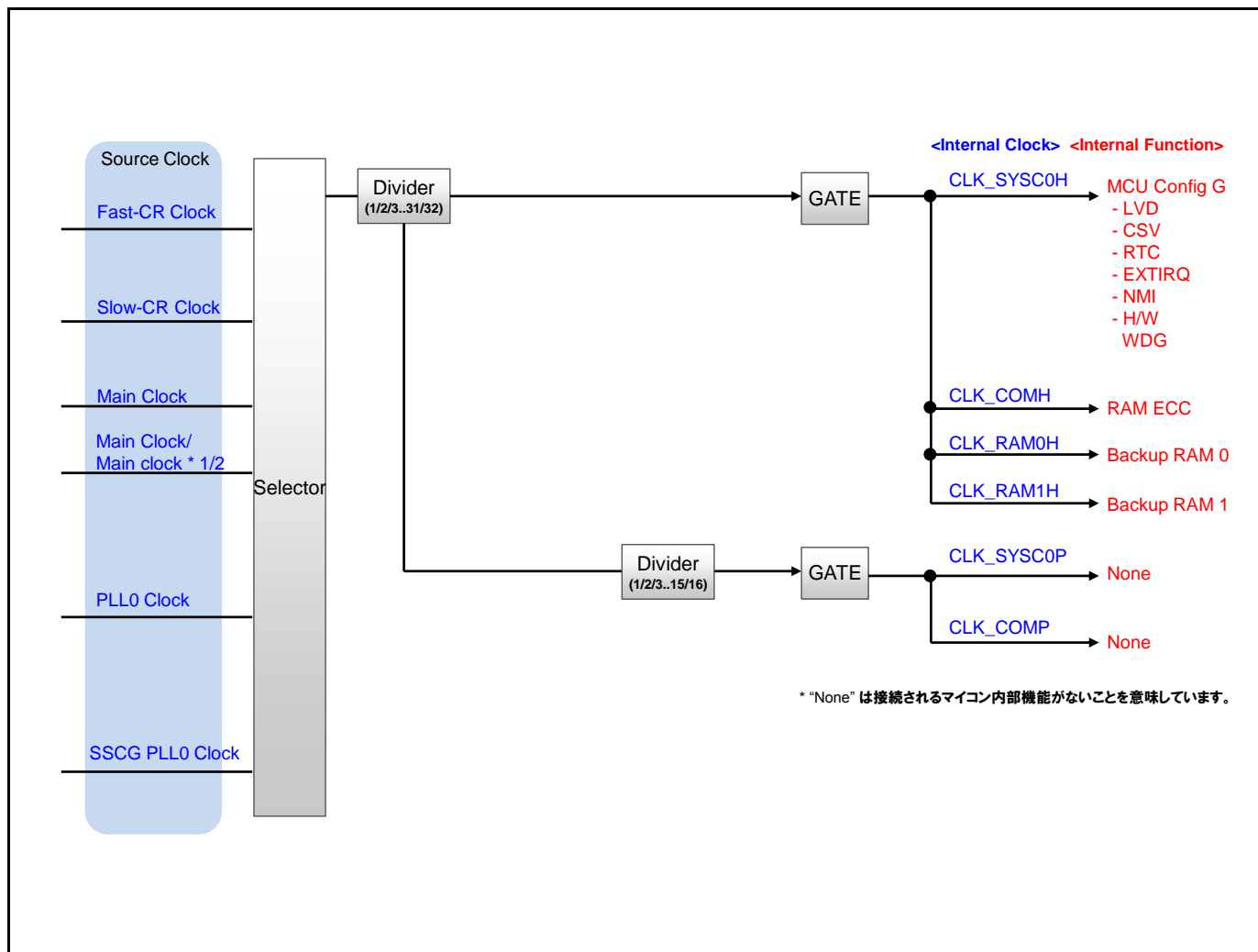


図 5. クロック分周/分配部ブロックダイアグラム (MCUC クロックドメイン)



2.4 周辺機能とソースクロックの組み合わせ確認方法

S6J3200/3300/3350/3360/3370/3400/3510 シリーズにおいて周辺機能とソースクロックの組み合わせを確認する場合、以下の手順に沿って確認ができます。


1. S6J3xxx シリーズ ハードウェアマニュアルに記載されているベースアドレスマッピング一覧表から、対象周辺機能のグループネームを参照してください。S6J3400 シリーズのベースタイマの場合、
 6 のとおりベースタイマは Common PERI #1 group に配置されています。

図 6. ベースアドレスマップ (S6J3400 シリーズ ハードウェアマニュアル)

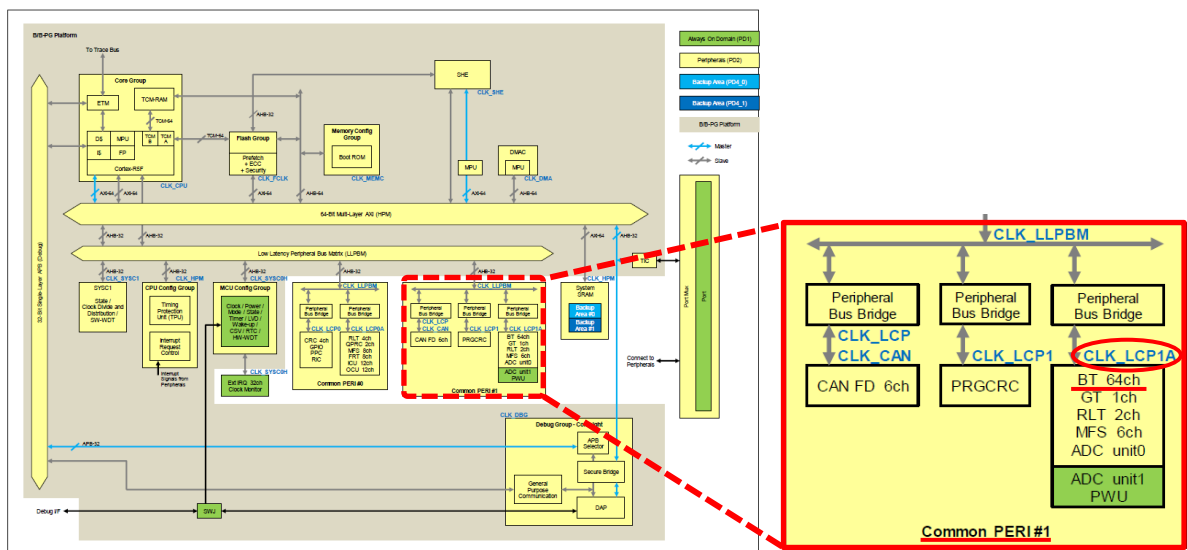


CHAPTER 7: Memory and Base Address Map

START Address	END Address	Group	Function	PPU_No
B484_D000	B484_D3FF	Common PERI #1	Base Timer ch.52	318
B484_D400	B484_D7FF	Common PERI #1	Base Timer ch.53	319
B484_D800	B484_DBFF	Common PERI #1	Base Timer ch.54	320
B484_DC00	B484_DFFF	Common PERI #1	Base Timer ch.55	321
B484_E000	B484_E3FF	Common PERI #1	Base Timer ch.56	322
B484_E400	B484_E7FF	Common PERI #1	Base Timer ch.57	323
B484_E800	B484_EBFF	Common PERI #1	Base Timer ch.58	324
B484_EC00	B484_EFFF	Common PERI #1	Base Timer ch.59	325

3. グループネームとクロックソースは、Traveo Family プラットフォームハードウェアマニュアルの“PLATFORM OVERVIEW Configuration”と“CLOCK SYSTEM”および S6J3xxx シリーズハードウェアマニュアルの“Block Diagram”の章に記載されています。S6J3400 シリーズの場合、図 7 のように、64 チャンルのベースタイマ(BT)は Common PERI #1 に配置され、そのクロックソースは“CLK_LCP1A”であることがわかります。

図 7. S6J3400 シリーズ ブロックダイヤグラム



4. 内部クロックの動作周波数については各製品のデータシートを参照してください。S6J3400 シリーズの場合、図 8 のとおりベースタイマのクロックソースである CLK_LCP1A の動作周波数は最大 40 MHz です。

図 8. 内部クロック周波数 (S6J3400 シリーズ)

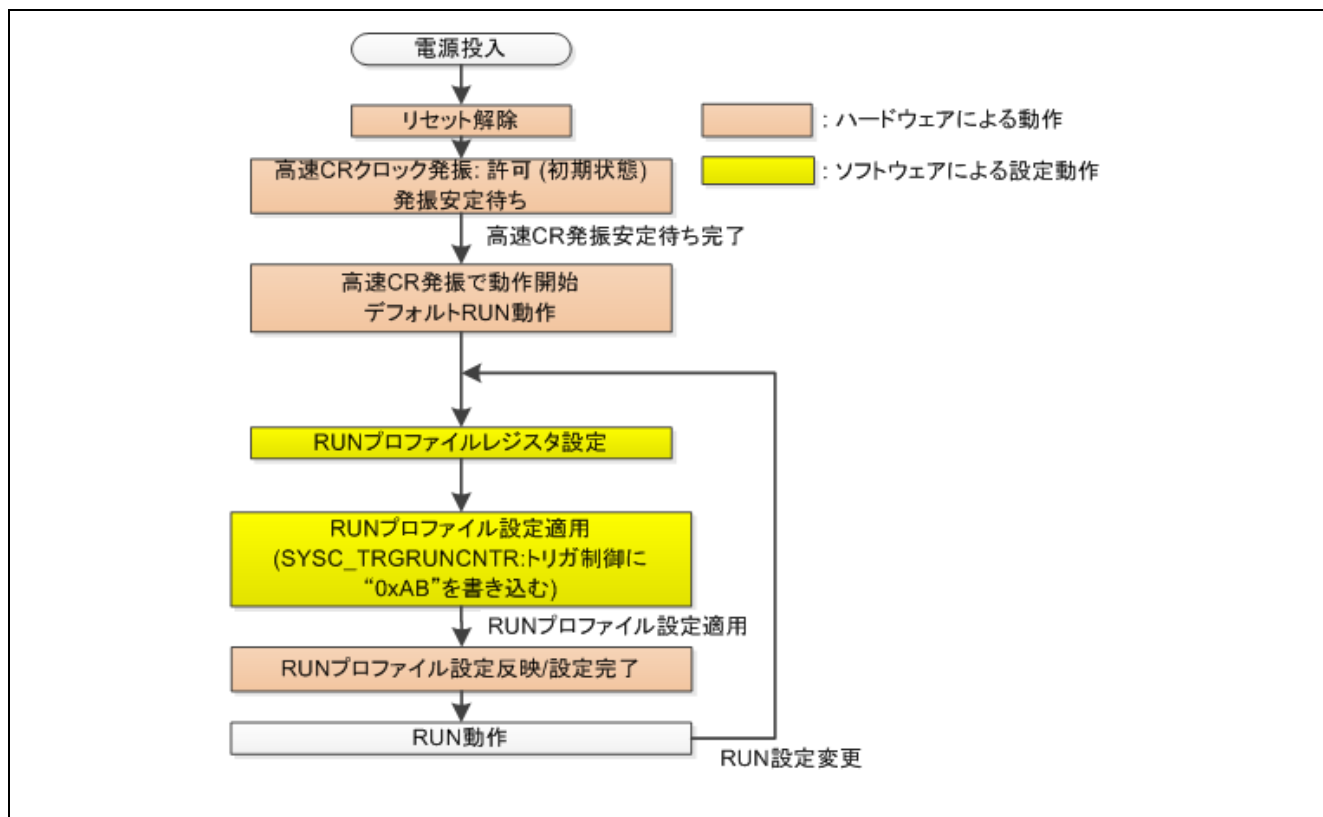
Parameter	Symbol	Pin Name	Conditions	Value				Unit	Remarks
				Min	Typ	Max *1	Max *2		
Internal clock frequency	fSSCG0out	-	-	200	-	320	320	MHz	
	fPLL0out	-	-	200	-	320	320	MHz	
	fCLK_CPU	-	-	-	-	132	80	MHz	
	fCLK_SHE	-	-	-	-	33	40	MHz	
	fCLK_FCLK	-	-	-	-	66	80	MHz	
	fCLK_ATB	-	-	-	-	66	40	MHz	
	fCLK_DBG	-	-	-	-	66	40	MHz	
	fCLK_HPM	-	-	-	-	33	40	MHz	
	fCLK_DMA	-	-	-	-	33	40	MHz	
	fCLK_MEMC	-	-	-	-	33	40	MHz	
	fCLK_SYSC1	-	-	-	-	33	40	MHz	
	fCLK_LLPBM	-	-	-	-	132	80	MHz	
	fCLK_LCP	-	-	-	-	66	80	MHz	
	fCLK_LCP0	-	-	-	-	33	40	MHz	
	fCLK_LCP0A	-	-	-	-	40	40	MHz	
	fCLK_LCP1	-	-	-	-	33	40	MHz	
	fCLK_LCP1A	-	-	-	-	40	40	MHz	

3 クロックシステムの設定方法

3.1 クロックシステム設定方法 (RUN 動作)

図 9 に RUN 動作のためのクロックシステム設定手順例を示します。

図 9. RUN 動作設定手順例



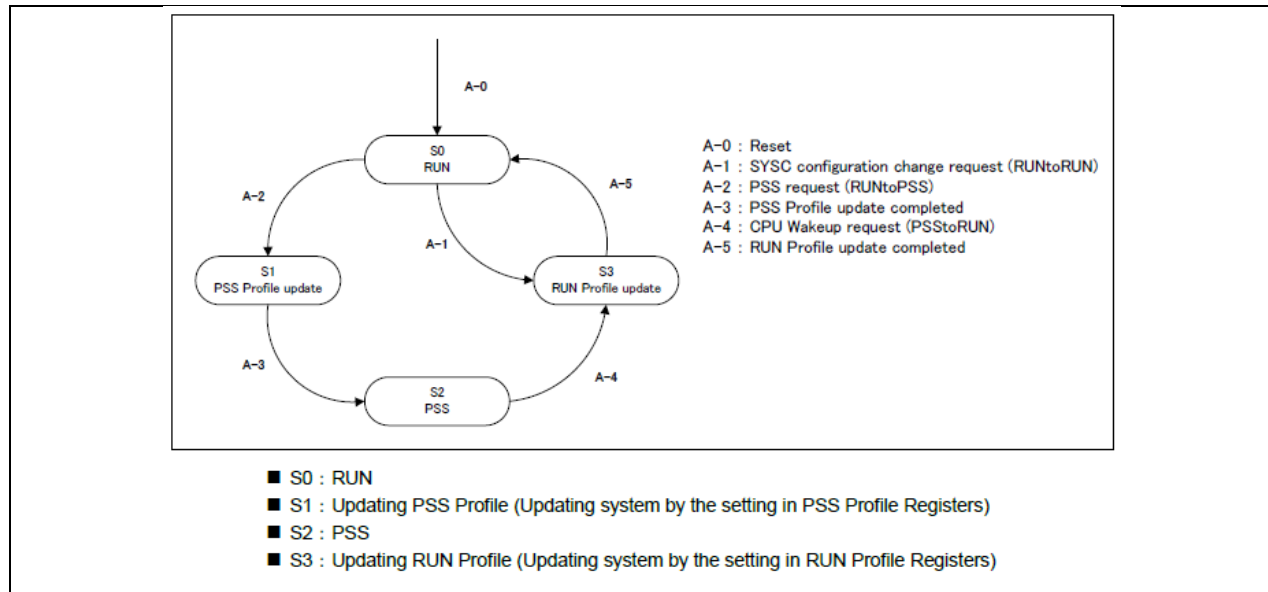
電源投入後、マイコンは高速 CR クロックで動作します。システム動作クロックを変更する場合は、ソフトウェアによる RUN プロファイルの設定および適用が必要です。

3.1.1 RUN プロファイル設定

大きく2つの低消費電力状態があり、設定により様々な低消費電力制御が可能です。図 10 に状態遷移図を示します。

- **RUN (normal operation):**
CPU がプログラム動作している状態です。初期化リセット後は本ステート(プログラム動作)で動作します。
RUN 状態では、RUN プロファイルの更新、PSS への移行ができます。
- **PSS (power-saving state):**
CPU がプログラムを停止している状態かつ低消費電力状態です。

図 10. 状態遷移図



以下のクロックに関する項目を RUN/PSS 状態において、それぞれ RUN プロファイルレジスタ/PSS プロファイルレジスタで設定します。クロック設定を変更する場合においても、プロファイルの設定および更新を行う必要があります。

- ソースクロック発振許可/禁止
- クロックドメイン制御 (ソース選択, 分周, 各ドメインクロックの発振許可/禁止)

3.1.2 RUN プロファイル設定適用

RUN/PSS プロファイル設定パラメータの設定を変更する場合、該当するレジスタへ値を書き込んだだけでは設定が反映されません。RUN プロファイルの設定を反映させるには以下の手順を行う必要があります。

1. 設定を変更するレジスタ全てに設定値を書き込む
2. システムステータスフラグ・割込みクリアレジスタ (SYSC0_SYSCICLR) にて RUN プロファイル更新完了 (メイン状態制御) フラグをクリア
3. RUN プロファイル更新トリガレジスタ (SYSC0_TRGRUNCNTR) に"0xAB"を書き込む
4. システムステータスレジスタ (SYSC0_SYSSTSR) の RUN プロファイル更新完了 (メイン状態制御) フラグビット (RUNDFO) をポーリングし、プロファイル更新が完了する (RUNDFO が"1"になる) まで待つ

制御回路が RUN プロファイルの内容を確認し、内容に問題がなければ以下の順で設定を反映します。

- システムステータスレジスタ (SYSC0_SYSSTSR) の RUNSTS0 に"1"をセット
- RUN プロファイルの内容を APPLIED プロファイルにコピー
- 以下の設定を順に反映
 - クロック発振許可/停止 (発振安定待ちも含む)
 - クロックスーパバイザの設定変更
 - 低電圧検出の設定変更
 - クロック動作設定 (ソースクロックの変更、分周、各クロックソース ON/OFF)
 - クロック停止設定 (ソースクロックの停止)
- RUN プロファイルの更新が完了後、システムステータスレジスタ (SYSC0_SYSSTSR) の RUNSTS0 を"0"クリア
- システムステータスレジスタ (SYSC0_SYSSTSR) の RUNDFO に"1"をセット

RUN プロファイルの設定内容に問題がある場合はプロファイルエラーとなり、システムエラー割込み要因レジスタ 1 (SYSC0_SYSEERRIR1) の RUNERRIF0 に"1"をセットします。このとき新しいプロファイルの内容は破棄され、現在使用しているプロファイルの内容で回路は動作します。

注意事項:

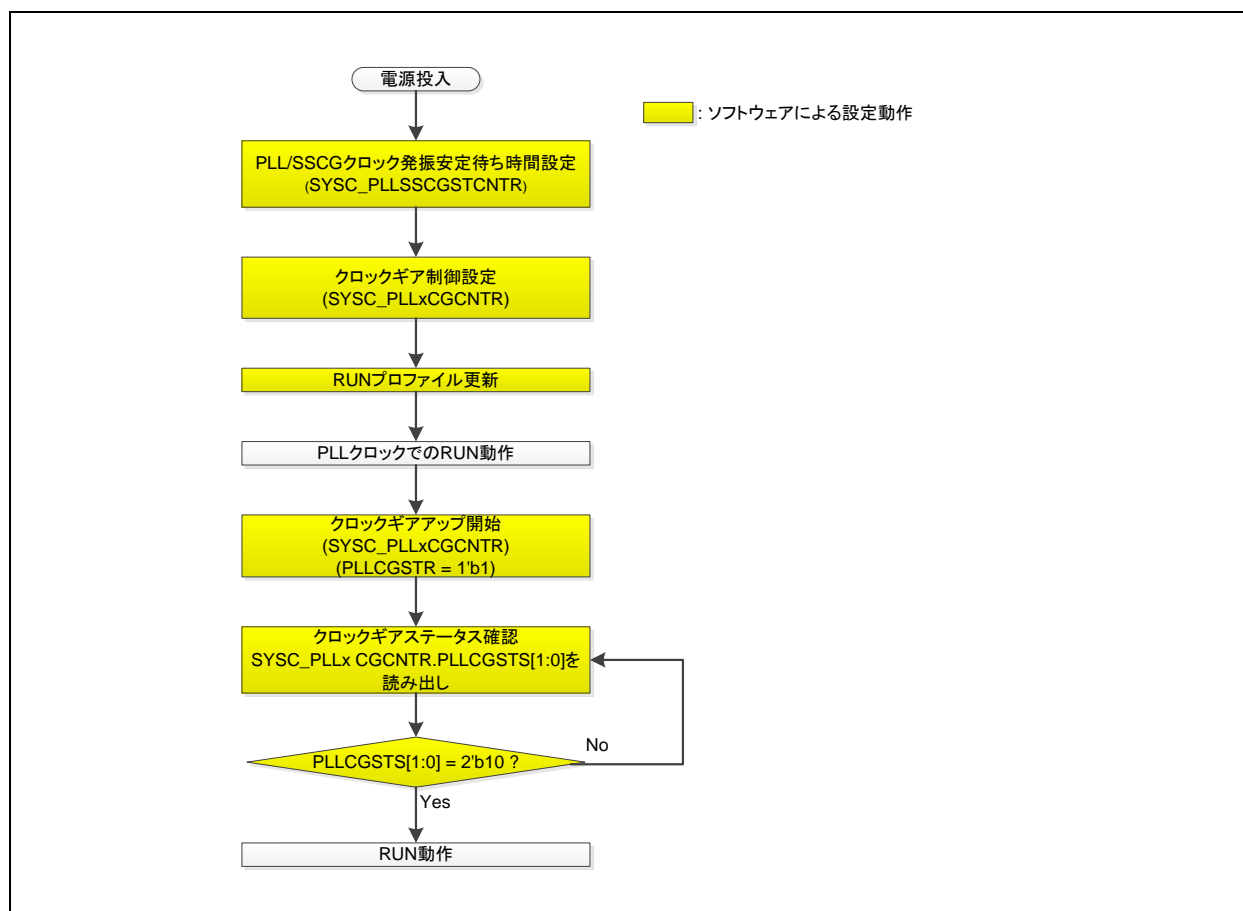
1. プロファイル更新中のプロファイル更新は禁止です。プロファイル更新中に再度プロファイルを更新した場合システムエラー割込みが発生し、再度更新しようとしたプロファイルは無効となります。
2. RUN プロファイルを更新する前に、プロファイルステータスレジスタ(SYSC0_SYSPROSTSR:RUNPSTS)を確認し、プロファイルエラーがないことを確認してください。プロファイルエラーの状態で RUN プロファイル更新を行った場合 NMI 割込みが発生し、RUN プロファイルの設定は破棄されます。
3. RUN プロファイル更新中は RUN プロファイルレジスタ群にライトアクセスすることは禁止です。RUN プロファイル更新中に RUN プロファイルレジスタ群にライトアクセスした場合、バスエラーとなり書き込んだデータは無効となります。

プロファイル設定や動作の詳細については、ハードウェアマニュアルの低消費電力の章を参照してください。

3.2 クロックシステム設定手順 (PLL 動作)

図 11 に PLL クロックの設定手順を示します。RUN プロファイル設定後、PLL クロックへの切り替えはクロックギアを使用することで、徐々に変化させることができます。

図 11. PLL クロック設定手順例



3.2.1 ソースクロック許可設定

ハードウェアリセット直後は外部/内蔵発振回路からのソースクロック（高速 CR クロック、低速 CR クロック、メインクロック/メイン 2 分周クロック）のみが発振許可となり、PLL クロックは発振禁止となります。システムクロックのソースクロックを PLL クロックに変更するためには PLL クロックの発振を許可にする必要があります。PLL クロックの発振を許可するためにはメインクロックが発振許可になっている必要があります。メインクロックの発振が禁止となっていた場合、プロファイルエラーとなります。

表 2 に示すレジスタのビットに“1”を書き込むことにより、RUN 動作時の該当クロックの発振を許可することができます。ただし、システムで使用されているクロックの設定は変更できません。また、RUN 動作時は、高速 CR クロックおよび、低速 CR クロックの発振を禁止にすることはできません。

表 2. ソースクロック発振許可設定例

レジスタ略称	ビット名	設定内容	設定値 (2 進数)
SYSC0_RUNCKSRER	PLLxEN	PLL クロック発振許可	1
	MOSCEN	メインクロック発振許可	1
	SCROSCEN	低速 CR クロック発振許可	1
	CROSCEN	高速 CR クロック発振許可	1

3.2.2 ソースクロックタイマ設定

クロックの発振安定待ち時間までクロック出力をゲーティングするためにソースクロックタイマを使用します。ソースタイマを使用するためには表 3 に示すレジスタを設定する必要があります。

表 3. ソースクロックタイマ設定例

レジスタ略称	ビット名	設定内容
SYSC_MOCTCPR	PSCL	メインクロックタイマの入力クロック分周比選択
	CMPR	メインクロックタイマのコンペア値
SYSC_MOCTTRGR	CGCPT	タイマ設定変更/タイマカウント開始
SYSC_PLLSSCGSTCNTR	PLLSTABS	PLL 発振安定待ち時間を選択

3.2.3 クロックギア設定

クロックギアはクロックギア回路に入力されたクロックを段階的に出力していきます。これにより周波数を徐々に変化させることができます。

クロックギアを使用せずにメインクロックから PLL クロックへの変更を行った場合、周波数が急激に変動することにより電源電流が大きく変動します。電源電流のオーバシュート/アンダシュートを避けるため、クロック変更時は必ずクロックギアを使用してください。

クロックギアを使用するためには表 4 に示すレジスタを設定する必要があります。

表 4. クロックギア設定例

レジスタ略称	ビット名	設定内容
SYSC_PLL0CGCNTR	PLLCGLP ³	クロックギア動作の 1 ステップにおけるループ回数
	PLLCGSTP ³	ギアアップ/ギアダウン時のステップ幅
	PLLCGSSN ³	クロックギア動作開始ステップ
	PLLCGSTR	クロックギア動作開始
	PLLCGEN ³	クロックギア動作許可

3.2.4 クロックギア動作開始方法

表 4 (PLLCGSTR を除く) の設定を行った後、以下の手順を行うことでクロックギアが動作を開始します。クロックギア動作開始後はクロックギアが完了するまで待ってください。

1. PLL クロックをドメインクロックとして選択
2. クロックギア制御レジスタ (SYSC_PLL0CGCNTR) の PLLCGSTR に"01"をセット
3. クロックギア制御レジスタ (SYSC_PLL0CGCNTR) の PLLCGSTS をポーリングし、クロックギアが停止する (PLLCGSTS が"10"になる) まで待つ

注意事項:

1. ステップ幅が小さく、ループ回数が大きいほど周波数は緩やかに変化していきます。
2. クロックギアを動作許可に設定すると、クロックギア入力クロックの発振安定待ち時間経過後から PLLCGSSN の設定に応じたクロックを出力します。ただし、クロックギア動作開始設定が行われるまでギアアップ/ギアダウンは行いません。

3.2.5 PLL クロック通倍/分周

PLL クロックの通倍/分周設定は RUN PLLx 制御レジスタ (SYSC0_RUNPLLxCNTR)で設定します。

PLL クロック設定例については 表 5 を参照してください。

表 5. クロック通倍/分周設定例

レジスタ略称	ビット名	設定内容
SYSC0_RUNPLLxCNTR	PLLDIVN	PLL クロック N 通倍設定ビット
	PLLDIVM	PLL クロック M 分周設定ビット
	PLLDIVL	PLL 入力クロック分周設定ビット

3.2.6 クロックドメイン 0 ソースクロック選択

ハードウェアリセット直後、全てのクロックドメインはソースクロックに高速 CR クロックが選択されている状態になります。システムクロックのソースクロックを PLL クロックに変更する場合はクロックドメイン 0 のソースクロックに PLL クロックを設定する必要があります。クロックドメイン 0 のソースクロックを PLL クロックに設定することでシステムクロックのソースクロックが PLL クロックになります。

RUN 動作時のクロックドメイン 0 のソースクロックは表 6 に示すレジスタを設定することによって選択することが可能です。RUN 動作時のクロックドメイン 0 のソースクロックを PLL クロックに設定する場合の設定値を表 6 に記します。

³ PLL クロック発振許可設定後の設定変更は禁止です。

表 6. クロックドメイン 0 ソースクロック設定例

レジスタ略称	ビット名	設定内容	設定値 (10 進数)
SYSC1_RUNCKSELR0	CD0CSL	クロックドメイン 0 ソースクロック選択	4

3.3 クロック設定手順例

図 12 にクロック設定手順例を示します。

図 12. PLL クロック設定手順例 (start.c)

```
static void ConfigureClocks(void)
{
    // At first, disable PSS profile update.
    // This setting is for wakeup from shutdown mode.
    // note: SYSC1 was cleared by hardware after PSS profile is updated, but SYSC0 was not cleared.
    SYSC0_PROTKEYR = SYSC_KEY_UNLOCK;
    SYSC0_PSSSEN0 = 0;

    // Stabilization time setting with Source Clock Timer

    // Set main oscillator and main PLL stabilization time
    //-----
    // Set new main oscillation stabilization time and trigger data update
    SYSC0_PROTKEYR = SYSC_KEY_UNLOCK; // unlock SYSC0
    SYSC_6.unMOCTCPR.stcField = (stc_sysc_6_moctcpr_field_t){ .u4PSCL = SYSC_MAINSCT_PRESCALER, // pre-scaler
                                                                .u16CMPR = SYSC_MAINSCT_CMPR }; // compare value

    // trigger configuration capture
    SYSC0_PROTKEYR = SYSC_KEY_UNLOCK; // unlock SYSC0
    SYSC_6.unMOCTTRGR.stcField = (stc_sysc_6_mocttrgr_field_t){ .u1CGCPT = 1 };

    // Set PLL / SSCG stabilization time
    SYSC0_PROTKEYR = SYSC_KEY_UNLOCK; // unlock SYSC0
    SYSC_7.unPLLSSCGSTCNTR.stcField = (stc_sysc_7_pllsscgstcntr_field_t){ .u4PLLSTABS = SYSC_PLLST_PLLSTABS,
                                                                            .u4SSCGSTABS = SYSC_PLLST_PLLSTABS, };

    // RUN Profile configuration (Power Domain setting)

    // Configure run profile
    //-----
    // Enable power domains
    SYSC0_PROTKEYR = SYSC_KEY_UNLOCK; // unlock SYSC0
    SYSC0_1.unRUNPDCFGR.stcField = (stc_sysc0_1_runpdcfgr_field_t){ .u1PD6_1EN = 1, // switch on PD6_1 ()
                                                                    .u1PD6_0EN = 1, // switch on PD6_0 ()
                                                                    .u1PD5_3EN = 1, // switch on PD5_3 ()
                                                                    .u1PD5_2EN = 1, // switch on PD5_2 ()
                                                                    .u1PD5_1EN = 1, // switch on PD5_1 ()
                                                                    .u1PD5_0EN = 1, // switch on PD5_0 ()
                                                                    .u1PD4_1EN = 1, // switch on PD4_1 (Backup RAM1)
                                                                    .u1PD4_0EN = 1, // switch on PD4_0 (Backup RAM0)
                                                                    .u1PD3EN = 1, // always on (Core)
                                                                    .u1PD2EN = 1 }; // always on (Peripheral)
}
```

Stabilization time setting with Source Clock Timer

Stabilization time setting for Main clock and PLL/SSCG clock.

- Input protect key of SYSC
- Set the Prescaler and compare value

- Input protect key of SYSC
- Change the main clock timer setting and start the timer counting

- Input protect key of SYSC
- Set the PLL and SSCG stabilization time

RUN Profile configuration (Power Domain setting)

- Input protect key of SYSC
- Power Domain enable setting.

RUN Profile configuration (Enable Oscillator)

```
// Enable oscillators
SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC0_1.unRUNCKSRER.stcField = (stc_sysc0_1_runcksrer_field_t){
    .u1SOSCEN = 0, // disable Sub Oscillation
    .u1PLL0EN = 1, // enable PLL0
    .u1SSCG0EN = 1, // enable SSCG0
    .u1MOSCEN = 1, // enable Main Oscillation
    .u1CROSCEN = 1, // RC Osc. (always on in RUN state)
    .u1SCROSCEN = 1 }; // Slow RC Osc. (always on in RUN state)
```

- Input protect key of SYSC
- Enable Oscillator setting
(Except for sub oscillation)

RUN Profile configuration (PLL/SSCG setting)

```
//-----
// Write Main / SSCG PLL settings
SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC0_1.unRUNPLL0CNTR.stcField = (stc_sysc0_1_runpll0cntr_field_t){
    .u8PLL0DIVN = (SYSC_MAIN_PLL_DIVN), // set PLL input multiplication value
    .u4PLL0DIVM = (SYSC_MAIN_PLL_DIVM), // set PLL output divider
    .u2PLL0DIVL = (SYSC_PLL_DIVL)); // set PLL input divider

SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC0_1.unRUNSSCG0CNTR0.stcField = (stc_sysc0_1_runsscg0cntr0_field_t){
    .u8SSCG0DIVN = (SYSC_MAIN_SSCG_DIVN), // set SSCG PLL input multiplication value
    .u4SSCG0DIVM = (SYSC_MAIN_SSCG_DIVM), // set SSCG PLL output divider
    .u2SSCG0DIVL = (SYSC_SSCG_DIVL)); // set SSCG PLL input divider
```

RUN Profile configuration (Clock gear)

```
// Clock gear
SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC_7.unPLL0CGCNTR.stcField = (stc_sysc_7_pll0cgcntr_field_t){
    .u8PLL0GLP = 4, // Loops per step
    .u2PLL0GSP = 1, // 2 steps
    .u6PLL0GSSN = 8, // Start step = 8
    .u1PLL0GSTR = 0, // Start gear operation
    .u1PLL0GEN = 1}; // Enable

SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC_7.unSSCG0CGCNTR.stcField = (stc_sysc_7_sscg0cgcntr_field_t){
    .u8SSCG0GLP = 4, // Loops per step
    .u2SSCG0GSP = 1, // 2 steps
    .u6SSCG0GSSN = 8, // Start step = 8
    .u1SSCG0GSTR = 0, // Start gear operation
    .u1SSCG0GEN = 1}; // Enable
```

- Input protect key of SYSC
- PLL clock gear setting.

- Input protect key of SYSC
- SSCG clock gear setting.

RUN Profile configuration (Clock source selection)

```
//-----
// Select clock sources
SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC0_1.unRUNCKSELR.stcField = (stc_sysc0_1_runckselr_field_t){
    .u3CDMCUCCSL = 5}; // Clock domain MCUC clock = SSCG0

SYSC1_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC1
SYSC1.unRUNCKSELR0.stcField = (stc_sysc1_runckselr0_field_t){
    .u4HSSPICSL = 0, // Hsspi clock domain = Fast CR
    .u1LAPP1ACSL = 0, // LAPP1A clock = CD0 (1=PLL0)
    .u1LAPP0ACSL = 0, // LAPP0A clock = CD0 (1=PLL0)
    .u1LCP1ACSL = 0, // LCP1A clock = CD0 (1=PLL0)
    .u1LCP0ACSL = 0, // LCP0A clock = CD0 (1=PLL0)
    .u3CD0CSL = 5}; // Clock Domain 0 = SSCG0
```

- Input protect key of SYSC
- Clock domain MCUC clock
→ Select SSCG
- Input protect key of SYSC
- Clock domain 0 clock
→ Select SSCG
- LAPP1A/LAPP0A/LCP1A/LCP0A
→ Select CD0


```
SYSC1_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC1
SYSC1.unRUNCKSELR2.stcField = (stc_sysc1_runckselr2_field_t){
    .u3TRCCSL = 5; // TRC clock = SSCG0
```

RUN Profile configuration (Enable clocks)

```
//-----
// Enable clocks
SYSC1_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC1
SYSC1.unRUNCKER1.stcField = (stc_sysc1_runcker1_field_t){
    .u1ENCLKCD3B1 = 0, // Disable CD3B1
    .u1ENCLKCD3B0 = 0, // Disable CD3B0
    .u1ENCLKCD3A1 = 0, // Disable CD3A1
    .u1ENCLKCD3A0 = 0, // Disable CD3A0
    .u1ENCLKCD3    = 0, // Disable CD3
    .u1ENCLKCD2B1 = 0, // Disable CD2B1
    .u1ENCLKCD2B0 = 0, // Disable CD2B0
    .u1ENCLKCD2A1 = 0, // Disable CD2A1
    .u1ENCLKCD2A0 = 0, // Disable CD2A0
    .u1ENCLKCD2    = 0, // Disable CD2
    .u1ENCLKCD1B1 = 0, // Disable CD1B1
    .u1ENCLKCD1B0 = 0, // Disable CD1B0
    .u1ENCLKCD1A1 = 0, // Disable CD1A1
    .u1ENCLKCD1A0 = 0, // Disable CD1A0
    .u1ENCLKCD1    = 0, // Disable CD1
    .u1ENCLKHSSPI = 0; // Disable HSSPI

    SYSC1_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC1
    SYSC1.unRUNCKER2.stcField = (stc_sysc1_runcker2_field_t){
        .u1ENCLKCD5B1 = 0, // Disable CD5B1
        .u1ENCLKCD5B0 = 0, // Disable CD5B0
        .u1ENCLKCD5A1 = 0, // Disable CD5A1
        .u1ENCLKCD5A0 = 0, // Disable CD5A0
        .u1ENCLKCD5    = 0, // Disable CD5
        .u1ENCLKCD4B1 = 0, // Disable CD4B1
        .u1ENCLKCD4B0 = 0, // Disable CD4B0
        .u1ENCLKCD4A1 = 0, // Disable CD4A1
        .u1ENCLKCD4A0 = 0, // Disable CD4A0
        .u1ENCLKCD4    = 0; // Disable CD4
```

- Input protect key of SYSC
- Disable clocks

- Input protect key of SYSC
- Disable clocks

RUN Profile configuration (Clock divider setting)

```
//-----
// Set clock dividers (valid setting for all main-PLL frequencies)
SYSC0_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC0
SYSC0_1.unRUNCKDIVR.stcField = (stc_sysc0_1_runckdivr_field_t){
    .u4MCUCPDIV = 0, // MCUconfig APB clock divider = 1
    .u5MCUCHDIV = 3; // MCUconfig AHB clock divider = 4

    // Set clock dividers (valid setting for all main-PLL frequencies)
    SYSC1_PROTKEYR      = SYSC_KEY_UNLOCK; // unlock SYSC1
    SYSC1.unRUNCKDIVR0.stcField = (stc_sysc1_runckdivr0_field_t){
        .u3HPMDIV = 3, // HPM clock divider = 4
        .u5TRCDIV = 3, // TRC Clock divider = 4
        .u2DBGDIV = 0, // DBG Clock divider = 1
        .u2ATBDIV = 1, // ATB Clock divider = 2
        .u5SYSDIV = 0; // System Clock divider = 1
```

- Input protect key of SYSC
- Divider setting for Clock domain
MCUC block

- Input protect key of SYSC
- Divider setting for following clock
-HPM
-TRC
-DBG
-ATB
-System clock

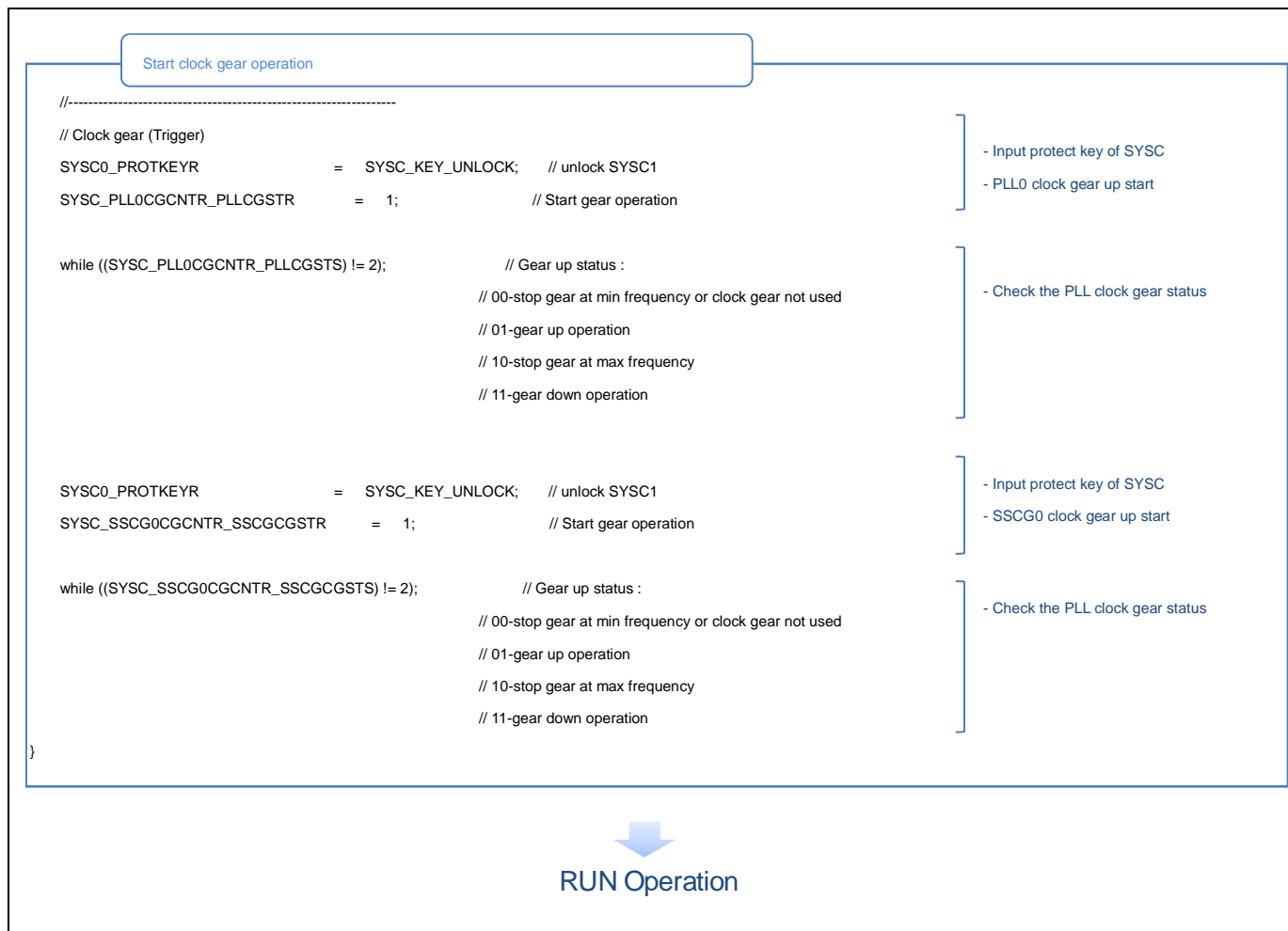
SYSC1_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC1		- Input protect key of SYSC
SYSC1.unRUNCKDIVR1.stcField	=	(stc_sysc1_runckdivr1_field_t) {	.u4HAPP1B1DIV = 0, // HAPP1B1 clock divider = 1		- Divider setting for following clock
			.u4HAPP1B0DIV = 0, // HAPP1B0 Clock divider = 1		-HAPP1B1DIV / HAPP1B0DIV
			.u4HAPP0A1DIV = 0, // HAPP0A1 Clock divider = 1		-HAPP0A1DIV / HAPP0A0DIV
			.u4HAPP0A0DIV = 0, // HAPP0A0 Clock divider = 1		-SYSC1
			.u4SYSC1DIV = 0, // SYSC1 Clock divider = 1		-External Bus
			.u3EXTBUSDIV = 7; // EXTBUS Clock divider = 128		
SYSC1_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC1		- Input protect key of SYSC
SYSC1.unRUNCKDIVR2.stcField	=	(stc_sysc1_runckdivr2_field_t) {	.u4LAPP1DIV = 3, // LAPP1 clock divider = 4		- Divider setting for following clock
			.u4LAPP0DIV = 3, // LAPP0 Clock divider = 4		-LAPP1 / LAPP0
			.u4LCP1DIV = 3, // LCP1 Clock divider = 4		-LCP1 / LCP0
			.u4LCP0DIV = 3, // LCP0 Clock divider = 4		-LCP
			.u2LCPDIV = 1; // LCP Clock divider = 2		
SYSC1_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC1		- Input protect key of SYSC
SYSC1.unRUNCKDIVR3.stcField	=	(stc_sysc1_runckdivr3_field_t) {	.u5LAPP1ADIV = 3, // LAPP1A clock divider = 4		- Divider setting for following clock
			.u5LAPP0ADIV = 3, // LAPP0A Clock divider = 4		-LAPP1A / LAPP0A
			.u5LCP1ADIV = 3, // LCP1A Clock divider = 4		-LCP1A / LCP0A
			.u5LCP0ADIV = 3; // LCP0A Clock divider = 4		

RUN Profile flag clear and start updating

//-----					
// Clear RUN Profile Done flag (SYSC_SYSSTSR_RUNDN)					
SYSC0_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC0		- Input protect key of SYSC
SYSC0_SYSCCLR_RUNDFCLR0	=	1;			- Clear the RUN profile update completion flag
// RUN Profile update enable					
SYSC1_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC1		- Input protect key of SYSC
SYSC1_RUNENR_RUNEN1	=	SYSC_TRIGGER_APPLY_RUN_PROFILE;			- Setting RUN Profile enable
// Write the trigger value to apply the RUN profile					
SYSC0_PROTKEYR	=	SYSC_KEY_UNLOCK;	// unlock SYSC0		- Input protect key of SYSC
SYSC0_TRGRUNCNTR	=	SYSC_TRIGGER_APPLY_RUN_PROFILE;	// trigger RUN-->RUN transition		- Start updating the RUN Profile enables

RUN Profile updating check

// Wait until the RUN profile is applied					- Wait until the RUN Profile is applied
while (SYSC0_SYSSTSR_RUNDF0 == 0);					



4 関連ドキュメント

4.1 Datasheets

- [S6J311E/D/C/B Series Datasheet \(Doc. No.002-05681\)](#)
- [S6J311A/9/8 Series Datasheet \(Doc. No.002-04632\)](#)
- [S6J3110 Series Hardware Manual \(Doc.No.002-10667\)](#)
- [S6J3120 Series Datasheet \(Doc.No.002-04863\)](#)
- [S6J3200 Series Datasheet \(Doc.No.002-05682\)](#)
- [S6J32E/F/G Series Datasheet \(Doc.No.002-10689\)](#)
- [S6J3310/20/30/40 Series Datasheet \(Doc.No.002-10635\)](#)
- [S6J3350 Series Datasheet \(Doc.No.002-10634\)](#)
- [S6J3360/70 Series Datasheet \(Doc.No.002-03359\)](#)
- [S6J3400 Series Datasheet \(Doc.No.001-97829\)](#)
- [S6J3510 Series Datasheet \(Doc.No.002-18647\)](#)
- [MB9D560 Series Datasheet \(Doc.No.002-05679\)](#)

4.2 Hardware Manuals

- [S6J3120 Series Hardware Manual \(Doc.No.002-04855\)](#)
- [S6J3200 Series Hardware Manual \(Doc.No.002-04852\)](#)
- [S6J32E/F/G Series Hardware Manual \(Doc.No.002-12500\)](#)
- [Traveo Family Hardware Manual Platform Part for S6J3200 Series \(Doc.No.002-04854\)](#)
- [S6J3310/20/30/40/50 Series Hardware Manual \(Doc.No.002-10185\)](#)
- [Traveo Family Hardware Manual Platform Part for S6J3310/3320/3330/3340/3350 Series \(Doc.No.002-07884\)](#)
- [S6J3360/70 Series Hardware Manual \(Doc.No.002-18302\)](#)
- [Traveo Family Hardware Manual Platform Part for S6J3360/3370 Series \(Doc.No.002-07884\)](#)
- [S6J3400 Series Hardware Manual \(Doc.No.002-09919\)](#)
- [Traveo Family Hardware Manual Platform Part for S6J3400 Series \(Doc.No.002-07884\)](#)
- [S6J3510 Series Hardware Manual \(Doc.No.002-18642\)](#)
- [Traveo Family Hardware Manual Platform Part for S6J3510 Series \(Doc.No.002-07884\)](#)
- [MB9D560 Series Hardware Manual \(Doc.No.002-07882\)](#)

改訂履歴

ドキュメント名: AN204455 - Traveo™ Family クロックシステムの設定方法

ドキュメント番号: 002-04456

Revision	ECN	変更者	発行日	変更内容
**	-	KHAS	07/31/2015	Initial release
*A	5474755	KOTH	10/14/2016	英語版 002-04455 Rev.*C の日本語版です。
*B	5954927	KOTH	11/02/2017	英語版 002-04455 Rev.*E の日本語版です。

セールス、ソリューションおよび法律情報

ワールドワイドな販売と設計サポート

サイプレスは、事業所、ソリューション センター、メーカー代理店、および販売代理店の世界的なネットワークを保持しています。お客様の最寄りのオフィスについては、[サイプレスのロケーション ページ](#)をご覧ください。

製品

ARM® Cortex® Microcontrollers	cypress.com/arm
車載用	cypress.com/automotive
クロック&バッファ	cypress.com/clocks
インターフェース	cypress.com/interface
IoT (モノのインターネット)	cypress.com/iot
メモリ	cypress.com/memory
マイクロコントローラ	cypress.com/mcu
PSoC	cypress.com/psoc
電源用 IC	cypress.com/pmic
タッチ センシング	cypress.com/touch
USB コントローラー	cypress.com/usb
ワイヤレス	cypress.com/wireless

PSoC® ソリューション

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6](#)

サイプレス開発者コミュニティ

[フォーラム](#) | [WICED IOT Forums](#) | [Projects](#) | [ビデオ](#) | [ブログ](#) | [トレーニング](#) | [Components](#)

テクニカルサポート

cypress.com/support

Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.


CYPRESS
 EMBEDDED IN TOMORROW™

Cypress Semiconductor
 198 Champion Court
 San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2015-2017. 本書面は、Cypress Semiconductor Corporation 及び Spansion LLC を含むその子会社 (以下「Cypress」という。)に帰属する財産である。本書面 (本書面に含まれ又は言及されているあらゆるソフトウェア若しくはファームウェア (以下「本ソフトウェア」という。)) を含む) は、アメリカ合衆国及び世界のその他の国における知的財産法令及び条約に基づき Cypress が所有する。Cypress はこれらの法令及び条約に基づく全ての権利を留保し、本段落で特に記載されているものを除き、その特許権、著作権、商標権又はその他の知的財産権のライセンスを一切許諾しない。本ソフトウェアにライセンス契約書が伴っておらず、かつ Cypress との間で別途本ソフトウェアの使用方法を定める書面による合意がない場合、Cypress は、(1) 本ソフトウェアの著作権に基づき、(a) ソースコード形式で提供されている本ソフトウェアについて、Cypress ハードウェア製品と共に用いるためにのみ、かつ組織内部でのみ、本ソフトウェアの修正及び複製を行うこと、並びに (b) Cypress のハードウェア製品ユニットに用いるためにのみ、(直接又は再販売者及び販売代理店を介して間接のいずれかで) 本ソフトウェアをバイナリーコード形式で外部エンドユーザーに配布すること、並びに (2) 本ソフトウェア (Cypress により提供され、修正がなされていないもの) が抵触する Cypress の特許権のクレームに基づき、Cypress ハードウェア製品と共に用いるためにのみ、本ソフトウェアの作成、利用、配布及び輸入を行うことについての非独占的で譲渡不能な一身専属的ライセンス (サブライセンスの権利を除く) を付与する。本ソフトウェアのその他の使用、複製、修正、変換又はコンパイルを禁止する。

適用される法律により許される範囲内で、Cypress は、本書面又はいかなる本ソフトウェア若しくはこれに伴うハードウェアに関しても、明示又は黙示をとわず、いかなる保証 (商品性及び特定の目的への適合性の黙示の保証を含むがこれらに限られない) も行わない。適用される法律により許される範囲内で、Cypress は、別途通知することなく、本書面を変更する権利を留保する。Cypress は、本書面に記載のある、いかなる製品若しくは回路の適用又は使用から生じる一切の責任を負わない。本書面で提供されたあらゆる情報 (あらゆるサンプルデザイン情報又はプログラムコードを含む) は、参照目的のためのみに提供されたものである。この情報で構成するあらゆるアプリケーション及びその結果としてのあらゆる製品の機能性及び安全性を適切に設計、プログラム、かつテストすることは、本書面のユーザーの責任において行われるものとする。Cypress 製品は、兵器、兵器システム、原子力施設、生命維持装置若しくは生命維持システム、蘇生用の設備及び外科的移植を含むその他の医療機器若しくは医療システム、汚染管理若しくは有害物質管理の運用のために設計され若しくは意図されたシステムの重要な構成部分としての使用、又は装置若しくはシステムの不具合が人身傷害、死亡若しくは物的損害を生じさせるようなその他の使用 (以下「本目的外使用」という。) のためには設計、意図又は承認されていない。重要な構成部分とは、その不具合が装置若しくはシステムの不具合を生じさせるか又はその安全性若しくは実効性に影響すると合理的に予想できるような装置若しくはシステムのあらゆる構成部分という。Cypress 製品のあらゆる本目的外使用から生じ、若しくは本目的外使用に関連するいかなる請求、損害又はその他の責任についても、Cypress はその全部又は一部をとわず一切の責任を負わず、かつ Cypress はそれら一切から本書により免除される。Cypress は Cypress 製品の目的外使用から生じ又は本目的外使用に関連するあらゆる請求、費用、損害及びその他の責任 (人身傷害又は死亡に基づく請求を含む) から免責補償される。

Cypress, Cypress のロゴ, Spansion, Spansion のロゴ及びこれらの組み合わせ, WICED, PSoC, Capsense, EZ-USB, F-RAM, 及び Traveo は、米国及びその他の国における Cypress の商標又は登録商標である。Cypress のより完全な商標のリストは、cypress.com を参照すること。その他の名称及びブランドは、それぞれの権利者の財産として権利主張がなされている可能性がある。