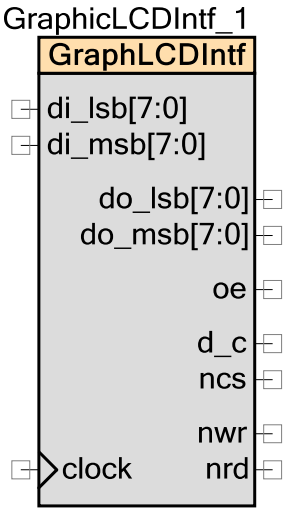
グラフィック LCD インタフェース (GraphicLCDIntf)

1.61

特長



- グラフィック LCD コントローラへの 8 または 16 ビット インタフェース
- 多くのグラフィック コントローラデバイスと互換
- 提供 SEGGER emWin グラフィックライブラリによるインタフェース
- 書き込みおよび読み取り処理の実行
- 読み込み「LOW」パルス幅は、2～255 サイクル
- 読み込み「HIGH」パルス幅は、1～255 サイクル
- 典型的な i8080 インタフェースを実装

概要説明

グラフィック LCD インタフェース (GraphicLCDIntf) コンポーネントは、グラフィック LCD コントローラとドライバデバイスのインタフェースを提供します。これらのデバイスは、一般に、LCD パネルに統合されます。これらのデバイスへのインタフェースは、一般に、i8080 インタフェースと呼ばれています。これは、歴史的な Intel 8080 マイクロプロセッサのパラレルバスインタフェースプロトコルに準拠しています。

このコンポーネントはSEGGER emWinグラフィックスライブラリを用いて動くよう設計されています。このグラフィックスライブラリは Cypress により提供され、Cypress デバイスと共に使用するためのもので、Cypressのウェブサイトである www.cypress.com/go/comp_emWin にあります。グラフィックスライブラリは図やレンダリングテキストおよび画像用のグラフィック機能の完全な機能を備えたセットを提供します。

GraphicLCDIntfを使用する場合

LCD コントローラとドライバデバイスは、一般に、LCD パネルに統合されます。それらはディスプレイ用のフレームバッファへのインタフェースを内蔵または提供し、そのバッファを管理します。GraphicLCDIntf コンポーネントは、このコントローラとの読み取りおよび書き込み処理を実行します。これらの処理には、以下のパラメータがあります:

- 「読み込み」もしくは「書き込み」
- address: d_c ピンで駆動される 1 ビットのアドレス

- Data (データ) (8 または 16 ビット): 書き込みは「do」で送信、読み取りは「di」で読み取ります

GraphicLCDIntf コンポーネントは、多数のコントローラに対応します。このコンポーネントの設定では、3 つのパラメータを使用できます。

- クロック周波数: このコンポーネントを駆動するクロックの周波数は、書き込み信号の最小パルス幅「LOW」によって、しばしば制限されます(この値はグラフィック LCD コントローラデータシートにあります)。書き込みパルスは、クロック1個分の期間「LOW」であるため、この要件を満たすようにクロック周波数を設定してください。
- 読み取りパルス幅「HIGH」: カスタマイザのこの設定は、クロックサイクル単位で評価されます。クロック周期をパルス幅「HIGH」のために設定したサイクル数に掛けたものは、コントローラの読み取りパルス幅「HIGH」の要件を満たす必要があります。
- 読み取りパルス幅「LOW」: パラメータは読み取りパルス幅「HIGH」パラメータと同じ方法で設定されています。読み取りパルス幅「LOW」のタイミングは、読み取りパルス幅のコントローラー要件と読み取りアクセス時間の要件を満たす必要があります。データは、アクティブな「LOW」読み取りパルスが終わる 1 クロック周期前にサンプリングされるため、パルス幅は、十分なアクセス時間を持つよう設定する必要があります。

以下に、該当する LCD コントローラの設定を示します。

Solomon Systech SSD1289

- クロック周波数: 20 MHz (50 ns)
- 読み取りパルス幅「HIGH」: 10 クロック周期 (500 ns)
- 読み取りパルス幅「LOW」: 10 クロック周期 (500 ns)

Solomon Systech SSD2119

- クロック周波数: 25 MHz (40 ns)
- 読み取りパルス幅「HIGH」: 13 クロック周期 (500 ns)
- 読み取りパルス幅「LOW」: 13 クロック周期 (500 ns)

Himax HX8347A

- クロック周波数: 28.5 MHz (35 ns)
- 読み取りパルス幅「HIGH」: 3 クロック周期 (105 ns)
- 読み取りパルス幅「LOW」: 11 クロック周期 (385 ns)

ILITEK ILI9325

- クロック周波数: 20 MHz (50 ns)
- 読み取りパルス幅「HIGH」: 3 クロック周期 (150 ns)
- 読み取りパルス幅「LOW」: 3 クロック周期 (150 ns)

Epson S1D13743

- クロック周波数: 33 MHz (33.3 ns)
- 読み取りパルス幅「HIGH」: 2 クロック周期 (67 ns)
- 読み取りパルス幅「LOW」: 5 クロック周期 (167 ns)

入出力接続

このセクションは GraphicLCDIntf コンポーネントの入出力接続を説明します。I/O リストのアスタリスク (*) は、その I/O の説明でリストされている条件において、I/O が*シンボル(ワイルドカード)に隠れている可能性があることを示します。

クロック

このコンポーネントを駆動するクロック。GraphicLCDIntf は、コンポーネントに接続されているクロックが1周期以上あれば完全に動作します。

di_lsb[7:0]

入力データバスの下位 8 ビット。読み込み処理中のデータに使用。

デバイスの入力ピンに接続し、このピンに対する「入力同期」選択をディスエーブルにします。同期出力信号を基にして駆動されるため、信号そのものは本質的には同期化されます。

di_msb[7:0] *

入力データバスの上位 8 ビット。読み込み処理中のデータに使用。16 ビット インタフェースモードのときのみ存在。

信号をデバイスの入力ピンに接続し、このピンに対する「入力同期」選択をディスエーブルにします。同期出力信号を基にして駆動されるため、信号そのものは本質的には同期化されます。

do_lsb[7:0]

出力データバスの下位 8 ビット。書き込み処理中のデータで使用されます。

do_msb[7:0] *

出力データバスの上位 8 ビット。書き込み処理中のデータで使用されます。16 ビット インタフェースモードのときのみ存在。

oe

データ バスの出力イネーブル。これは通常データバスの入出力ピンコンポーネントの出力イネーブルに接続されています。この信号が使用される方法については、「図解マクロ情報」を参照してください。

d_c

データ/コマンド信号。「HIGH」の場合はデータ処理、「LOW」の場合はコマンド処理を示します。

ncs

アクティブ「LOW」チップ選択。

nwr

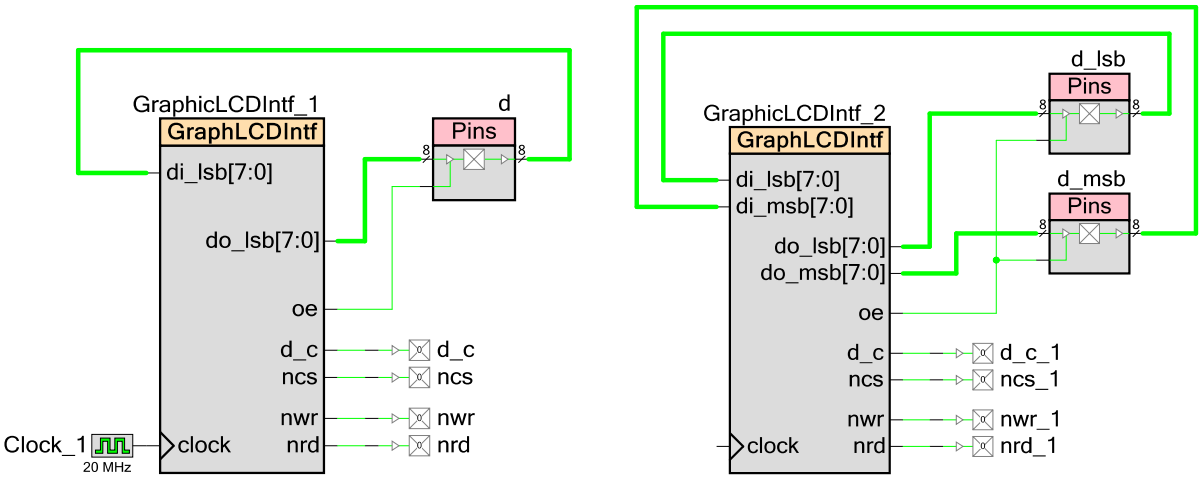
アクティブ「LOW」書き込み制御信号。

nrd

アクティブ「LOW」読み取り制御信号。

回路図マクロ情報

コンポーネントカタログには、標準の記号エントリに加え、2 つのマクロが PSoC Creator により提供されています。1 つは、ピンおよびクロックに接続される 8 ビットの実装用マクロです。もう 1 つは、ピンおよびクロックに接続される 16 ビットの実装用マクロです。

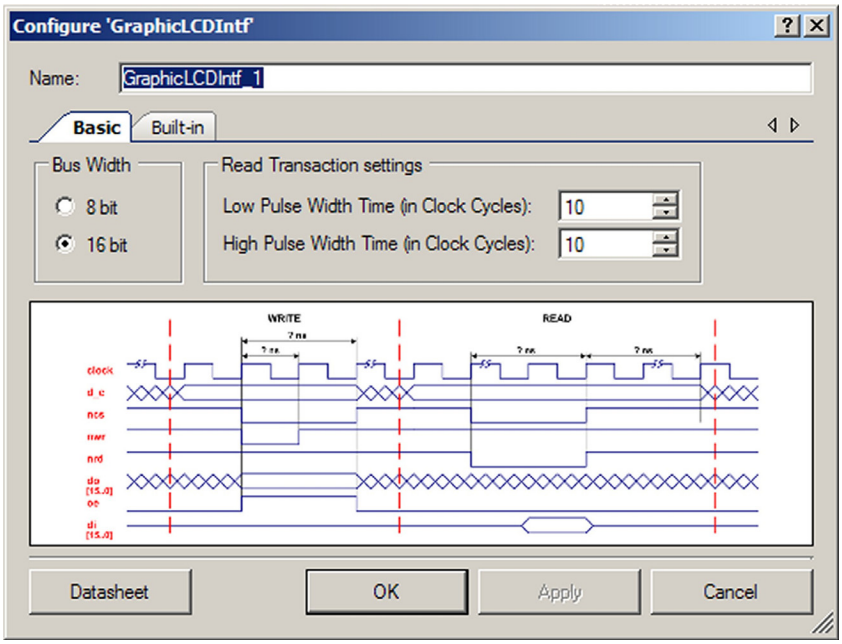


それぞれのマクロでは、クロックが 20 MHz に設定され、パルス幅設定は初期設定のままになります。これが SSD1289 コントローラ用の適切な設定です。

「同期入力」オプションは、すべてのデータ ピンで選択が解除され、すべてのピンの API 生成はオフになります。

コンポーネントパラメータ

GraphicLCDIntf コンポーネントを設計図上にドラッグし、ダブルクリックして **[Configure] (設定)** ダイアログを開きます。初期設定の GraphicLCDIntf 設定は、Solomon Systech SSD1289 コントローラを操作するための適切な設定です。



バス幅

コンポーネントがグラフィック LCD コントローラへの 8 または 16 ビット パラレルインタフェースをサポートするか決定します。初期設定は **16 ビット**です。

Low Pulse Width Time (「LOW」パルス幅時間)

コントローラで、読み取りパルス幅「LOW」に必要なクロック周期数を決定します。この値は、2～255 クロック周期に設定できます (読み取り値は、パルス終了の 1 クロック前にサンプリングされる必要があるため、最小は 2 です)。初期設定は **10** です。

High Pulse Width Time (「HIGH」パルス幅時間)

コントローラで、読み取りパルス幅「HIGH」に必要なクロック周期数を決定します。この値は、1～255 クロック周期に設定できます。初期設定は **10** です。

クロックの選択

このコンポーネントには、内部クロックはありません。クロックソースを必ず取りつけてください。このコンポーネントは、それに接続されているクロック1個から動作します。

配置

GraphicLCDIntf は、UDB アレイ全体に配置され、すべての配置情報は、*cyfitter.h* ファイルを通して API に提供されます。

リソース

リソース	リソースのタイプ			API メモリ (バイト)		ピン (外部入出力ごと)
	データパス セル	PLD	ステータス セル	フラッシュ	RAM	
8ビットインタフェース	1	4	2	109	1	12
16ビットインタフェース	2	4	3	120	1	20

アプリケーション プログラミング インタフェース

アプリケーションプログラミングインタフェース (API) ルーチンにより、ソフトウェアを使用してコンポーネントを設定できます。次の表は、各関数へのインタフェースとその説明を示しています。続くセクションでは、各関数について詳しく説明します。

初期設定において PSoC Creator は特定の設計のコンポーネントの最初のインスタンスに対して、インスタンス名「GraphicLCDIntf_1」を割り当てます。インスタンス名は、識別子の構文ルールに従った固有の値に変更できます。インスタンス名は、コンポーネントで生成されるすべてのグローバル関数名、変数、定数記号の接頭辞になります。分かりやすくするために、次の表で使用されているインスタンス名は「GraphicLCDIntf」とします。

関数	説明
GraphicLCDIntf_Start()	GraphicLCDIntfインタフェースを開始します。
GraphicLCDIntf_Stop()	GraphicLCDIntfインタフェースをディスエーブルにします。
GraphicLCDIntf_Write8()	8 ビットパラレル インタフェースで、書き込み処理を開始します。
GraphicLCDIntf_Write16()	16 ビットパラレルインタフェースで、書き込み処理を開始します。
GraphicLCDIntf_Read8()	8 ビットパラレルインタフェースで、読み取り処理を開始します。
GraphicLCDIntf_Read16()	16 ビットパラレルインタフェースで、読み取り処理を開始します。
GraphicLCDIntf_Sleep()	設定を保存し、GraphicLCDIntfをディスエーブルにします。
GraphicLCDIntf_Wakeup()	設定を復元し、GraphicLCDIntfをイネーブルにします。
GraphicLCDIntf_Init()	GraphicLCDIntf初期設定を初期化または復元します。
GraphicLCDIntf_Enable()	GraphicLCDIntfをイネーブルにします。
GraphicLCDIntf_SaveConfig()	GraphicLCDIntfの構成を保存します。
GraphicLCDIntf_RestoreConfig()	GraphicLCDIntfの構成を復元します。

グローバル変数

変数	説明
GraphicLCDIntf_initVar	グラフィックLCDインタフェースが初期化されたかどうかを示します。変数は、0に初期化され、最初にGraphicLCDIntf_Start() が呼び出されると1に設定されます。GraphicLCDIntf_Start() ルーチンを最初に呼び出してから、再初期化することなく、コンポーネントが再起動できるようにします。 コンポーネントの再初期化が必要な場合、GraphicLCDCtrl_Start()もしくはGraphicsLCDCtrl_Enable()関数を呼び出す前に、GraphicLCDCtrl_Init()関数を呼び出すこともできます。

void GraphicLCDIntf_Start(void)

説明: この関数はActive modeパワーテンプレートビットまたはクロックのゲーティングを必要に応じてイネーブルにします。操作するコンポーネントを設定します。

パラメータ: なし

戻り値: なし

副作用: なし

void GraphicLCDIntf_Stop(void)

説明: この関数はActive modeパワーテンプレートビットまたはゲートクロックを必要に応じてディスエーブルにします。

パラメータ: なし

戻り値: なし

副作用: なし

void GraphicLCDIntf_Write8(uint8 d_c, uint8 data)

説明: この関数は8 ビットパラレルインタフェースでの書き込み処理を開始します。この書き込みはposted(要求配信)書き込みであるため、この関数は、インタフェースに対し書き込みが実際に終了する前に戻ります。コマンドのキューがいっぱいである場合は、この関数は、この書き込みリクエストのスペースが空くまで、戻り値を返しません。

パラメータ: d_c: データ (1) またはコマンド (0) を示します。d_c ピンに渡されます

data: do_lsb[7:0] ピンで送信されるデータ

戻り値: なし

副作用: なし

void GraphicLCDIntf_Write16(uint8 d_c, uint16 data)

説明: この関数は 16ビットパラレルインタフェース上の書き込み処理を開始します。この書き込みはposted(要求配信)書き込みであるため、この関数は、インタフェースに対し書き込みが実際に終了する前に戻ります。コマンドのキューがいっぱいである場合は、この関数は、この書き込みリクエストのスペースが空くまで、戻り値を返しません。

パラメータ: d_c: データ (1) またはコマンド (0) を示します。d_c ピンに渡されます

data: do_msb[7:0] (上位バイト) および do_lsb[7:0] (下位バイト) ピンに送られるデータ

戻り値: なし

副作用: なし

uint8 GraphicLCDIntf_Read8(uint8 d_c)

説明: この関数は8ビットパラレルインタフェース上で書き込み処理を開始します。現在要求されている書き込みが全て完了した後に、読み込みが実行されます。この関数は、読み込みが完了するまで待機し、その後に読み込み値を返します。

パラメータ: d_c: データ (1) またはコマンド (0) を示します。d_c ピンに渡されます。

戻り値: di_lsb[7:0] ピンからの 8 ビットの読み取り値

副作用: なし

uint16 GraphicLCDIntf_Read16(uint8 d_c)

説明: この関数は 16ビットパラレルインタフェース上で読み取り処理を開始します。現在要求されている書き込みが全て完了した後に、読み込みが実行されます。この関数は、読み込みが完了するまで待機し、その後に読み込み値を返します。

パラメータ: d_c: データ (1) またはコマンド (0) を示します。d_c ピンに渡されます。

戻り値: do_msb[7:0] (最上位バイト) と do_lsb[7:0] (最下位バイト) ピンからの 16 ビットの読み取り値

副作用: なし

void GraphicLCDIntf_Sleep(void)

説明: これは、コンポーネントのスリープを準備するのに推奨されるルーチンです。GraphicLCDIntf_Sleep()ルーチンは、現在のコンポーネントの状態を保存します。次に、GraphicLCDIntf_Stop() 関数、さらに GraphicLCDIntf_SaveConfig() 関数を呼び出して、ハードウェアの設定を保存します。必要に応じて Active mode パワーテンプレートビットまたはクロックゲーティングをディスエーブルにします。

CyPmSleep() または CyPmHibernate() 関数を呼び出す前に、GraphicLCDIntf_Sleep() 関数を呼び出します。電源管理関数については、*『PSoC Creator システム・リファレンスガイド』*を参照してください。

パラメータ: なし

戻り値: なし

副作用: なし

void GraphicLCDIntf_Wakeup(void)

説明: これは、GraphicLCDIntf_Sleep() が呼び出された際に、コンポーネントにその状態を復元するための、望ましいルーチンです。GraphicLCDIntf_Wakeup() 関数は、GraphicLCDIntf_RestoreConfig() 関数を呼び出して、設定を復元します。GraphicLCDIntf_Sleep() 関数が呼び出される前にコンポーネントがイネーブルされていた場合、GraphicLCDIntf_Wakeup() 関数はコンポーネントも再度イネーブルにします。必要に応じてActive mode パワーテンプレートビットまたはクロックゲーティングをイネーブルにします。

パラメータ: なし

戻り値: なし

副作用: 最初に GraphicLCDIntf_Sleep() または GraphicLCDIntf_SaveConfig() 関数を呼び出さずに GraphicLCDIntf_Wakeup() 関数を呼び出すと、予期せぬ動作が起きることがあります。

void GraphicLCDIntf_Init(void)

説明: この関数はカスタマイザの [Configure] (設定) ダイアログの設定に従って、コンポーネントを初期化または復元します。GraphicLCDIntf_Start() ルーチンが GraphicLCDIntf_Init() を呼び出し、これがコンポーネントの動作を開始する望ましい方法であるため、GraphicLCDIntf_Init() を呼び出す必要はありません。読み取り「LOW」および「HIGH」パルス幅を定義する静的コンポーネント設定のみが、初期値に復元されます。

パラメータ: なし

戻り値: なし

副作用: これはコンポーネントを再初期化しますが、FIFO からのデータはクリアしません。そしてコンポーネントハードウェアステートマシンをリセットしません。現在の処理は、バス上で実行されます。

void GraphicLCDIntf_Enable(void)

説明: この関数はハードウェアを起動し、コンポーネントの動作を開始します。GraphicLCDIntf_Start() ルーチンがこの関数を呼び出すので、GraphicLCDIntf_Enable() を呼び出す必要がありません。これはコンポーネントの操作を開始するときに推奨される方法です。

パラメータ: なし

戻り値: なし

副作用: なし

void GraphicLCDIntf_SaveConfig(void)

説明: この関数は、コンポーネントの設定と一時保持レジスタ内容を保存します。この関数は、[Configure] ダイアログで定義されている、または該当する API で変更される、現在のコンポーネントパラメータ値も保存します。この関数は、GraphicLCDIntf_Sleep() 関数によって呼び出されます。読み取り「LOW」および「HIGH」パルス幅を定義する、コンパイル時間コンポーネント設定は保存されています。

パラメータ: なし

戻り値: なし

副作用: なし

void GraphicLCDIntf_RestoreConfig(void)

説明: この関数はGraphicLCDIntfの維持されないレジスタの構成を復元します。GraphicLCDIntf_Wakeup によって API が呼び出され、コンポーネントの一時保存レジスタ内容を復元します。

パラメータ: なし

戻り値: なし

副作用: このAPI がGraphicLCDIntf_SaveConfigの前に呼び出される場合は、読み取り「LOW」および「HIGH」パルス幅のコンポーネント設定が、カスタマイザで提供されている値に復元されます。

ファームウェアソースコードのサンプル

PSoC Creator は、[Find Example Project（サンプルプロジェクトを検索）] ダイアログに多数のサンプルプロジェクトを提供しており、そこには回路図およびサンプル コードが含まれています。コンポーネント固有の例を見るには、[Component Catalog] または回路図中当該コンポーネントのインスタンスからダイアログを開きます。一般例については、[Start Page] または **[File（ファイル）]** メニューからダイアログを開きます。必要に応じてダイアログにある **Filter Options** を使用し、選択できるプロジェクトのリストを絞り込みます。

詳細は PSoC Creator ヘルプの「Find Example Project（プロジェクト例を見つける）」を参照してください。

機能説明

バスの処理

このインタフェースは、読み取りまたは書き込み処理のいずれかを実行します。これらの処理には、以下のパラメータがあります:

- 「読み込み」もしくは「書き込み」
- address: この場合、d_c ピン上で駆動されている 1 ビット アドレスです
- Data (データ) (8 または 16 ビット): 書き込みは「do」で送信、読み取りは「di」で読み取られます。

実装は、CPU が (データで使用されるのと同じ) FIFO を使用して、コマンド バイトをコンポーネントに送信することを想定しています。このコマンド バイトが読み取りまたは書き込みを示し、d_c ビットを提供します。

アイドル状態

インタフェース上で読み取りも書き込みも行われていないとき、インタフェースはアイドル状態にあります。この条件での出力ピンの値は、以下の通りです。

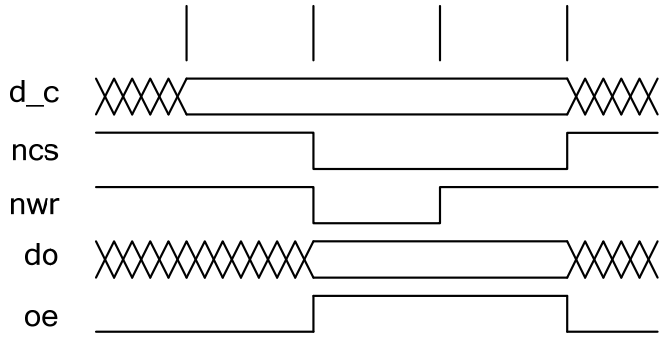
- d_c: Don't care(影響なし、通常は、最後の状態のまま)
- ncs: 1
- nwr: 1
- nrd: 1
- do: Don't care(影響なし、通常は、最後の状態のまま)
- oe: 0

読み取りおよび書き込み処理の説明で、リストされていない信号はアイドルです。

書き込み処理

図 1 はパラレルインタフェースでの書き込み処理のタイミング図です。

図 1. 書き込み処理タイミング図



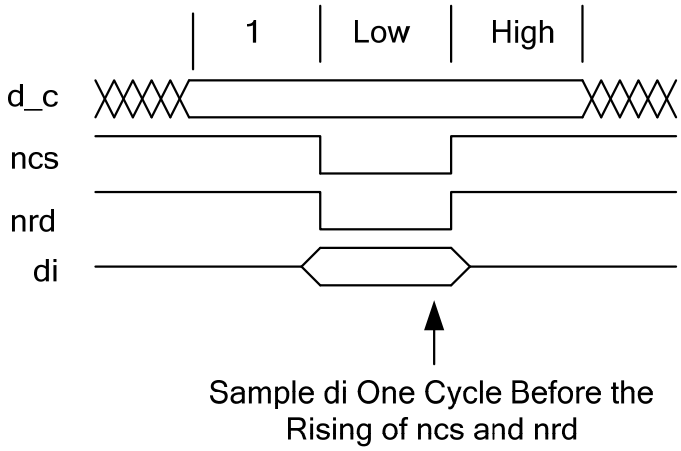
この図は、書き込み処理が 3 つのクロック周期を必要としていることを示しています。タイミング図は、ビット幅に関わらず同じです。この処理は、直ちに別の読み取りまたは書き込み処理の前または後に続いて実行できます。または、書き込み処理の前または後で、アイドル状態になる場合があります。

CPU へのインタフェースにより、CPU が書き込み要求を要求できるようになります (アドレスとデータを提供し書き込みをリクエストしてから、処理がパラレルバスで実際に完了する前に実行します)。実装により、CPU を著しくストールさせることなく、CPU が 2 つの書き込みリクエストを持つことができます。

読み込み操作

図 2 はパラレルインタフェース上の読み取り処理のタイミング図です。

図 2. 読み取り処理タイミング図



この図は、読み取り処理が、読み取りパルス幅の「LOW」および「HIGH」設定により、さまざまなクロック周期数を必要とすることを示しています。タイミング図は、ビット幅に関わらず同じです。データ入力は、ncs と nrd

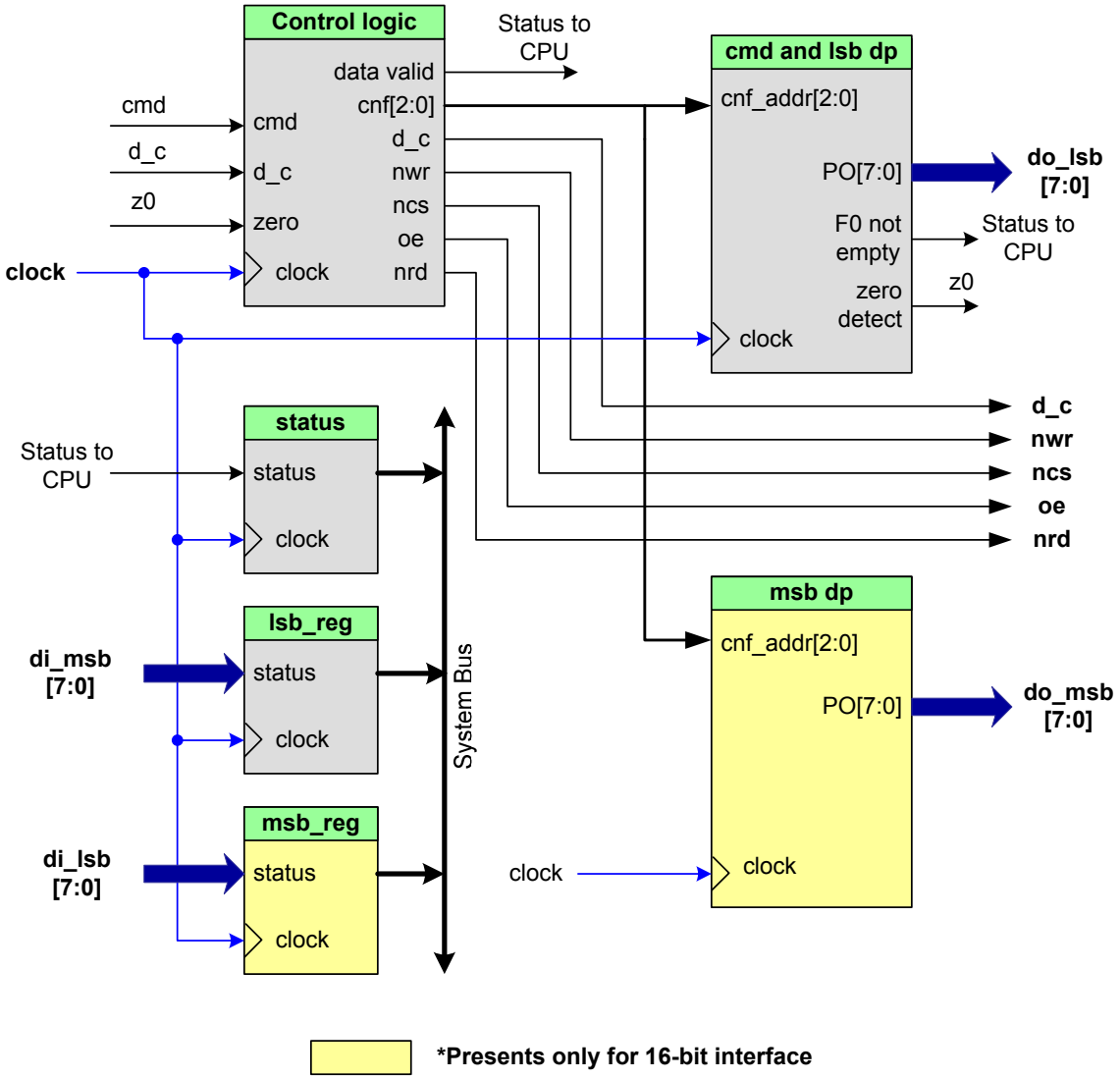
「LOW」パルスの終わりの前に、クロック周期 1 回をサンプリングすることに注意してください。この処理は、直ちに別の読み取りまたは書き込み処理の前または後に続いて実行できます。または、読み取り処理の前または後で、アイドル状態になる場合があります。

読み取りおよび書き込みの順序は維持されます (読み取りは、書き込み要求が完了する前に実行されます)。読み取りでは、CPU が実行前に、読み取り処理の完了を待つ必要があります。

ブロック図と設定

GraphicLCDIntfコンポーネントは、設定されているUDBセットとして実装されます。図 3 はこの実装を示します。

図 3. ブロック図



レジスタ

GraphicLCDIntf_STATUS_REG

ビット	7	6	5	4	3	2	1	0
値	予約済み						data_valid	F0_half_ empty

- F0_half_empty: 設定されると、コマンド/データ FIFO に 2 バイト以上の空きがあります。
- data_valid: CPU にとって読み取りデータが有効な場合にセットされます。このビットは、CPU がレジスタを読み取る際にクリアされます。

GraphicLCDIntf_DIN_LSB_DATA_REG

ビット	7	6	5	4	3	2	1	0
値	di_lsb[7:0]							

- 読み取り処理の、入力データバスの下位 8 ビット。

8ビットインタフェースでは、GraphicLCDIntf_Read8() API 関数でレジスタ値を読み取ることができます。値は、16ビット インタフェースでは、GraphicLCDIntf_Read16() API 関数から返される値のLSB(最も重要度が低いバイト)です。

GraphicLCDIntf_DIN_MSB_DATA_REG

ビット	7	6	5	4	3	2	1	0
値	di_msb[7:0]							

- 読み取り処理用の入力データバスの上位 8 ビット

レジスタ値は、16 ビット インタフェースでは、GraphicLCDIntf_Read16() API 関数から返される値の MSB(最も重要度が高いバイト)です。

注 DIN_LSB_DATA_REG および DIN_MSB_DATA_REG ビットは、CPU ファームウェアがこれらのレジスタを読み取る際にクリアされます。

DC 電気的特性と AC 電気的特性

以下の値は、期待される性能を示しており、初期特性データを基にしています。

「名目配線での最大」タイミング特性

パラメータ	説明		Min	Typ	Max [1]	単位
f_{CLOCK}	コンポーネントクロック周波数		–	–	33	MHz
t_{AS}	アドレスセットアップタイム		1	–	–	t_{CY_clock} [2]
t_{PWLW}	パルス幅「LOW」書き込み		–	1	–	t_{CY_clock}
t_{PWHW}	パルス幅「HIGH」書き込み		3	–	–	t_{CY_clock}
t_{PWLR}	パルス幅「LOW」読み取り		2	–	255	t_{CY_clock}
t_{PWHR}	パルス幅「HIGH」読み取り		1	–	255	t_{CY_clock}
t_{AH}	アドレスホールド時間					
		書き込み	2	–	–	t_{CY_clock}
		読み取り	t_{PWHR}	–	–	t_{CY_clock}
t_{CYCLE}	クロックサイクル期間					
		書き込みサイクル	4	–	–	t_{CY_clock}
		読み込みサイクル	t_{PWLR} + t_{PWRH} + 1	–	–	t_{CY_clock}
t_{DSW}	データセットアップ時間		–	1	–	t_{CY_clock}
t_{DHW}	データホールド時間		–	1	–	t_{CY_clock}
t_{ACC}	データアクセス期間		–	t_{PWHR} – 1	–	t_{CY_clock}
t_{DHR}	出力ホールドタイム		–	0	–	t_{CY_clock}

[1]これらの「名目値」は、通常ルーティング状況における、コンポーネントの最大安全動作周波数です。コンポーネントをより高いクロック周波数で実行することができますが、STA 結果を使用して、タイミング要件を検証する必要があります。

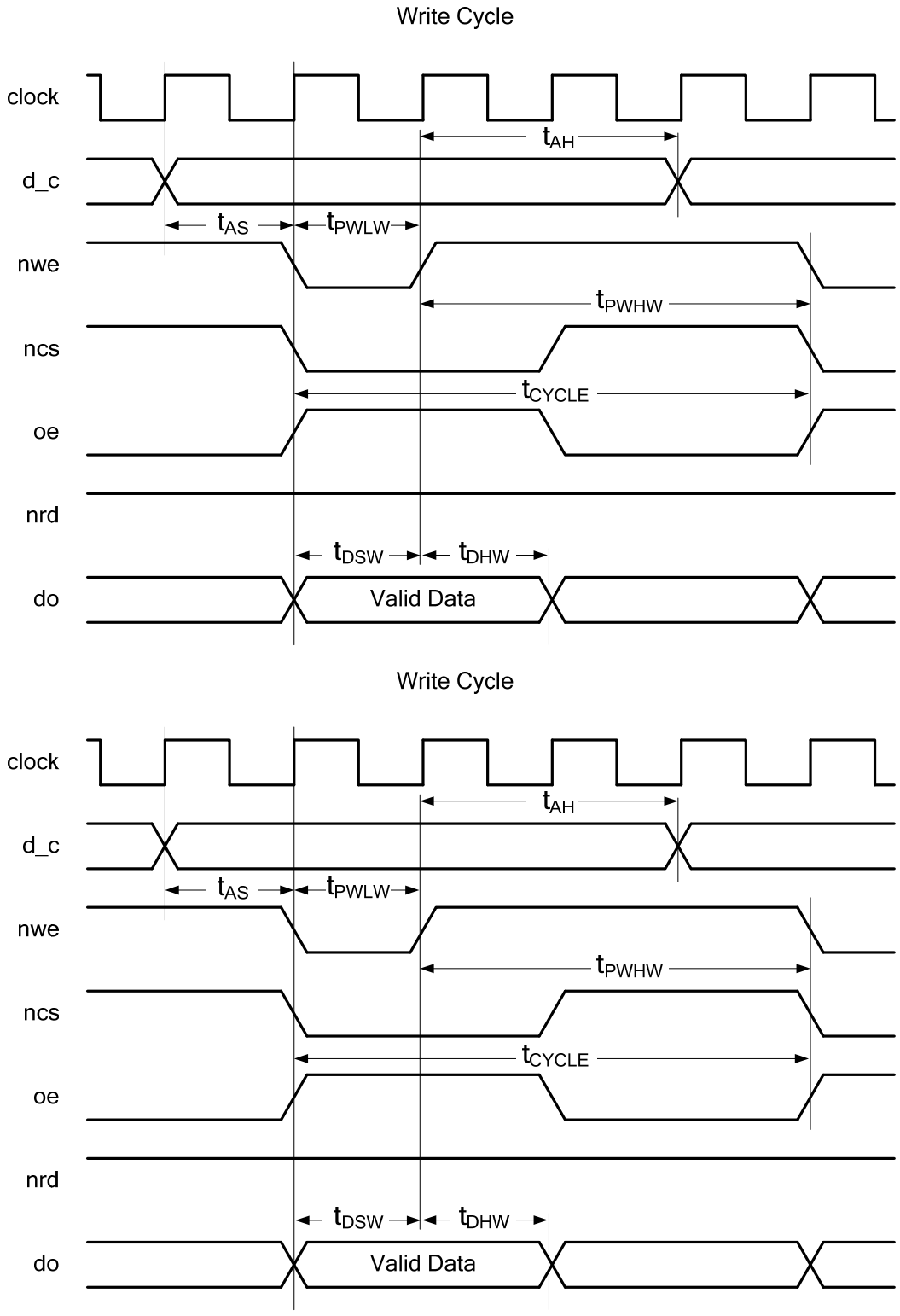
[2] $t_{CY\_clock} = 1/f_{CLOCK}$. これは 1 クロック周期のサイクルタイムです

「すべての配線での最大」タイミング特性

パラメータ	説明		Min	Typ	Max[3]	単位
f_{CLOCK}	コンポーネントクロック周波数		–	–	25	MHz
t_{AS}	アドレスセットアップタイム		1	–	–	t_{CY_clock}
t_{PWLW}	パルス幅「LOW」書き込み		–	1	–	t_{CY_clock}
t_{PWHW}	パルス幅「HIGH」書き込み		3	–	–	t_{CY_clock}
t_{PWLR}	パルス幅「LOW」読み取り		2	–	255	t_{CY_clock}
t_{PWHR}	パルス幅「HIGH」読み取り		1	–	255	t_{CY_clock}
t_{AH}	アドレスホールド時間					
		書き込み	2	–	–	t_{CY_clock}
		読み取り	t_{PWHR}	–	–	t_{CY_clock}
t_{CYCLE}	クロックサイクル期間					
		書き込み	4	–	–	t_{CY_clock}
		読み取り	t_{PWLR} + t_{PWRH} + 1	–	–	t_{CY_clock}
t_{DSW}	データセットアップ時間		–	1	–	t_{CY_clock}
t_{DHW}	データホールド時間		–	1	–	t_{CY_clock}
t_{ACC}	データアクセス期間		–	t_{PWHR} – 1	–	t_{CY_clock}
t_{DHR}	出力ホールドタイム		–	0	–	t_{CY_clock}

[3]「すべての配線」で最大とはコンポーネントインスタンスがこれらの動作速度以下で(ゆっくり)作動する場合、適合タイミングはこのコンポーネントにとって問題ではないことを意味します。

図 4. データ 処理のタイミング図



特性データ用の STA 結果の使用方法

名目配線の最大値は、静的タイミング分析 (STA) を使って、複数のテストパスから収集されます。次の方法を用いて、STA 結果と共に設計の最大値を計算することができます。

f_{CLOCK} 最大コンポーネント・クロック周波数は、名前の付いたコンポーネント・クロック (この場合は CLK) として、クロック要約の中のタイミング結果の中に出ています。次の図表は、クロック制限の例を示します。

- Clock Summary Section

Clock	Type	Nominal Frequency	Required Frequency	Maximum Frequency	Violation
CLK	Sync	20.000 MHz	20.000 MHz	57.019 MHz	
ClockBlock/clk_bus	Async	60.000 MHz	60.000 MHz	N/A	
ClockBlock/dclk_0	Async	20.000 MHz	20.000 MHz	N/A	
CyBUS_CLK	Sync	60.000 MHz	60.000 MHz	N/A	
CyILO	Async	1.000 kHz	1.000 kHz	N/A	
CyIMO	Async	3.000 MHz	3.000 MHz	N/A	
CyMASTER_CLK	Sync	60.000 MHz	60.000 MHz	N/A	
CyPLL_OUT	Async	60.000 MHz	60.000 MHz	N/A	

残りのパラメータは、実装固有のものであり、クロックサイクル単位で表されます。これらは、2 つのカテゴリに分けることができます。

- コンポーネントの設定に用いられるパラメータは:

  t_{PWLW}　書き込み信号の最小パルス幅「LOW」時間

  t_{PWLR}　読み取り信号の最小パルス幅「LOW」時間

  t_{PWHR}　読み取り信号の最小パルス幅「HIGH」時間

  コンポーネントを設定する際に、これらのパラメータをどのように使用するかについての詳しい説明は、1 ページの「GraphicLCDIntf を使用する場合」に記載されています。

- コンポーネントの実装に基づいて確定されるパラメータは:

  t_{PWHW}　書き込み信号の最小パルス幅「HIGH」時間

  t_{AS}　nwr/nrd 信号の立ち下がりエッジ前に、アドレス信号が有効になる最小時間

  t_{AH}　nwr/nrd 信号の立ち上がりエッジ後に、アドレス信号が有効になる最小時間

  t_{CYCLE}　一回の処理(書き込み/読み取り) がインタフェースで実行される期間

  t_{DSW}　書き込み信号の立ち上がりエッジ前に、データが有効になる最小時間

  t_{DHW}　書き込み信号の立ち上がりエッジの後に、データが有効になる最小時間

  t_{ACC}　読み取り信号の負のエッジ後に、データがサンプリングされる最小時間

  t_{DHR}　nrd 信号の立ち上がりエッジ後に、データが有効になる最小時間

コンポーネントの変更

ここでは、過去のバージョンからコンポーネントに加えられた主な変更を示します。

バージョン	変更の説明	変更の理由 / 影響
1.61	.cyreファイルに含めるとき、すべてのコンポーネント API に CYREENTRANTキーワードを追加	すべてのAPIが真に再入可能とは限りません。コンポーネント API ソースファイルに含まれているコメントは、どの関数が(再入可能)候補であるかを示しています。 この変更では、安全な(フラグまたはクリティカルセクションによる並行コールから保護された)方法で使用される再入可能ではない関数のコンパイラ警告を制限する必要があります。
	コンポーネント中の不正なタイミング パスをマークするために、タイミング制約を追加	使用されていないパスをタイミング分析から取り除きます。これにより、誤ったタイミング違反メッセージを回避することができます。
1.60.a	データシートからの関連キットへの参照を削除	
1.60	DPクロックに対するFIFOブロックステータス信号を再サンプル	コンポーネントが、すべてのSoC 3およびPSoC 5シリコンで同じタイミング結果で機能することを可能にします。
	データシートに特性データを追加	
	データシートのマイナーな編集と更新	