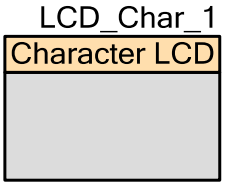
キャラクタ LCD

1.50



特長

- 業界標準の Hitachi HD44780 LCD ディスプレイ ドライバ チッププロトコルを実装しています。
- 1 つの I/O ポートで 7 つの I/O ピンのみを必要とします。
- ユーザ定義のカスタム文字を作成する、内蔵文字エディタを搭載しています。
- 水平および垂直棒グラフをサポートしています。

概要説明

キャラクタ LCD コンポーネントには、Hitachi 44780 標準 4 ビット インターフェースに従う、1、2 または 4 行 LCD モジュールを簡単に使用できる、一連のライブラリ ルーチンが含まれています。コンポーネントは、水平および垂直棒グラフを実装する API を提供します。または、独自のカスタム文字を作成し、表示できます。

キャラクタLCDを使用する場合

製品のユーザにテキスト データを表示する、または設計やデバッグ中に開発者にテキスト データを表示する場合に、キャラクタ LCD コンポーネントを使用します。

入出力接続

ここでは、キャラクタ LCD のさまざまな入出力接続について説明します。

LCD_Port – ピン エディタ

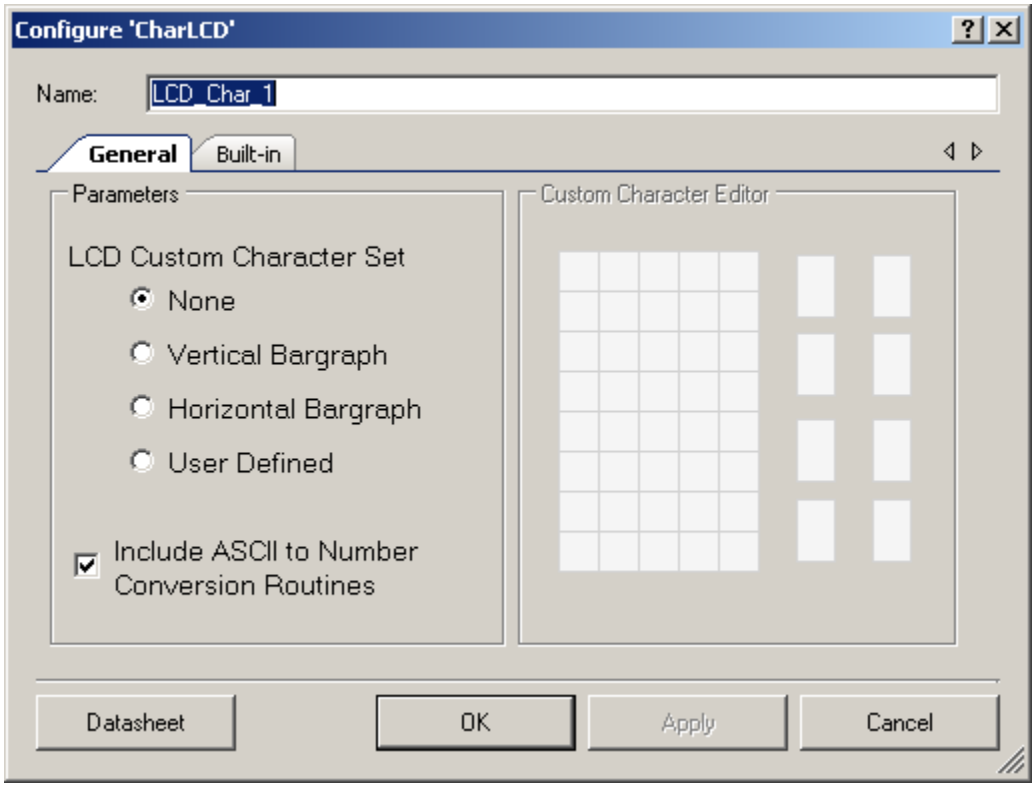
LCD は物理ポートの 7 つの連続したピンを使用します。希望するポートにキャラクタ LCD を配置するには、デザイン ワイド リソース ピン エディタを使用します。ピン エディタを使うと、このコンポーネントのデジタル ポートを空いているどの出力ポートにでも配置できます。

注 選択したポートのピン 1 または 0 で開始するために、7 つのピンを配置できますが、ポートをスパンしない場合があります。これらのピンは LCD ポートにのみ使用され、他の目的には利用できません。

ソフトウェア API がすべての読み取りと書き込みを管理するため、キャラクタLCDポートへの直接アクセスは必要ありません。LCD モジュールと PSoC 論理ポート間のピン接続は、「機能説明」で詳しく解説します。

コンポーネント パラメータ

キャラクタ LCD コンポーネントを設計上にドラッグし、ダブルクリックして [Configure] (設定) ダイアログを開きます。



パラメータ

LCD Custom Character Set (LCD カスタム文字セット)

このパラメータは次のオプションを選択できます:

- **None** (デフォルト) – カスタム文字について何もしない。
- **Vertical Bargraph (垂直棒グラフ)** – カスタム文字と API を生成し、垂直棒グラフを操作します。
- **Horizontal Bargraph (水平棒グラフ)** – カスタム文字と API を生成し、水平棒グラフを操作します。
- **User Defined (ユーザ定義)** – カスタム文字と API を作成して、操作します。

コンポーネントが文字をロードした後は、LCD_Char_PutChar() 関数とカスタム文字定数 (ヘッダ ファイルから) を使用して、文字を表示できます。

変換ルーチン

Include ASCII to Number Conversion Routines オプションを選択すると、生成されるコードに複数の API 関数が追加されます。(API テーブルと関数の説明については、これらのルーチンの説明を参照してください。)

Custom Character Editor (カスタム文字エディタ)

Custom Character Editor を使うと、GUI を使って、ユーザ定義の文字セットを簡単に作成できるようになります。それぞれ 8 文字は、5x8 ピクセルまでのサイズが可能です。一部のハードウェアでは、5x7 を超えるピクセルは表示できない場合があります。

Custom Character Editor を使用するには、**User Defined** を **LCD Custom Character Set** のオプションとして選択します。次に、編集する文字のサムネイルをクリックします。

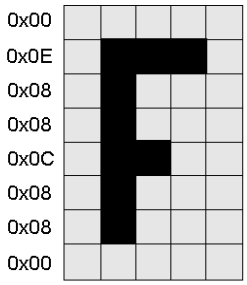
文字のピクセルを切り替えるには、拡大文字ビューで選択されているピクセルをクリックします。複数のピクセルをクリックしてドラッグし、切り替えることもできます。

カスタム文字セットの作成後は、GUI が 8 つのカスタム文字のルックアップ配列を生成します。これで、ルックアップ配列を LCD モジュールにロードできます。デフォルトで、選択または作成された場合に、LCD_Char_Start() ルーチンがカスタム文字をロードします。

コンポーネントの機能により、カスタム文字セットをコードに作成し、実行時にロードできるようになります。その場合、最後にロードした文字セットは以前に文字セットに上書きされて有効になります。コンポーネントの GUI を使用して作成した元のカスタム文字セットを復元するには、ソース コード extern uint8 const CYCODE LCD_Char_customFonts[]; に次の行を追加する必要があります。実行時に、LCD_Char_LoadCustomFonts() はそのコードをパラメータとして使用し、LCD モジュールに元の文字セットをロードできます。

図 1 は、8 バイトのカスタム文字ルックアップ配列行にエンコードされたカスタム文字を示しています。

図 1. カスタム文字エンコーディング




```
        Custom character «F»:
{0x00, 0x0E, 0x08, 0x08, 0x0C, 0x08, 0x08, 0x00}
```

図に示されているように、文字の各行は、単一のバイトとしてエンコードされます。ここでは 5 つの最下位ビットのみが使用されます。上位行の最初の文字は、カスタムフォント配列の最初のバイトでエンコードされます。最初の文字の次行は配列の 2 バイト目です。2 番目の文字の最初の行は配列の 9 バイト目というように続きます。カスタム フォント アレイ全体は、合計のアレイ サイズが 64 バイトとなる、8 つのカスタム文字から構成されます。

リソース

構成	デジタル ブロック	API メモリ (バイト)		ピン (外部入出力ごと)
		フラッシュ	RAM	
なし	0	685	3	7
垂直	0	1276	3	7
水平	0	1276	3	7
ユーザ定義	0	823	3	7
なし + 変換ルーチン	0	932	3	7

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。次の表に、各機能に対するインタフェースを「include」ファイルによって提供される関連定数とともに示します。次のセクションでは、各関数について詳しく説明します。

デフォルトで、PSoC Creator は、インスタンス名「LCD_Char_1」を、特定のプロジェクトにおける最初のコンポーネント インスタンスに割り当てます。コンポーネントのインスタンス名は、識別子の文法ルールに従って固有の名前に変更できます。インスタンス名は、すべてのグローバル関数名、変数名、定数名の接頭辞になります。分かりやすいよう、次の表ではインスタンス名「LCD_Char」を使用しています。

関数	説明
LCD_Char_Start()	モジュールを開始し、カスタム文字セットが定義されている場合は、それを LCD にロードします。
LCD_Char_Stop()	LCD をオフにします。
LCD_Char_DisplayOn()	LCD モジュールの表示をオンにします。
LCD_Char_DisplayOff()	LCD モジュールの表示をオフにします。
LCD_Char_PrintString()	Null で終わる文字列を、文字ごとに画面に表示します。
LCD_Char_PutChar()	現在の位置で、単一の文字を LCD モジュール データ レジスタに送信します。
LCD_Char_Position()	カーソルを提供された行と列に一致する位置に設定します。
LCD_Char_WriteData()	1 バイトのデータを LCD モジュール データ レジスタに書き込みます。

関数	説明
LCD_Char_WriteControl()	LCD モジュール制御レジスタに単一のバイト命令を書き込みます。
LCD_Char_ClearDisplay()	LCD モジュール画面からデータをクリアします。
LCD_Char_IsReady()	レディ ビットが設定されるまで、LCD をポーリングします。
LCD_Char_Sleep()	コンポーネントのスリープ モードを準備します。
LCD_Char_Wakeup()	コンポーネント構成を復元し、LCD をオンにします。
LCD_Char_Init()	コンポーネントの標準動作に必要な初期化を実行します。
LCD_Char_Enable()	表示をオンにします。
LCD_Char_SaveConfig()	提供されている API はスリープモードに入る前に必要なデータを保存します。
LCD_Char_RestoreConfig()	提供されている API はスリープモードを終了した後で保存されたデータを復元します。

次のオプション関数は、ユーザ選択カスタム フォントを選択すると、必要に応じて組み込まれます。ユーザ定義であっても、PSoC Creator で生成されていても、LCD_Char_LoadCustomFonts() 関数には、すべてのカスタム フォント セットが付属します。LCD_Char_LoadCustomFonts() 関数は、ユーザ定義または棒グラフ文字を LCD ハードウェアにロードするために使用できます。ツールで作成されたカスタム フォントをロードする場合は、この関数を使う前に、カスタム フォントへのポインタをプロジェクトにインポートする必要があります (LCD_Char_LoadCustomFonts() の説明を参照)。デフォルトで、LCD_Char_Init() ルチンはユーザ選択カスタム フォントをロードします。棒グラフの描画コマンドは、棒グラフが選択され、棒グラフの簡易および動的な調整が有効になっている場合に生成されます。

オプションのカスタム フォント関数	説明
LCD_Char_LoadCustomFonts()	LCD モジュールに、カスタム文字をロードします。
LCD_Char_DrawHorizontalBG()	水平棒グラフを描画します。棒グラフ文字セットが選択されている場合にのみ利用できます。
LCD_Char_DrawVerticalBG()	垂直棒グラフを描画します。棒グラフ文字セットが選択されている場合にのみ利用できます。

次のオプション関数は、必要な場合に備えて含まれています。

オプションの数字とASCII変換ルーチン	説明
LCD_Char_PrintInt8()	キャラクタLCDモジュールに、8ビット値を2つのASCII文字による16進数にて表示します。
LCD_Char_PrintInt16()	キャラクタLCDモジュールに、16ビット値の4つのASCII 文字による16進数にて表示します。
LCD_Char_PrintNumber()	16ビット値の10進数値を、左揃えの ASCII 文字として表示します。

void LCD_Char_Start(void)

説明: この関数は、次のように、LCD ハードウェア モジュールを初期化します。

- 4 ビット インターフェイスを有効にする
- 表示をクリアする
- カーソルの自動増分を有効にする
- カーソルを開始位置にリセットする

また、カスタマイザの GUI で定義されている場合は、カスタム文字セットを LCD に設定します。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_Stop(void)

説明: LDC 画面の表示をオフにします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_PrintString(char8 * string)

説明: 現在のカーソル位置から、Null 終端された文字列を画面に書き込みます。

パラメータ: char8 * string: LCD モジュールの画面に、Null 終端された ASCII 文字のアレイが表示されます。

戻り値: なし

注意事項: なし

void LCD_Char_PutChar(char8 character)

説明: 現在のカーソル位置において、個々の文字を画面に書き込みます。名前付き値を通して、カスタム文字を表示するために使用されます。(LCD_Char_CUSTOM_0() through LCD_Char_CUSTOM_7().

パラメータ: char8 character: LCD モジュールの画面に表示される ASCII 文字。

戻り値: なし

注意事項: なし

void LCD_Char_Position(uint8 row, uint8 column)

説明: カーソルを、引数**行**および**列**で指定された位置に移動します。

パラメータ: uint8 row: カーソルの位置を示す行番号。最小値は 0 です。

uint8 column: カーソルの位置を示す列番号。最小値は 0 です。

戻り値: なし

注意事項: なし

void LCD_Char_WriteData(uint8 dByte)

説明: 現在の位置で、データを LCD RAM に書き込みます。書き込みの完了時に、指定した入力モードにより、位置が増分または減分されます。

パラメータ: dByte: LCD モジュールに書き込まれるバイト値。

戻り値: なし

注意事項: なし

void LCD_Char_WriteControl(uint8 cByte)

説明: LCD モジュールにコマンド バイトを書き込みます。異なる LCD モデルは、それぞれ独自のコマンドを持つことができます。そのモデルに有効なコマンドについては、特定の LCD データシートをご覧ください。

パラメータ: cByte: LCD モジュールのコマンド レジスタにロードされるコマンドを表す 8 ビット値。下の表で指定されている有効なコマンド パラメータ:

値	説明
LCD_Char_CLEAR_DISPLAY	表示をクリアする
LCD_Char_RESET_CURSOR_POSITION LCD_Char_CURSOR_HOME	カーソルと LCD をホーム位置に戻す
LCD_Char_CURSOR_LEFT	左カーソルの移動方向を設定する
LCD_Char_CURSOR_RIGHT	右カーソルの移動方向を設定する
LCD_Char_DISPLAY_CURSOR_ON	表示とカーソルを有効にする
LCD_Char_DISPLAY_ON_CURSOR_OFF	表示を有効にし、カーソルをオフにする
LCD_Char_CURSOR_WINK	表示を有効にし、カーソルをオフにして、カーソル ウィンクを設定する
LCD_Char_CURSOR_BLINK	表示とカーソルを有効にし、カーソル ブリンクを設定する
LCD_Char_CURSOR_SH_LEFT	カーソルを移動して表示の左にシフトする
LCD_Char_CURSOR_SH_RIGHT	カーソルを移動して表示の右にシフトする
LCD_Char_DISPLAY_2_LINES_5x10	表示を 2 行、10 文字に設定する

戻り値: なし

注意事項: なし

void LCD_Char_ClearDisplay(void)

説明: 画面の内容をクリアし、カーソル位置を行および列の0 にリセットします。適切な引数で LCD_Char_WriteControl() を呼び出し、表示を有効にします。

パラメータ: なし

戻り値: なし

注意事項: カーソルの位置を 0,0 にリセットします。

void LCD_Char_IsReady(void)

説明: レディ ビットが設定されるまで、LCD をポーリングします。

パラメータ: なし

戻り値: なし

注意事項: ピンを HI-Z に変更します。

void LCD_Char_DisplayOff(void)

説明: 表示をオフにしますが、LCD モジュールをリセットすることはありません。適切な引数で LCD_Char_WriteControl() を呼び出し、表示を無効にします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_DisplayOn(void)

説明: 初期化*なし*で表示をオンにします。適切な引数で LCD_Char_WriteControl() を呼び出し、表示を有効にします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_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望ましいルーチンです。LCD_Char_Sleep() ルーチンは、現在のコンポーネント状態を保存します。次に、LCD_Char_Stop() 関数と LCD_Char_SaveConfig() を呼び出して、ハードウェア構成を保存します。

CyPmSleep() または CyPmHibernate() 関数を呼び出す前に、LCD_Char_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

注意事項: コンポーネントのピンのドライブ モードは変更しません。この目的には、ポート コンポーネントの API を使用してください。キャラクタLCD は、独自のプロトコルを持つインターフェース コンポーネントであるため、コンポーネントのピン状態を保存または復元した後で、コンポーネントの再初期化が必要になります。

void LCD_Char_Wakeup(void)

説明: コンポーネントの設定を復元し、LCD をオンにします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_Init(void)

説明: コンポーネントの標準動作に必要な初期化を実行します。また、[Configure (設定)] ダイアログで定義されている場合は、LCD_Char_Init() がカスタム文字セットをロードします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_Enable(void)

説明: 表示をオンにします。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_SaveConfig(void)

説明: 提供されているAPIは、スリープ モードに入る前に必要なデータを保存します。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_RestoreConfig(void)

説明: 提供されているAPIはスリープ モードを終了した後で保存されたデータを復元します。

パラメータ: なし

戻り値: なし

注意事項: なし

void LCD_Char_LoadCustomFonts(const uint8 * customData)

説明: 8 つのカスタム文字 (棒グラフまたはユーザ定義のフォント) を LCD モジュールにロードし、実行中にカスタム フォントを使用します。カスタム文字設定がカスタマイザで選択されている場合にのみ、利用可能です。

パラメータ: Const uint8 * customData: バイト アレイの先頭へのポインタ。5x8文字には1文字につき8バイトが必要となるため、配列は 64バイト長となる必要があります。

戻り値: なし

注意事項: LCD モジュールに保存されている可能性のある、これまでのカスタム文字を上書きします。

void LCD_Char_DrawHorizontalBG(uint8 row, uint8 column, uint8 maxCharacters, uint8 value)

説明: 水平棒グラフを描画します。水平または垂直棒グラフが選択されている場合のみ、利用可能です。

パラメータ: uint8 row: 棒グラフ最初の文字の行。

uint8 column: 棒グラフ最初の文字の列。

uint8 maxCharacters: 棒グラフが消費する文字の総数。棒グラフ選択により、高さまたは幅を示します。各文字は、5 ピクセル幅、8 ピクセル高です。

uint8 value: 描画する影付きピクセルの数。棒グラフの総ピクセル長（高さ）を超えることはできません。

戻り値: なし

注意事項: なし

void LCD_Char_DrawVerticalBG(uint8 row, uint8 column, uint8 maxCharacters, uint8 value)

説明: 垂直棒グラフを描画します。水平または垂直棒グラフが選択されている場合のみ、利用可能です。

パラメータ: uint8 row: 棒グラフ最初の文字の行。

uint8 column: 棒グラフ最初の文字の列。

uint8 maxCharacters: 棒グラフが消費する文字の総数。棒グラフ選択により、高さまたは幅を示します。各文字は、5 ピクセル幅、8 ピクセル高です。

uint8 value: 描画する影付きピクセルの数。棒グラフの総ピクセル長（高さ）を超えることはできません。

戻り値: なし

注意事項: なし

void LCD_Char_PrintInt8(uint8 value)

説明: 8 ビット値を2つのASCII 文字表現によってキャラクタLCD モジュールに表示します。

パラメータ: uint8 value: 16 進数 ASCII 文字で表示される 8 ビット値。

戻り値: なし

注意事項: なし

void LCD_Char_PrintInt16(uint16 value)

説明: 16ビット値を4つのASCII文字表現によってキャラクタLCDモジュールに表示します。

パラメータ: uint16 value: 16進数ASCII文字で表示される16 ビット値。

戻り値: なし

注意事項: なし

void LCD_Char_PrintNumber(uint16 value)

説明: 16 ビット値の10進数値を、左揃えの ASCII 文字として表示します。

パラメータ: uint16 value: 10進数としてASCII 文字で表示される 16 ビット値。

戻り値: なし

注意事項: なし

ファームウェア ソースコードの例

PSoC Creator は、Find Example Project ダイアログに数多くのプロジェクト例を提供しており、そこには回路図およびコード例が含まれています。コンポーネント固有の例を見るには、Component Catalog または回路図に置いたコンポーネント 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 (スタート ページ)」または **File** 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。

詳しくは、PSoC Creator ヘルプの「Find Example Project (サンプルプロジェクトを探す)」を参照してください。

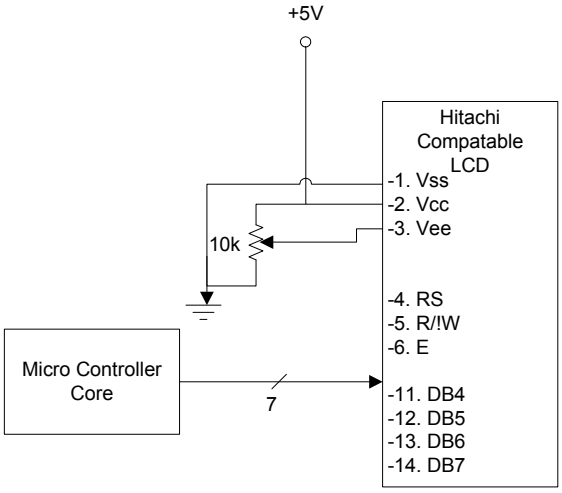
機能説明

LCD モジュールは、英数文字と限られたカスタム フォントにビジュアル表示を提供します。API が必要に応じて PSoC を設定し、標準の Hitachi LCD 表示 ドライバと PSoC 装置間で簡単なインターフェースを可能にします。

次の表は、物理 LCD モジュール ピンに対する LCD 論理ポート ピンのマッピングを示しています。LCD の論理ポートは、最初または 2 番目のポート物理ピンにマップできますが、ポートをスパンすることはできないことに注意してください。つまり、論理的に LogicalPort_0 は、ポート 2、ピン 0、またはポート 2、ピン 1 のいずれかになります。書き込みのデータ整列に必要なシフト数を軽減することで、ピン 0 で開始するために LCD 論理ポートを強制するピン エディタの使用が効果的に改善されます。

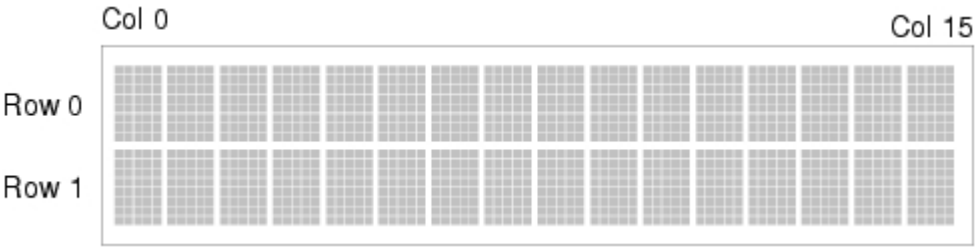
論理ポート ピン	LCD モジュール ピン	説明
LCDPort_0	DB4	データ ビット 0
LCDPort_1	DB5	データ ビット 1
LCDPort_2	DB6	データ ビット 2
LCDPort_3	DB7	データ ビット 3
LCDPort_4	E	LCD 有効化 (新しいデータが利用可能なことを確認するストローブ)
LCDPort_5	RS	レジスタの選択 (データまたは制御入力データの選択)
LCDPort_6	R/!W	読み込み/書き込みなし(LCD のレディ ビットをポーリングするための切り替え)

図 2. ピン エディタ図



LCD_Char_Position() 関数は、次のように表示のアドレス決定を管理します。0 行、0 列 は左上にあり、列番号は右に向かって増えます。4 行の表示では、0 行、20 列のアドレスを 2 行、0 列にマップするため、0 行の 19 列を超える書き込みが 2 行で破損します。これは、標準の 2x16 Hitachi モジュールでは問題ありません。

図 3. 2x16 Hitachi LCD モジュール



コンポーネントの変更

ここでは、前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
1.50.c	デフォルトのカスタマ フォントセットを復元するためにデータ シートに命令を追加しました	
1.50.b	void LCD_Char_WriteControl(uint8 command) API を削除しました。	void LCD_Char_WriteControl(uint8 command) API は API セクションで 2 回説明されているため、追加説明は省略しました。

バージョン	変更の説明	変更の理由 / 影響
	LCD_Char_DrawHorizontalBG() および LCD_Char_DrawVerticalBG() API の説明から呼び出し LCD_Char_LoadCustomFonts() の必要性についての注記を削除しました。	これでコンポーネントの LCD_Char_Start() ルーチンで実行されるため、LCD_Char_DrawHorizontalBG() または LCD_Char_DrawVerticalBG() を使用する前に LCD_Char_LoadCustomFonts() を呼び出す必要はありません。
1.50.a	データシートの LCD_Char_WriteData()、LCD_Char_WriteControl()、LCD_Char_Sleep() API の説明を更新しました	関数 LCD_Char_WriteData() のパラメータの説明がなかったため、適切な説明を追記しました。LCD_Char_Sleep() 説明がかなり不足していたため、詳細情報を加えて更新しました。LCD_Char_WriteControl() の説明は可能な入力パラメータ値の詳細により拡張されました。
	データシートのマイナーな編集と更新	
1.50	LCD_Char_Sleep()、LCD_Char_Wakeup()、LCD_Char_Enable()、LCD_Char_Init()、LCD_Char_SaveConfig()、LCD_Char_RestoreConfig() API が追加されました。	これは、低電力モードをサポートし、ほとんどのコンポーネントの共通インターフェースを提供するためです。
	スリープ モード API の宣言を含む新しい API ファイル - *CharLCD_PM.c* が追加されました。	低電力モードをサポートするためです。
	LCD_Char_Init() API に LCD_Char_LoadCustomFonts() への呼び出しが追加されました。	[Configure (設定)] ダイアログで選択したカスタム フォントは、プロジェクトで最初に LCD_Char_Start() が呼び出されたときに自動的にロードされます。
1.40	表示のタイミング要件を満たすため、LCD_Char_Start() 関数の初期化シーケンスに、遅延が追加されました。	高周波数でシステムを実行する際に存在していた、初期化の失敗を防止します。
1.30	キャラクタLCDスキーマで、デジタル ポートコンポーネントをピン コンポーネントに置換されました。	古いデジタル ポートは、使用中止になり、新しいピン コンポーネントに置換されました。
	IsReady 関数の説明が追加されました。	データシートの旧バージョンには、LCD_Char_IsReady() の説明がありませんでした。
	関数 LCD_Char_DelayUS() が削除され、LCD_Char_CyDelay() または LCD_Char_IsReady() に置換されました。	これは、一部の失敗の原因となったコンポーネントのタイミング問題を解決しました。
1.20.a	シリコン リビジョンとの互換性について知らせる情報をコンポーネントに追加しました。	このツールは、コンポーネントが不整合のシリコン上で使用された場合にエラー/警告を発します。エラーが表示されたら、対象デバイスをサポートするリビジョンにアップデートしてください。
1.20	記号の更新	企業標準に準拠するよう更新されました。
1.10.a	シリコン リビジョンとの互換性について知らせる情報をコンポーネントに追加しました。	このツールは、コンポーネントが不整合のシリコン上で使用された場合にエラー/警告を発します。エラーが表示されたら、対象デバイスをサポートするリビジョンにアップデートしてください。

バージョン	変更の説明	変更の理由 / 影響
1.10	バージョン 0.2 からのさまざまな更新	0.2 バージョンは、アルファ ビルドに含まれていましたが、完全な機能性を持つコンポーネントではありませんでした。

Copyright © 2005-2012 Cypress Semiconductor Corporation 本文書に記載される情報は、予告なく変更される場合があります。Cypress Semiconductor Corporation は、サイプレス製品に組み込まれた回路以外のいかなる回路を使用することに対しても一切の責任を負いません。特許又はその他の権限下で、ライセンスを譲渡又は暗示することもありません。サイプレス製品は、サイプレスとの書面による合意に基づくものでない限り、医療、生命維持、救命、重要な管理、又は安全の用途のために仕様することを保証するものではなく、また使用することを意図したものでもありません。さらにサイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことを合理的に予想される、生命維持システムの重要なコンポーネンツとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

PSoC Designer™及び Programmable System-on-Chip™は、Cypress Semiconductor Corp.の商標、PSoC®は同社の登録商標です。本文書で言及するその他全ての商標又は登録商標は各社の所有物です。

全てのソースコード(ソフトウェア及び/又はファームウェア)は Cypress Semiconductor Corporation (以下「サイプレス」)が所有し、全世界(米国及びその他の国)の特許権保護、米国の著作権法並びに国際協定の条項により保護され、かつそれらに従います。サイプレスが本書面によるライセンシーに付与するライセンスは、個人的、非独占的かつ譲渡不能のライセンスであって、適用される契約で指定されたサイプレスの集積回路と併用されるライセンシーの製品のみをサポートするカスタムソフトウェア及び/又はカスタムファームウェアを作成する目的に限って、サイプレスのソースコードの派生著作物を複製、使用、変更、そして作成するためのライセンス、並びにサイプレスのソースコード及び派生著作物をコンパイルするためのライセンスです。上記で指定された場合を除き、サイプレスの書面による明示的な許可なくして本ソースコードを複製、変更、変換、コンパイル、又は表示することは全て禁止されます。

免責条項:サイプレスは、明示的又は黙示的を問わず、本資料に関するいかなる種類の保証も行いません。これには、商品性又は特定目的への適合性の黙示的な保証が含まれますが、これに限定されません。サイプレスは、本文書に記載される資料に対して今後予告なく変更を加える権利を留保します。サイプレスは、本文書に記載されるいかなる製品又は回路を適用又は使用したことによって生ずるいかなる責任も負いません。サイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネンツとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

ソフトウェアの使用は、適用されるサイプレスソフトウェアライセンス契約によって制限され、かつ制約される場合があります。