携帯ゲーム機(PSP)にも使われているフルカラーTFT液晶を制御する

FRマイコン基板を使ったディジタル・フォト・フレームの設計事例 ソフトウェア編

関連データ

飯島幸太

前回(2008年9月号，pp.139-146)では，ディジタル・フォト・フレーム(DPF)の製作事例のハードウェア編として，システム概要，ハードウェアの構成や配線方法などを紹介した．今回はソフトウェア編として，液晶装置のドライブ方法やフラッシュROMへの書き込み方法，DPFアプリケーションの使い方について解説する．また，Windowsアプリケーションの利用方法やインストール手順についても紹介する．(筆者)

1．TFT液晶に表示させるには

● 液晶装置のインターフェース

液晶装置への信号線は28本で構成されます．その内訳は，RGBの画素データを送信する信号線が24本(各RGBに8本ずつ)と，水平同期信号(HSync)，垂直同期信号(VSync)，RGBの画素データが有効であるかどうかを示すイネーブル信号(DISP)，クロック信号(CK)がそれぞれ1本ずつです．一般的にRGBディジタル・インターフェース[1]と呼ばれます．この28本の信号線で液晶装置への表示を制御します．

● 液晶画面の構成

図1に液晶画面の構成を示します．全体で525×286ピクセルの領域があり，表示有効画素領域は480ピクセル×272ピクセルです．白色で示した上下左右の2ピクセル部分は，HSyncやVSyncをHIレベルにしているにも関わらず，設定されたデータが表示されない領域になります．液晶表示のソフトウェア処理を簡単にするため，この部分にもRGBデータを設定することにしました．また，有効画素領域以外の領域(HSyncおよびVSyncがLowの領域)は，RGBデータを指定しても無視されます．

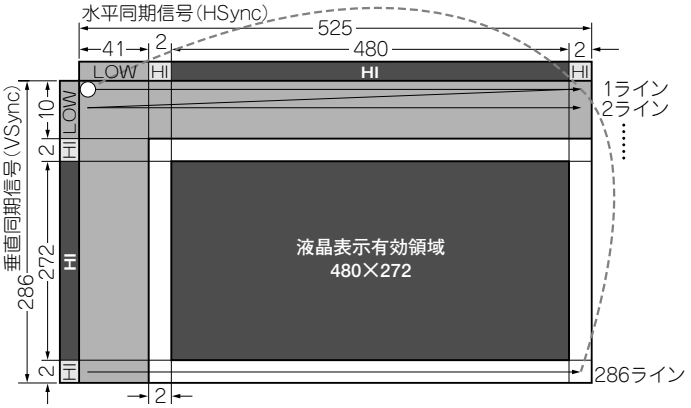

図1　画面の構成イメージ

RGBデータは，左上(白丸)を原点に左から右へ順番に送る．HSyncとVSyncをHighで設定した領域(液晶表示有効領域)は液晶画面に表示されるが，上下左右十2の領域(図中の白色)は無視される．286ラインまで送り終わったら，再度1ラインから繰り返す．この制御をメイン・ループで繰り返し実行する．

液晶画面を更新するには，525×286 = 150,150クロックを送信する必要があります．クロック周波数は7.83～9.26 MHz[2]が要求されています．

● GPIOで液晶表示をコントロール

液晶に映像を表示し続けるためには，ダイナミック駆動で全領域にデータを繰り返し送信する必要があります．例えば，7セグメントLEDをダイナミック駆動させた場合をイメージしてください．高速に表示を更新した方がチラツキが抑えられ，奇麗な表示が行われるのを容易に想像できるでしょう．つまり，マイコン側からデータを高速に伝えられれば，1画面のフレーム・レートが向上し，奇麗な映像が表示できます．

本マイコン(FR60)で，データを高速に伝えられる方法を検討した結果，最終的に落ち着いたのは，常にGPIO制御をメイン・ループ中で制御する方法でした．**リスト1**にメイン・ループから常に呼び出される処理(`Lcd_DisplayProc`関数)を示します．

FR60でGPIO出力する場合は，ビット幅が8ビットに制限されます．RGBデータ(24ビット)出力なら8ビットずつ3回に分けて出力します．また，クロック(CK)端子はデューティ比が40～60%で指定されています．デューティ比を50%に近づけるため，**リスト1**中でクロックをLow→Highにするタイミングを，Greenの色データを設定する行の下に挿入しています．さらに，液晶装置が求める周波数に達していないため，DPFの表示に問題が発生する可能性もあります(右掲の**コラム1**を参照)．

KEYWORD ── RGB画素データ，水平同期信号(Hsync)，垂直同期信号(Vsync)，有効画素領域，RGBディジタル・インターフェース，ダイナミック駆動，GPIO制御，USBターゲット，バルクOUT転送，Vectorテーブル

コラム 1　液晶装置が求めるクロック周波数は 9.00MHz

今回の製作で，フラッシュ ROM2 から画像データを読み出して，RGB データを書き込む（液晶装置へ送り出す）ときのクロック周波数は 1.06MHz（実測）でした．液晶装置が求める周波数の約 9 分の 1 です．液晶装置が求めるクロックの最低周波数は 7.83MHz なので，明らかに要求を達成できていません．クロックが遅い場合は，フリッカ・ノイズの発生などの問題が発生する可能性があると，マニュアルには記載されています．

クロック周波数を上げられるアセンブラ・コードを試しましたが，あまり効果はありませんでした．C 言語によるアセンブラの展開なども確認しながら，試行錯誤を繰り返した結果，現状の転送タイミングとなりました．もしかすると，同じ型番の液晶装置でも製造ロットが異なれば，不快なノイズの発生する可能性があります．そのような場合は，RGB データを設定するタイミングと，クロック（CK）を High/Low するタイミングを微調整してみてください．

● 内蔵フラッシュ ROM に画像データを書き込む

フラッシュ ROM に書き込む必要がある画像データ・サイズは，484 × 276 × 3 バイト = 400,752 バイト（表示有効領域：480 × 272 と上下左右の白領域：+2）です．8,192 バイト単位で画像データを書き換える[注1]ように設計したため，49 回の書き込みで 1 画像の書き換えが完了します．

リスト 2 に C 言語で記述したフラッシュ ROM 書き換え処理のプログラムを示します．フラッシュ ROM を書き換える際は，フラッシュ ROM 上のプログラムは実行できません．RAM 上で RGB データ書き込みプログラムを実行する必要があります．RGB データ書き換えのたびにプログラムを RAM にコピーして実行するのは，効率がよくありません．そのため，起動時に RAM 領域へフラッシュ ROM 書き込み用のプログラムを一度だけコピーし，呼び出しが必要なときに RAM のアドレスを指定し，関数呼び出しを行うようにしました．RGB データ書き換え中に割り込み処理が発生し，Vector テーブルを参照すると，フラッシュ ROM にアクセスするため，割り込みは禁止です．

リスト 1　液晶表示ソースの一部（Lcd_DisplayProc 関数）

```
/*-------------------------------------------------------------
 *  @brief        液晶 1 画面表示処理
 *  @par     解説:
 *-------------------------------------------------------------*/
void Lcd_DisplayProc(void)
{
  int i;
  /*******************************************************/
  /* 表示許可時のみクロック出力する                      */
  /*******************************************************/
  if(g_bLcdVisible){
/*******************************************************/
/* ■ 出力画像領域のアドレス指定（SW により切り替える）*/
/*******************************************************/
    if(PORT_SW){
      g_pData = (volatile BYTE*)0x00080000; /* Flash ROM1 */
    }else{
      g_pData = (volatile BYTE*)0x00180000; /* Flash ROM2 */
    }
    g_pDataDammy = g_pData;            /* ダミーのアドレスも同一 */
/*******************************************************/
/* ■ VSYNC を LOW にして、表示無効領域 10 ライン分送信する */
/*******************************************************/
    for(i = 0 ; i < dLCD_VSYNC_LOW_TIMING ; i++){
      Lcd_Display_Horizon_Dammy();
    }
/*******************************************************/
/* ■ VSYNC を HI にして，276 ライン分送信する　（表示有効領域）*/
/*    ※上下 2 ライン分の無視領域も含む                */
/*******************************************************/
    for(i = 0 ; i < dLCD_VSYNC_HI_TIMING ; i++){
      Lcd_Display_Horizon();
    }
  }
}

/*-------------------------------------------------------------
/**
 *  @brief        1 ライン送信（525 クロック送信）
 *  @par     解説:
*-------------------------------------------------------------*/
void Lcd_Display_Horizon(void)
{
  int i;
/*******************************************************/
/* ■ HSYNC が LOW のまま 41 回のクロック出力           */
/*    色データを設定しているが無効である　クロックのデューティ比を一定に */
/*    するためにダミー・データを同じタイミングで設定している */
/*******************************************************/
  for(i = 0 ; i < dLCD_HSYNC_LOW_TIMING ; i++){
    dLCD_DATA_R = *(g_pDataDammy+0);
    dLCD_DATA_G = *(g_pDataDammy+1);
    dLCD_CTRL_PORT = dLCD_VHI_HLO_CLK_HI;
    dLCD_DATA_B = *(g_pDataDammy+2);
    g_pDataDammy += 3;
    dLCD_CTRL_PORT = dLCD_VHI_HLO_CLK_LO;
  }
/*******************************************************/
/* 484 ピクセルのデータを転送                          */
/*******************************************************/
  for(i = 0 ; i < dLCD_HSYNC_HI_TIMING ; i++){
    dLCD_DATA_R = *(g_pData+0);          /* RED の色データ   */
    dLCD_DATA_G = *(g_pData+1);          /* GREEN の色データ */
    dLCD_CTRL_PORT = dLCD_VHI_HHI_CLK_HI; /* クロックを HI   */
    dLCD_DATA_B = *(g_pData+2);          /* BLUE の色データ  */
    g_pData += 3;
    dLCD_CTRL_PORT = dLCD_VHI_HHI_CLK_LO; /* クロックを LOW  */
                          /* HI->LOW で色が読み込まれる */
  }
}
```

Low→Highのタイミング

注 1：内蔵 RAM に 8K バイトのバッファを用意し，ホストから UART や USB で受信したデータを一時的に蓄えて，その単位で書き込み処理を実施する．フラッシュ ROM の消去はセクタ単位だが，書き込みは 1 ハーフ・ワード（2 バイト）単位で可能．