AVR287：HIDと大容量記憶のUSBホスト実演

要点

- AVR® USB OTG簡略ホストに基く
- AT90USB647/1287で動作
- ブート可/不可標準USBマウス支援
- USBハブ機能支援（大容量記憶装置のみ）
- 大容量記憶：
  - 簡略塊命令(RBC:Reduced Block Commands)
  - SFF-8020iまたはMMC-2(ATAPI)命令群
  - UFI(代表的にフロッピーディスクドライブ(FDD)装置)
  - SFF-8070i命令群
  - SCSI透過命令一式

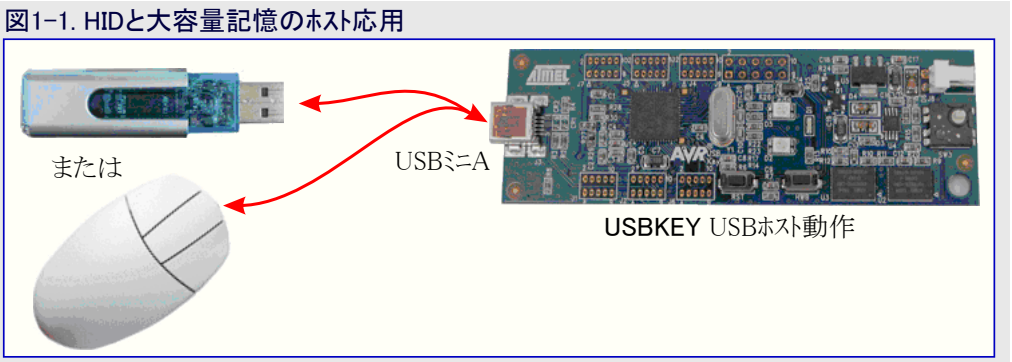
1. 序説

この頃、周辺装置は益々USBインターフェースを使っています。組み込み応用内にUSB標準装置を支援することができるUSBホスト インターフェースを持つことは大きな利益です。

この資料の狙いはUSBのHIDクラス(USBマウス)と大容量記憶クラスに基いたUSBホスト応用をどう始め、そして実装するのかを記述することです。そして最後に(HIDと大容量記憶の)両方のUSBクラスと、ファイル システム(FAT12/16/32)をも管理するAT90USB(7系)の簡単な例を紹介します。

AVR USBファームウェア枠組み(http://www.atmel.com)、HID仕様と大容量記憶の知識(http://www.usb.org)に精通していると仮定されます。

図1-1. HIDと大容量記憶のホスト応用



2. 動作の理屈

2.1. HIDクラス

2.1.1. HID構成設定

HIDクラスは1つの制御エンドポイント(EP0)、1つの割り込みINエンドポイント、そして任意選択の1つの割り込みOUTエンドポイントが必要です。通常、制御エンドポイント(EP0)は列挙(接続検出)手順を処理するためだけに用いられます。割り込みINエンドポイントは(マウス応用での釦押下のような)装置からホストへの入力(IN)報告転送に使われ、割り込みOUTエンドポイントは(キーボード応用でのLED状態のような)ホストから装置への出力(OUT)報告転送に使われます。

結論として、標準USBマウス応用は1つの既定制御エンドポイントと1つの割り込みINエンドポイントが必要です。

2.1.2. データ転送

USB HID(USBマウス)応用はホストとマウス間での簡単なデータ交換です。

ホストは各P時間(ポーリング間隔時間)で利用可能な新しいデータがあるかをマウスに問い、マウスはそれが利用可能ならばデータを送り、さもなければ利用可能なデータが無いことをホストへ告げるためにNAK(No Acknowledge)を送ります。

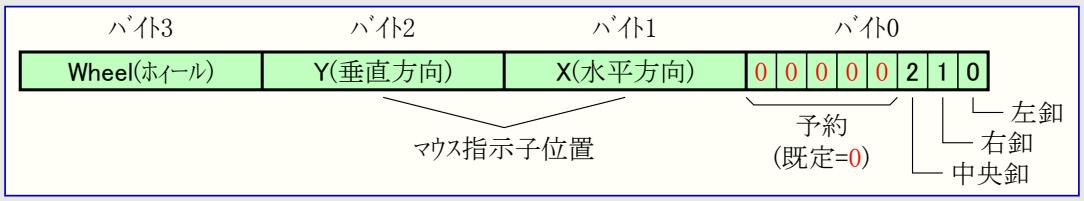
ホストへ送られるデータは'報告'と呼ばれます。この報告は以下の構造を持ちます。

8ビット AVR® マイクロ コントローラ

応用記述

Rev. 8229A-09/09, 8229AJ2-05/21

図2-1. USB'報告'構造



釦が押される、またはﾏｳｽが移動される毎に、この報告がﾎｽﾄへ送られます。これらのﾊﾞｲﾄはこの順(ﾊﾞｲﾄ0⇒ﾊﾞｲﾄ3)で読まれます。

2.2. 大容量記憶ｸﾗｽ

2.2.1. 大容量記憶構成設定

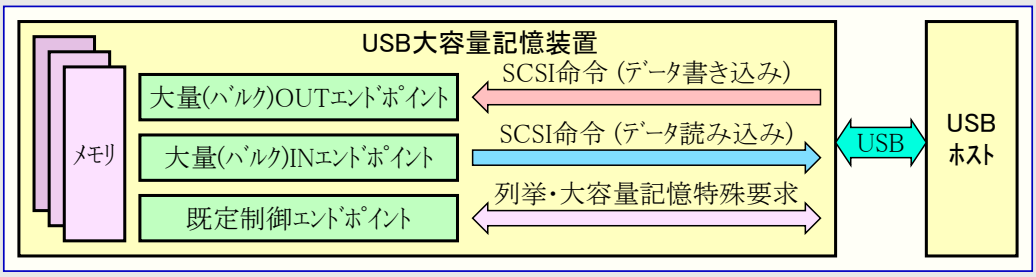
大容量記憶応用は状態とﾃﾞｰﾀの転送を実行するのに2つの大量(ﾊﾞﾙｸ)ｴﾝﾄﾞﾎﾟｲﾝﾄ(1つのINと1つのOUT)を使います。ｴﾝﾄﾞﾎﾟｲﾝﾄ0(制御ｴﾝﾄﾞﾎﾟｲﾝﾄ)は列挙(接続検出)、異常管理、論理装置番号(LUN:Logic Unit Number)値を決めるためだけに使われます。

2.2.2. ﾃﾞｰﾀ転送

大容量記憶応用に関するUSBﾃﾞｰﾀ交換はSCSI(Small Computer System Interface)命令に基きます。言い換えれば、大容量記憶応用はﾌｧｲﾙ転送を管理するためにﾎｽﾄによって送られる一式のSCSI命令です。

大容量記憶ｸﾗｽはLUNにより、同時に多数の記憶装置の管理を装置に許します。

図2-2. USB大容量記憶応用概要

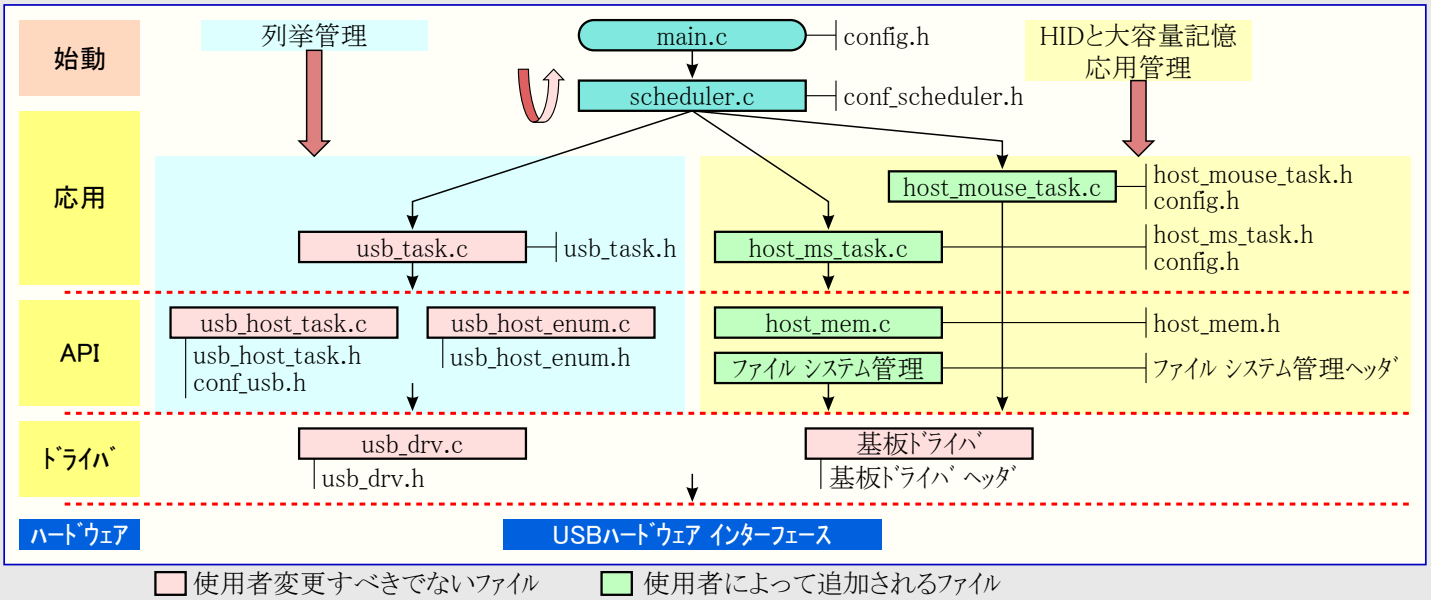


より多くの詳細情報については「USB大容量記憶ｸﾗｽ大量(ﾊﾞﾙｸ)のみ転送仕様」(http://www.usb.org)を読んでください。

3. Atmel® ｿﾌﾄｳｪｱ基本設計

以下はHIDと大容量記憶のﾎｽﾄ ﾌｧｰﾑｳｪｱの基本設計概要で、操作で必要とする全てのﾌｧｲﾙが出ています。

図3-1. HIDと大容量記憶のﾎｽﾄ基本設計



HIDクラス(マウス)の管理は"host_mouse_task.c"ファイルで実装され、一方大容量記憶クラスは"host_ms_task.c"ファイルで実装されます。計画機構がusb_task(void)、host_mouse_task(void)、host_ms_task(void)の関数を周期的に呼び出します。これら3つの関数の動作は以下に記述されます。

■ usb_task(void)
- USB動作形態検出
- USBホスト/装置列挙(接続検出)

■ host_mouse_task(void)
- マウスが接続されたかを調査
- マウスからデータを得て、使用者応用を実行
- マウスがホストから取り去られたかを調査

■ host_ms_task(void)
- 大容量記憶装置が接続されたかを調査
- 使用者応用を実行
- 大容量記憶装置がホストから取り去られたかを調査

3.1. 列挙(接続検出)

装置がホストへ接続した時に列挙(接続検出)が始まります。装置インターフェースがUSBホスト ファームウェア低位作業によって受け入れられた場合、ホストは("conf_usb.h"ファイルで定義される)支援インターフェースの一覧と装置記述子を比較し、そしてhost_mouse_task()とhost_ms_task()は接続通知に遭遇します(Is_new_device_connection_event()マクロがTRUEを返します)。

conf_usb.hファイルでUSB_HUB_SUPPORTが許可の場合、現在の受け入れられた装置数はGet_nb_device()関数によって返され、現在の受け入れられたインターフェース数はGet_nb_supported_interface()関数によって返されます。各装置の各インターフェースに対して、プログラムはGet_class(i)、Get_subclass(i)、Get_protocol(i)マクロによって装置のクラス、補助クラス、規約符号をアクセスします。

プログラムは接続された装置が支援するクラスのインターフェースを持つかを調べ、装置インターフェースが受け入れられるなら、conf_usb.hファイルで"HOST_AUTO_CFG_ENDPOINT"が許可されているか否かのどちらかに依存して"host_auto_configure_endpoint"関数または"User_configure_endpoint()"関数によってパイプを形成します。

構成設定完了時、ホストは装置へSet_configratiion()要求を送り、DEVICE_READY状態に戻ります。

ホスト マウス作業については、新規の装置接続通知を得た時に、それがマウス接続かを調べるためにクラスと規約を比較します。そうなら、INパイプを解除してデータ転送の準備をします。

以下が対応する関数のコードです。

コード3-1. ホストHIDマウス装置検出

```
if(Is_new_device_connection_event())                                    // 装置接続
{
   mouse_connected=0;
   for(i=0;i<Get_nb_supported_interface();i++)
   {
      if(Get_class(i)==HID_CLASS  && Get_protocol(i)==HID_PROTOCOL_MOUSE)
      {
         mouse_connected=1;
         host_hid_set_idle();
         host_get_hid_report_descriptor();
         LOG_STR_CODE(log_mouse_connect);
         PIPE_MOUSE_IN=host_get_hwd_pipe_nb(Get_ep_addr(i,0));
         Host_select_pipe(PIPE_MOUSE_IN);
         Host_continuous_in_mode();
         Host_unfreeze_pipe();
         break;
      }
   }
}
```

大容量記憶作業に関しては、それが新規装置接続事象に遭遇する時に、"複数大容量記憶装置"機能を支援するため、各装置の各インターフェースを検索します。現在のインターフェースが大容量記憶クラスなら、dms[n]装置大容量記憶配列内に現在の装置指標を格納してdms_connected変数に最大大容量記憶装置数を記録します。その後、INパイプとOUTパイプを構成設定して全てのUSBドライブ(大容量記憶装置)を初期化します。

以下が対応する関数のコードです。

コード3-2. ホスト大容量記憶装置検出

```
if(Is_new_device_connection_event())
{
   for(k=0;k<=Get_nb_device()-1;k++)
   {
      Host_select_device(k);
      new_dms=TRUE;
      for(i=0;i<Get_nb_supported_interface();i++)
      {
         if(Get_class(i)==MS_CLASS)
         {
            LOG_STR_CODE(log_ms_connect);
            if (dms_connected!=0) // 他のDMS接続?
            {
               for(n=0;n<USB_MAX_DMS_NUMBER;n++)
               {
                  if(dms[n].device_index==k)
                  {
                     new_dms=FALSE;
                  }
               }
            }
            if(new_dms)
            {
               dms_connected++;  // TODO：USB_MAX_DMS_NUMBER検査
               dms[dms_connected-1].device_index=k;
// 大容量記憶IN/OUTｴﾝﾄﾞﾎﾟｲﾝﾄに関連付ける正しい物理ﾊﾟｲﾌﾟ取得
               if(Is_ep_addr_in(Get_ep_addr(i,0)))
               {  // 真なら、大容量記憶INﾊﾟｲﾌﾟに関連付け
                  dms[dms_connected-1].pipe_in=usb_tree.device[k].interface[i].ep[0].pipe_number;
                  dms[dms_connected-1].pipe_out=usb_tree.device[k].interface[i].ep[1].pipe_number;
               }
               else
               {  // 偽なら、逆で関連付け
                  dms[dms_connected-1].pipe_in=usb_tree.device[k].interface[i].ep[1].pipe_number;
                  dms[dms_connected-1].pipe_out=usb_tree.device[k].interface[i].ep[0].pipe_number;
               }
// 接続された大容量記憶装置内の論理装置数(LUN)取得
               Select_dms(dms_connected-1);
               dms[dms_connected-1].nb_lun=host_get_lun();
               // USB論理装置番号比較
               lun=0;
               for(n=0;n<dms_connected-1;n++)
               {
                  lun +=dms[n].nb_lun;
               }
               // 全USB装置初期化
               for(n = 0; n < dms[dms_connected-1].nb_lun; n++)
               {
                  host_ms_inquiry();
//                host_read_format_capacity(lun); // いくつかの装置が容量読み込み命令前に本命令が必要
                  host_read_capacity(lun, &capacity);
                  host_ms_request_sense();
                  while (CTRL_GOOD != host_test_unit_ready(lun));
                  host_read_capacity( lun, &capacity );
                  lun++;
               }
               break;
            }
         }
      }
   }
}
```

3.2. データ転送

3.2.1. ホストHIDマウス作業

マウスが接続された場合、その作業は受信した有効なIN一式があるかを見るために毎回それに移行して調べます。データが利用可能な時にプログラムはHost_read_byte()関数によってデータを読み、使用者関数を実行します。その後、ホストはHost_send_in()関数によってIN命令を送り、次のデーター一式に備えます。

コード3-3. ホストHIDマウス データ転送

```
Host_select_pipe(PIPE_MOUSE_IN);
if(Is_host_in_received())
{
   if(Is_host_stall()==FALSE)
   {
      i=Host_read_byte();
      new_x=(S8)Host_read_byte();
      new_y=(S8)Host_read_byte();
      if(new_x==0)
      { Led0_off(); Led1_off();}
      else if(new_x>0)
      { Led0_on(); Led1_off();}
      else
      { Led0_off(); Led1_on();}

      if(new_y==0)
      { Led2_off(); Led3_off();}
      else if(new_y>0)
      { Led2_on(); Led3_off();}
      else
      { Led2_off(); Led3_on();}
   }
   Host_ack_in_received();
   Host_send_in();
}
```

これらの操作は評価目的用に実装されています。使用者は"現状そのまま"として現在の関数を使えますが、自身のデータ処理を実装するのも自由です。

3.2.2. ホスト大容量記憶作業

先に言及したように、大容量記憶応用はファイル転送を管理するためにホストによって送られる一式のSCSI命令です。基本的なデータ転送はhost_get_data()関数とhost_send_data()関数によって実行します。

使用者応用はSCSI符号化部、ファイル システム復号部を実装すべきです(AVR144応用記述を参照してください)。

4. 例

4.1. 概要

この理屈の全てが複合して表れるかもしれないので、ここは評価目的でUSBホストHIDクラスとホスト大容量記憶クラスの実装についての素早い評価を許すための簡単な例です。

この構成設定に於いて、実演基板は最初にUSBインターフェースによってPCに接続され、USB大容量記憶装置として働き、実演データを準備します。そして、実演基板はPCから切断され、大容量記憶データ転送を見るために大容量記憶装置に接続されます。USBホストHIDマウス応用を見るため、実演基板に標準USBマウスを接続することもできます。

USBホストHIDと大容量記憶の応用はAtmelのウェブサイト(http://www.atmel.com)で入手可能なソフトウェア一式を使うことによってホスト能力を持つどのAVR USBにも実装することができます。

4.2. ハードウェア

両ソフトウェア一式は利用可能なスタータ キットで走行することができます。執筆時点で、ホストHIDマウスと大容量記憶の一式は(AT90USB647/1287を特徴とする)STK®526またはUSBKEYの一式で走行することができ、実演基板に接続された大容量記憶装置はUSBフラッシュディスクまたは大容量記憶装置ファームウェアが走行するスタータ キットの1つになり得ます。

ホスト基板は(ミニAプラグが挿入される)ホスト動作で接続されるべきです。USB装置基板に電力を供給するために、外部電源が必要とされ、そして構成設定されます。

4.3. ソフトウェア

4.3.1. 動作説明

USBホストHIDマウス作業に関し、標準USBマウスを実演基板に接続した場合、マウスの動きがLEDで見えます。

一旦列挙(接続検出)されると、マウス移動または釦クリック毎にIN事象が起動されます。ホストはUSBインターフェースによってデータを読みます。X軸に対する動きはLED0とLED1を点灯し(X>0でLED0点灯、X<0でLED1点灯)、一方Y方向はLED2とLED3を点灯します(Y>0でLED2点灯、Y<0でLED3点灯)。

USBホスト大容量記憶作業に関し、一旦列挙(接続検出)されると、ファイルは基板のデータ フラッシュ ディスクと大容量記憶装置間で交換することができます。ジョイスティックの右方向は大容量記憶装置の"OUT"ディレクトリの内容を読み、基板のデータ フラッシュの"IN"ディレクトリにそれを書くことを許し、データ フラッシュから大容量記憶装置への"左"に関するその逆もです。

4.3.2. 構成設定

正しい動作を保証するために各マイクロ コントローラに於いていくつかの項目が定義されなければなりません。ソフトウェア一式は変更の必要がなく、"現状そのまま"として動き、以下で与えられた値で構成設定されます。

■ "conf_usb.h"ファイルに於けるUSB構成設定

- USBホスト機能を支援するにはUSB_HOST_FEATUREが許可されなければなりません。
- PCへの接続を支援するにはUSB_DEVICE_FEATUREが許可されなければなりません。
- HOST_STRICT_VID_PID_TABLEが許可の場合、VID_PID_TABLE配列は支援するVIDとPIDを含まなければなりません。
- CLASS_SUBCLASS_PROTOCOL配列はHIDクラス マウス規約(全ての標準USBマウスを支援するためにブート可とブート不可の両補助クラスが含まれるべきです。)と大容量記憶インターフェース(大容量記憶クラス、SCSI補助クラス、大量(バルク)のみ規約)を含まなければなりません。これらのクラス/補助クラス/規約の全ては必要とされる場合、HUBに関して使用者によって追加される必要があります。
- 多数の大容量記憶装置の使用を望む場合、USB_USB_SUPPORTが許可されるべきです。
- HOST_STRICT_VID_PID_TABLEは許可または禁止のどちらにもすることができます。各種供給者からの各種製品を支援するために、このマクロは禁止されることが推奨されます。

■ USB大容量記憶構成設定

- USBホスト大容量記憶の例プログラムを動かすために"config.h"ファイルのHOST_SYNC_MODEが許可されなければなりません。

5. 結び

AtmelのUSB解決策は広範囲の応用に関して強力且つ簡単なシステムを提供でき、そしてそれは必要な応用記述で使われる統合されたソフトウェア枠組みで、お客様に対してほぼ全ての市場の需要を網羅し、同時に売出しまでの最短時間にすることができます。

6. 関連文献

- AVR USB製品データシート
- AT90USBxxxマイクロ コントローラ用USBソフトウェア ライブラリ (AVR276)
- USBマウス実演 (AVR270)
- USB大容量記憶実演 (AVR273)
- USB HIDクラス仕様
- USB大容量記憶クラス仕様
- AT32UC3x, AT90USBx, ATmega32U4用Atmelファイル システム管理の使い方 (AVR114)

以下で入手可能です。

http://www.atmel.com

http://www.usb.org

本社

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
TEL 1(408) 441-0311
FAX 1(408) 487-2600

国外営業拠点

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
TEL (852) 2245-6100
FAX (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
TEL (33) 1-30-60-70-00
FAX (33) 1-30-60-71-11

Atmel Japan
104-0033 東京都中央区
新川1-24-8
東熱新川ﾋﾞﾙ 9F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ株式会社
TEL (81) 03-3523-3551
FAX (81) 03-3523-7581

製品窓口

ｳｪﾌﾞｻｲﾄ
www.atmel.com

技術支援
avr@atmel.com

販売窓口
www.atmel.com/contacts

文献請求
www.atmel.com/literature

本応用記述はAtmelのAVR287応用記述(doc8229.pdf Rev.8229A-09/09)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。